

北京师范大学

# 硕士学位论文

论文题目：多核芯片的温敏低功耗调度研究

作 者： 闫佳琪

导 师： 骆祖莹 副教授

系别年级： 信息科学与技术学院

学 号： 201121210009

学科专业： 通信与信息系统

完成日期： 2014年 4月

北京师范大学研究生院

## 北京师范大学学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：

日期： 年 月 日

## 关于论文使用授权的说明

学位论文作者完全了解北京师范大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属北京师范大学。学校有权保留并向国家有关部门或机构递交论文的复印件和电子版，允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，可以允许采用影印、缩印或其它复制手段保存、汇编学位论文（保密学位论文解密后执行此规定）。

保密论文注释：经本人申请，学校批准，本学位论文定为保密论文，密级： ，期限： 年  
自 年 月 日起至 年 月 日止，解密后适用本授权书。

非保密论文注释：本学位论文不属于保密范围，适用本授权书。

本人签名： \_\_\_\_\_ 日期： \_\_\_\_\_

导师签名： \_\_\_\_\_ 日期： \_\_\_\_\_

# 多核芯片的温敏低功耗调度研究

## 摘要

当前，面向复杂应用的高性能片上系统为了规避和减轻功耗墙问题，延续摩尔定律，采用了实时温度功耗管理与多核并行计算结构两种主要的技术手段。

实时功耗温度管理(DPTM)通过对任务的准确预测与合理调度，可以有效地降低片上系统的运行能耗与峰值温度。为了获得更好的实时功耗温度管理调度效果，基于精确任务负载预测模块，本文在第一部分提出了一种新颖的任务调度算法VP-TALk，并结合已有的多种流行调度算法，构建了一个完整的实时功耗温度管理原型系统。本文首先分析并总结已有的三种较为流行的实时系统调度算法。在此基础上，提出VP-TALk算法。该算法通过计算出最优电压-频率对的理想值，进而选择出两组与理想值相邻的电压-频率对，获得两个现实的工作状态。VP-TALk算法将任务负载分配到这两个工作状态，考虑核心温度和任务实时性的条件，以获得最优的实时功耗温度管理效果。最后基于简单地机器学习方法，我们综合四种源算法、构建了一套完整的实时功耗温度管理原型系统。实验结果表明：当本文系统的任务预测组合模块能达到平均误差为2.89%的情况下，在相同的设定峰值温度约束下，与已有调度算法的能耗值相比，尽管假设了更为敏感的功率-温度影响关系，但对于较高的工作负载率，本文所提出的VP-TALk调度算法仍能够获得平均14.33%的能耗降低；本文所提出实时功耗温度管理原型系统可以获得接近于理想的能耗优化效果。

多核片上系统(MPSOC)的低功耗设计与实时功耗温度管理(DPTM)是目前较为热点的研究问题。本文采用了一种自下而上的建模方法，对MPSOC结构级热分析方法进行了研究，提出了三种具有不同算法复杂度与精度的热分析方法：模块级方法BlockTAM、核级方法CoreTAM、考虑本核内模块相互影响的改良核级方法BlockInsideCoreTAM。这三种算法均具有简单、高效、与现有简化模型兼容、易于扩展、能够解决温度对漏电流的影响等优点。实验数据表明：对核数较多MPSOC进行热分析的时候，CoreTAM算法的复杂度低但精度也低，BlockTAM算法的精度高但复杂度也高，而BlockInsideCoreTAM则具有算法复杂度低和精度高的优点，其中局部热点的温度增量平均误差可以控制在4%以下，热分析的速度实现了近50倍以上的分析加速，两者均是较为理想的结构级热分析方法；

采用BlockInsideCoreTAM等增量式的热分析建模方法，可以快速进行用于降低热点温度的MPSOC布图规划过程。

**关键词：** 实时功耗管理；实时温度管理；任务调度；热分析；多核片上系统；体系结构级分析

# **On Temperature and Power Management for MPSoC**

## **ABSTRACT**

To tackle the Power Wall problem on high performance chip processors oriented for heavy-load applications and continue Moore's Rule, Dynamic Power and Temperature Management (DPTM) and Multi-chip Processor Architecture are adopted in the field of Integrated Circuits (IC) design.

Optimal DPTM methods can effectively cut down the soaring power consumption and alleviate the problem of chip temperature. In order to get better scheduling results, this paper mainly accomplish three things. First, with principles derived from analyzing three previous methods as thumb rules, we obtain an improved DPTM algorithm, named VP-TALK, that carefully schedule the processor's running and dormant behaviors. Besides, we propose a combined predicting model. It may predict the workload on the chip so as to draw out optimal but unpractical frequency (F) and voltage (V). This F & V pair decides two distinct pairs of F & V, with which VP-TALK schedules the processor according to both the core temperature and remaindering work load. Finally, combining the workload prediction method and four DPTM algorithms, we further build a DPTM control system. Even though our model assume a tighter and more sensitive relationship between energy and temperature, experiments show that: 1) the workload prediction's error is as less as 2.89%; 2) under even more tough assumptions about thermal and power interrelation and the same peak temperature ceiling value, our proposed DPTM algorithm gains averagely 14.33% energy saving comparing to previous algorithms when the workload ratio is comparatively high; 3) comprehensive DPTM control system's managing effect is near to the most ideal one.

Efficient thermal analysis plays a key role in the temperature-aware floorplan design for Multi-Processor System-on-Chip (MPSoC) and DPTM. This paper adopts bottom-up modeling method to study architecture-level MPSoC thermal analysis method. First extract relative thermal resistance between functional modules with HotSpot software, then, based on these parameters, we propose three analysis methods with different accuracy and algorithm complexity: Block-level Temperature Analysis Method (BlockTAM), Core-level Temperature Analysis Method (Core-

TAM) and Block Inside Core Temperature Analysis Method (BlockInsideCoreTAM). Experiment shows that BlockTAM and BlockInsideCoreTAM substantially reduce the time for MPSoC thermal analysis with guarantee of accuracy: speedup as high as 100 times is achieved with average temperature delta error as low as 3%. Both are ideal system-level analysis method.

**KEY WORDS:** Dynamic Power Management, Dynamic Temperature Management, Task Scheduling, Thermal Analysis, MPSoC, Architecture Level

# 目 录

摘 要 .....	I
ABSTRACT .....	III
插图索引 .....	VII
表格索引 .....	VIII
1 研究前言 .....	1
1.1 研究背景与相关科学问题 .....	1
1.2 目前已有研究成果及其缺陷 .....	1
1.3 本文内容及其贡献 .....	2
1.4 文章结构安排 .....	3
2 基于任务精确预测的实时功耗温度管理 .....	4
2.1 实时系统的工作负载模型 .....	4
2.2 实时系统的热分析模型 .....	4
2.3 实时系统的功耗分析模型 .....	4
2.4 目前流行的动态温度与功耗调度策略 .....	5
2.4.1 考虑漏电流的温度敏感技术（简称TALk技术） .....	5
2.4.2 基于固定周期的技术（简称Pattern-Based, PB技术） .....	6
2.4.3 M阶震荡调节技术（简称M-Oscillating, MO技术） .....	6
2.4.4 小结与评价 .....	6
2.5 基于电压预测的温敏漏电流技术（简称VP-TALK技术） .....	8
2.6 动态温度与功耗管理系统 .....	9
2.6.1 启发性示例 .....	9
2.6.2 基于时间序列分析的工作负载预测 .....	10
2.6.3 基于在线评价的动态温度与功耗管理系统 .....	12
2.6.4 基于单一调度策略的动态温度与功耗管理系统 .....	13
3 动态温度与功耗管理系统的模拟验证 .....	14
3.1 模拟实验环境 .....	14
3.2 VP-TALK算法的动态温度与功耗管理效果验证 .....	14
3.3 基于在线评价的动态温度与功耗管理系统的效果评测 .....	16
3.4 本章总结 .....	19
4 考虑温度对漏电流功耗影响的MPSoC结构级热分析算法 .....	21
4.1 多核芯片热分析的研究对象建模 .....	21

4.1.1	芯片的散热系统 .....	21
4.1.2	多核架构及其电热分布 .....	22
4.1.3	芯片热分析及HotSpot模块级模型 .....	24
4.1.4	电热耦合效应：温度对漏电流功耗的影响 .....	26
4.2	3种MPSoC结构级热分析算法 .....	27
4.2.1	模块级热分析算法（简称BlockTAM算法） .....	27
4.2.2	核级热分析方法（简称CoreTAM算法） .....	29
4.2.3	考虑核内模块相互影响的改良核级热分析方法（简称BlockInsideCoreTAM算法） .....	30
5	MPSOC结构级热分析算法的验证实验 .....	32
5.1	验证实验平台 .....	32
5.2	验证实验所用到的测例 .....	32
5.3	针对MPSOC结构级热分析算法精度的实验数据与分析 .....	34
5.4	针对MPSOC结构级热分析算法速度的实验数据与分析 .....	38
5.5	小结 .....	39
6	研究总结与展望 .....	40
6.1	本文总结 .....	40
6.2	未来工作展望 .....	41
	参考文献 .....	43
	学术成果 .....	46
	致谢 .....	47

## 插图索引

图 2.1 基于电压预测的温敏漏电流技术的流程图 .....	8
图 2.2 各种调度算法的能耗效果随负载率变化的比较 .....	10
图 2.3 各种调度算法的峰值温度效果随负载率变化的比较 .....	11
图 3.1 各种算法的能耗调度效果比较 .....	15
图 3.2 各种算法的峰值温度调度效果比较 .....	15
图 3.3 基于在线评价的动态温度与功耗管理系统VS各种源算法均值（能耗调度效果） .....	17
图 3.4 基于在线评价的动态温度与功耗管理系统VS源算法均值（峰值温度调度效果） .....	17
图 3.5 基于在线评价的动态温度与功耗管理系统VS理想调度效果（能耗调度效果） .....	18
图 3.6 基于在线评价的动态温度与功耗管理系统VS理想调度效果（峰值温度调度效果） .....	18
图 4.1 IC散热系统的结构示意 .....	21
图 4.2 Intel公司FCLGA10封装的散热系统结构 .....	22
图 4.3 Alpha21264芯片的物理布局 .....	23
图 4.4 HotSpot对芯片的电路等效热分析模型 .....	25
图 4.5 Alpha21264芯片的温度分布 .....	26
图 4.6 考虑电热耦合效应的多核芯片最高温度与静态功耗的迭代求解 .....	27
图 4.7 多核实时功耗温度管理的单模块与单核热分析简化等效电路模型 .....	29
图 5.1 测例二i7-3960X芯片布局 .....	33
图 5.2 测例二i7-3960X核内的布局 .....	34
图 5.3 对测例1中各模块的静态功耗计算误差 .....	37
图 5.4 对测例1中各能模块峰值温度的计算误差 .....	38

## 表格索引

表 2.1	$A, B, \alpha, \beta, \gamma, \delta, \mu, \eta$ 的取值 .....	5
表 3.1	VP-TALK算法调度效果与已有算法的比较 .....	16
表 4.1	导热材料的热设计参数 .....	22
表 5.1	本文测例各功能模块的面积与功耗参数设定 .....	36
表 5.2	多核芯片核内局部热点的温度计算误差对比 .....	36
表 5.3	多核芯片各核静态功耗的计算误差对比 .....	37
表 5.4	1000组热分析各算法计算耗时及加速倍数X对比 .....	37

# 1 研究前言

## 1.1 研究背景与相关科学问题

当前，面向复杂应用的高性能片上系统为了规避和减轻功耗墙（Power Wall）问题[1]，延续摩尔定律[2]，采用了两种主要的技术手段。首先，必须在芯片运行中，通过合理任务调度来降低芯片的运行能耗和峰值工作温度。因此，对芯片进行实时功耗温度管理(DPTM)的算法研究就具有重要的理论意义与广阔的应用前景，是目前电子设计自动化(EDA)研究的一个热点问题。最初为了降低芯片运行功耗、延长设备电池的使用寿命，研究人员提出了运用动态电压调节技术（DVS）对系统动态功耗进行实时功耗管理（DPM）[3–6]。然而，随着IC进入纳米工艺，漏电流静态功耗已经超过动态功耗，成为芯片功耗的主要来源，而且漏电流和工作温度之间存在指数关系[7–9]，如对于65nm工艺，当温度从60摄氏度增加到80摄氏度，芯片漏电流会增加21%。其次，目前IC业界已经普遍采用多核并行计算结构来提升芯片性能（通量）、降低设计复杂度。采用多核并行计算架构的多核片上系统(MPSoC)带来了热点分散的问题，即每个核都会产生一个局部热点[10]。为了将MPSoC多个热点的温度控制在一个安全阈值内，必须在设计与运行阶段，以功能模块与处理器核为单位，对芯片的功耗分布[11–13]与任务调度[14–16]进行优化，为此需要在结构级对芯片进行快速准确的热分析[13,14,17]。鉴于纳米工艺CMOS器件的漏电流随着工作温度的升高而指数增加，漏电流功耗与温度之间存在直接的依赖关系，即电热耦合效应[14]。为了提高分析的精度，必须在结构级热分析方法研究中考虑电热耦合效应[14–16,18]。

## 1.2 目前已有研究成果及其缺陷

针对任务调度领域，研究人员开始针对微处理器和大型服务器系统进行实时温度管理（DTM）[19–21]。为了对片上系统进行功耗、温度的统一调度与管理，最近开始出现了实时功耗温度管理(DPTM)的研究报道[22–25]，在考虑漏电流、温度相互作用关系和实时任务的时间限制这两个前提下，采用不同的DPTM策略来达到最小化运行能耗的目的。在DPTM研究中，为了提高DPTM系统的降温降耗效果，必须对系统的任务负载进行精确的预测，事实上，任务负载的轻重决定了不同方法的DPTM效果。对于多核芯片的

热分析，受惠于电热分析的相似性，可以采用有限差分方法(PDF)可以进行全芯片三维热分析，获得温度分布的精确解[26]；为了考虑温度对功耗的影响，可以采用迭代方法来逼近最后的精确解[27]。基于PDF求解的HotSpot是目前广泛采用的热分析工具软件，能够用于MPSoC的结构级热分析，也能够对电热耦合效应进行求解[27]。尽管PDF方法可以获得高精度的求解方案，但这类方法的算法复杂度非常高，不满足MPSoC布图规划和实时功耗温度管理对结构级快速求解的需求[13,17]。为了对结构级设计的温度分布进行快速求解，出现过多种加速算法[10,11,14,15,17,18]。文献[11]采用最简单的物理距离模型，速度最快、精度最差，无法进行精确的MPSoC温度求解。文献[10,14,15]省略了核间的侧向热阻、来简化温度求解，其优点是速度快，缺点是降低了求解的精度。文献[28]采用基于学习的自回归算法进行在线温度分析，提高热分析速度的同时、也降低了求解的精度。总之，求解加速的代价是降低了求解的精度。为了考虑温度对功耗的影响（LDT），精确的求解算法必须采用迭代的方法进行逼近求解[13]。在现有结构级热分析算法中，为了提高求解速度，文献[10]没有考虑LDT，文献[15]采用线性模型来拟合LDT，文献[14,16]采用分段拟合系数矩阵来求解LDP效应，其结果会带来求解精度不同程度的降低。

### 1.3 本文内容及其贡献

为了弥补上文指出的已有研究的不足之处，本文对温度敏感的实时功耗调度和多核芯片的热分析方法这两个不同领域，分别做了较为深入的研究，并取得了如下成果。首先，为了构建一个高效的DPTM系统，本文不仅提出了一种具有高精度的组合式任务预测方法，而且还提出一种新的DPTM任务调度算法VP-TALK，并进一步集成了一个基于负载预测的DPTM原型系统，该系统主要包括工作负载预测、任务实时调度两大模块。

(1)基于组合任务预测方法的负载预测模块：根据频率范围，先将对应于复杂应用的任务分为随机/周期/趋势三种组分，然后采用灰色模型/傅里叶模型/RBF神经网络模型分别对这三种组分进行精确分析，最后将三部分预测结果合成为复杂任务的预测值。  
(2)基于多种调度算法的实时调度模块：先根据对工作负载率的精确预测值、计算出最优工作状态的电压/频率理想值，再从系统的电压/频率对的实际设定值中选取相邻的两个工作状态，最后考虑系统实时性、温度上限限制、静态功耗与温度的敏感关系以及芯片模式切换代价等多种因素，利用机器学习的方法，选择一种最优的调度策略。大量的模拟实验表明，(1)在负载预测方面，本文实时功耗温度管理系统所采用的组合任务预测方法胜过众多的相关模型及算法，平均误差仅为2.89%；(2)在节能效果方

面，当负载率高于55%时，基于相同的峰值温度约束，本文所提出的VP-TALK算法分别比Pattern-based、M-oscillating和TALK对比算法节能约20.5%、11.0%、11.5%；(3)本文实时功耗温度管理原型系统的调度效果接近于理想调度效果。

其次，本文采用自下而上的策略，使用HotSpot提取MPSoC功能模块之间的热相关系数，建立了模块级热分析方法BloTAM；如图2所示，每个核内只产生一个热点，我们可以仅依靠热点之间的热相关系数、建立一个算法复杂度非常低的核级热分析方法CoreTAM；为了提高CoreTAM的精度，我们进一步提出了考虑本核内模块相互影响的改良核级方法BlockInsideCoreTAM。与现有的结构级热分析算法相比，本文所提出的三种方法具有简单、高效、与现有简化模型兼容、易于扩展、考虑LDT影响等优点，可以满足温敏MPSoC设计对高效、精确的结构级热分析方法的需求。与HotSpot软件的实验结果相比，本文方法的实验数据表明：(1)热分析精度方面而言，对多核心MPSoC进行局部热点温度分析的结果表明，BlockTAM和BlockInsideCoreTAM仅产生4%以下的温度增量平均误差(2)热分析速度方面而言，在对具有16核心的CPU进行温敏布图规划的热分析过程中，BlockTAM和BlockInsideCoreTAM可以提供50倍左右的计算加速，10倍以上的运算加速效果

## 1.4 文章结构安排

文章结构安排如下：第二章介绍基于高精度组合式任务预测方法的DPTM原型系统，调度对象仅限于单一处理器。第四章给出大量模拟实验数据，以证实该调度系统在降低功耗和温敏控制上的优越性。第四章将研究对象扩展为多核处理器，提出三种结构级热分析方法。为衡量这三种热分析计算模型的精确度与加速效果，第五章中设计了若干实验测例，并给出了模拟热分析结果。最后一章对全文作出总结。

## 2 基于任务精确预测的实时功耗温度管理

### 2.1 实时系统的工作负载模型

本文讨论的实时系统的工作负载模型具有简单地结构。系统会周期性地分配一段时间 $D$ , 某一任务的必须在该截止时间以前完成。该任务在最坏情况下所需要的执行时间为 $W$ 。本文中假设任务的截止时间等于系统周期性分配的时间片, 并且等价地只考虑一个周期内任务的执行情况。根据任务的性质, [29]与[30]等文献研究了如何估计 $(D, W)$ 数据对的值。本文中我们认为工作负荷是发送至实时系统的网络流量的归一化形式。

### 2.2 实时系统的热分析模型

为了研究处理器内核(Die)的热传导特性, 文献[4,30,31]等都广泛采用了等效RC电路方法进行热分析, 并采用式2-1进行内核工作温度的分析

$$\frac{dT}{dt} = \frac{P}{C_{th}} - \frac{T - T_{amb}}{R_{th}C_{th}} = \alpha P - \beta(T - T_{amb}) \quad (2-1)$$

式2-1中 $T$ 和 $T_{amb}$ 分别代表芯片的温度与环境温度,  $P$ 代表芯片在时刻 $t$ 的功耗,  $R_{th}$ 与 $C_{th}$ 分别为等效热阻与等效热容。

### 2.3 实时系统的功耗分析模型

多数处理器拥有两种主要模式, 即工作状态和休眠状态: 只有在工作状态下处理器被充足供电, 并执行计算任务; 否则, 处理器将进入休眠状态以减少功耗, 同时降低自身温度。工作状态下的功耗为:

$$P_{active} = CV_{dd}^2 f + N_{gate}I_{leakage}V_{dd} \quad (2-2)$$

式2-2中的第一项代表动态功耗, 第二项代表静态功耗。当给定供电电压 $V_{dd}$ 后, 工作频率 $f$ 为

$$f = \frac{(V_{dd} - V_t)^\mu}{(V_{dd}T_{max})^\eta} \times 4.2824 \times 10^{14} \approx C_1 V_{dd} \quad (2-3)$$

表 2.1  $A, B, \alpha, \beta, \gamma, \delta, \mu, \eta$  的取值

参数	$A$	$B$	$\alpha$	$\beta$	$\gamma$	$\delta$	$\mu$	$\eta$
取值	1.143E-12	1.013E-14	466.403	-1224.741	6.282	6.909	1.19	1.20

由于与工作电压成正比，我们可以使用式2-4计算动态功耗

$$P_{active} = C_2 V_{dd}^3 \quad (2-4)$$

通过HSPICE软件进行的曲线拟合，与温度、电压相关的漏电流可写为

$$I_{leakage} = I(V_0, T_0)(AT^2 \exp(\frac{\alpha V_{dd} + \beta}{T}) + B \exp(\gamma V_{dd} + \delta)) \quad (2-5)$$

式2-5中  $A, B, \alpha, \beta, \gamma, \delta, \mu, \eta$  是经验参数，由芯片的生产工艺参数所决定。本文的模拟实验默认选择采用65nm的工艺参数，具体参数数值见表2.1。

当工作温度  $T$  在300K到380K的正常范围变化时， $\exp(\frac{1}{T})$  的波动变化很小。当给定了  $V_{dd}$  后，文献[23]通过引入两个参考温度  $TH$  和  $TL$  进一步将漏电流简化为温度的二次函数。于是，与漏电流相关的静态功耗可以用式2-6计算

$$P_{leakage} = N_{gate}(\hat{A}T^2 + \hat{B})V_{dd} \quad (2-6)$$

2-6中， $\hat{A} = \frac{I_{leakage}(TH, V_{dd}) - I_{leakage}(TL, V_{dd})}{TH^2 - TL^2}$ ， $\hat{B} = I_{leakage}(TH, V_{dd}) - \hat{A}TL^2$ 。此外，处理器在两种工作状态之间的切换是通过改变工作电压来实现的，状态切换将带来额外的开销，包括能耗开销  $p_r$  与延时开销  $c_r$  [24]。整体而言，工作状态切换跨度越大（切换电压差越大），其能耗和时间的开销也就越大。

## 2.4 目前流行的动态温度与功耗调度策略

### 2.4.1 考虑漏电流的温度敏感技术（简称TALK技术）

Gang Qu[22,32]提出的考虑漏电流的温度敏感技术根据工作负载和截止时间的不同，来控制不同时间段处理器的工作/休息状态：当负载量大并且温度较低时、处理器处于激活工作状态；当负载量小并且温度较高时，处理器切换到睡眠状态以减小能耗，以降低温度。具体而言，该算法会实时监测两个重要的调度指标。其一是当前工作负载量和剩余空闲时间的比例  $\eta$ ，其二是将芯片调至睡眠状态下的温度下降速率与将芯片调至工作状态下的温度上升速率的比例  $\theta$ 。如果前者大于后者处理器说明任务较重，休息状态降温效率不高，于是处理器执行任务，而如果前者小于后者说明任务较轻，休息状态降温效

率高，便将处理器就处于休息状态以实现降温。其中 $\eta$ 比较直观且容易计算，而 $\theta$ 则较为复杂。首先，要计算处理器当前的工作温度 $T_{current}$ ，而后要分别计算该芯片在工作和睡眠状态下的稳定温度 $K_1, K_2$ （即经过了无穷长时间后的温度），最后用式2-7计算得到速率比值。

$$\theta = \frac{T_{current} - K_2}{K_1 - T_{current}} \quad (2-7)$$

#### 2.4.2 基于固定周期的技术（简称Pattern-Based， PB技术）

基于固定周期的算法将任务的截止时间或者运行周期D等分为n个时间片段，每段长 $\Delta = D/n$ ，采用PB算法的处理器将工作于特定规则的模式中[23,32]：执行 $\Delta = D/n$ 时间后便切入休眠模式，以减少功耗并降低温度。文献[23]与[31]证明：如果重复这种运行模式足够多次，处理器将达到温度的平衡值，并进入稳定状态，即每个周期的初始温度和结束温度将趋向于稳定值，以便于分析。

#### 2.4.3 M阶震荡调节技术（简称M-Oscillating， MO技术）

上面介绍的TALK算法和PB算法都要求处理器的工作速度要大于或者等于负载率 $W/D$ 。具有DVS或者DVFS功能的实时性系统往往只允许芯片可以工作在几个电压档上，并可以根据负载率调节实际处于的电压值。这就会导致芯片一直工作在高于需求的速度上。这不仅增加了动态功耗和静态功耗，因为他们分别是电压的三次方和一次方函数，还会加速温度的上升，提高平衡温度的值，最终导致漏电流以二次方的速度增长。一种显而易见的解决方法是允许处理器在两个不同速度下工作。文献[30]证明，如果采用两个最接近的速度完成分配给处理器的任务，那么相对于采用其他的工作速度组合，处于该速度组下处理器的温度是最优的。如果进一步地将这种两步策略应用在m个时间片中[32]，不仅温度可以进一步优化，还可以将D时间内的总功耗表达为m的函数，而且必然存在能耗最小化的m值[25]。由于要考虑电压切换所付出的时间开销和能耗开销，[25]给出了m所具有确定的上限值Ceil。

#### 2.4.4 小结与评价

作为温敏调度算法，TALK参照剩余任务量与当前温度、来合理地调度任务。然而，简单的开关模式无法利用DVS技术，只能工作在固定速度。而且状态切换所导致的时间、能耗开销也是不可避免的。根据切换时间和能耗开销[24]，从全速工作转变为零电压将产生最大的能耗和时间开销。

无论采用TALk还是PB算法，都要求处理器工作在大于 $W/D$ 的速度上。大多数具有DVS或DVFS功能的实时系统通常只允许芯片的电压为若干离散值，根据负载率来调整电压工作档。这往往会导致芯片实际上工作高于任务所需的速度，不仅增加了近似与电压三次方成正比的动态功耗和与电压近似成正比的静态功耗，而且加速了温度的攀升，抬高了平衡态时的温度，进一步导致漏电流近似平方速度的增长。

G.Quan等[25]提出的MO算法存在两个主要缺陷。首先，假设功率为温度的线性函数，使得峰值温度较PB有很大降低。其次是在实际应用中不能忽略低工作负载率情况：当 $W/D$ 小于处理器支持的最低工作速度时，MO只能退化为PB，以防止不必要的功耗增加。

## 2.5 基于电压预测的温敏漏电流技术（简称VP-TALK技术）

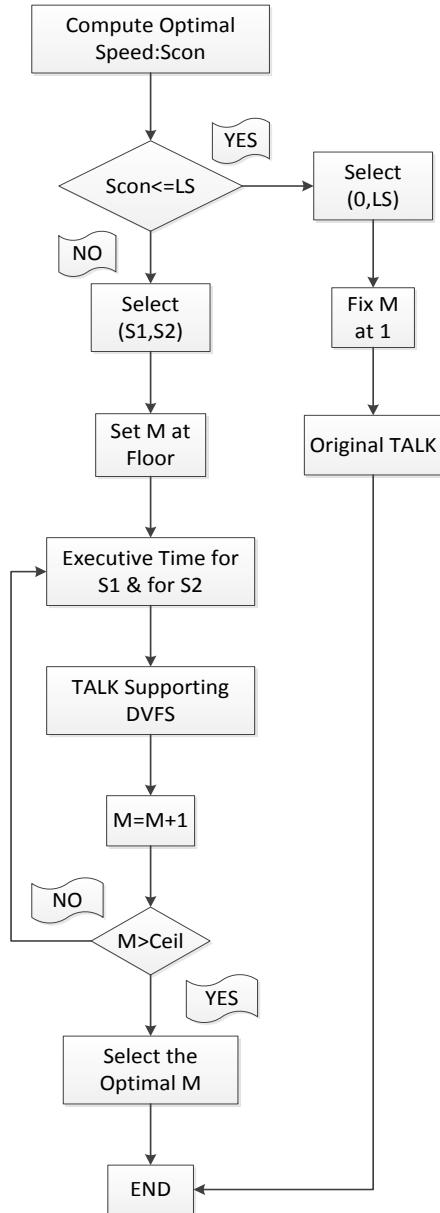


图 2.1 基于电压预测的温敏漏电流技术的流程图

根据我们在DPTM领域的研究经验，可以获得如下关于DPTM改进的几点经验准则：

- 1) 必须考虑温度对静态功耗的影响。本文将功耗定为温度的二次函数。在温度限制下，DPTM系统最好具有温敏调控功能。

- 2) 如果能够准确预测负载量，确定任务所需的工作频率或者工作电压，就可以提前调度，更好的满足实时性。
- 3) 芯片电压选取。由于工作电压决定了运行速度，应该采用MO的电压选取方法，即使得芯片运行速度刚好满足工作负载的需求，以最大程度地利用DVFS技术来降低能耗。
- 4) 调度过程中，必须考虑电压切换（状态切换）所带来的额外能耗、时间开销。

基于以上观察，我们提出一种改良后的TALK算法，即具有电压选择的TALK算法，本文称为VP-TALK，其算法流程图如图2.1所示。

与MO相似，VP-TALK首先需要假设电压可以连续调节、以获得理论上的最优工作速度 $S_{con}$ 。在 $S_{con} \leq LS$ （芯片最低工作速度）时，VP-TALK等同于原始的TALK算法。在 $S_{con} > LS$ 时，该算法选用两档邻接的速度 $S_1$ 和 $S_2$ ，使得 $S_1 \leq S_{con} < S_2$ 。不同于MO的等分 $M$ 段时间段，VP-TALK采用更灵活的、电压可调的温敏TALK来对每一小段的工作状态进行调度，即对每一个小的时间片，采用TALK中定义的两个调度指标（ $\eta$ 与2-7）来决定是将芯片设置为休眠或是工作状态。由于 $m$ 的数量由切换工作状态的代价和任务的实时性所限制，其上下限分别记为Ceil和Floor[25]。VP-TALK的应用前提是假设我们已经通过某种预测的方法预测出了任务负载量，从而，在任务到达前调度就已经开始，所以认为是实时性的。

## 2.6 动态温度与功耗管理系统

### 2.6.1 启发性示例

在之前的分析中，我们已经指出，较轻工作负载时的MO必然要退化至PB的方法。这是因为当工作所需电压低于可选的最小电压值时，MO中阶梯型工作电压策略无法通过逼近最优工作电压的方式节省动态功耗。为了探究工作量与最优调度算法之间的关系，我们设定任务长度为10秒，并考虑比文献[25]更强的温度对漏电流影响，我们在工作负载率全区间（5%-95%）范围内，对TALK、PB与MO这3种已有调度源算法的调度效果进行了考察。图2.2给出了能耗的数据，图2.3给出了温度数据。

我们可以得出如下结论：

- 1) 当工作负载率（W/D）低于50%（近似值）时，三种源算法的峰值温度低于310K（37°C），峰值温度对系统的性能与可靠性没有影响；在系统能耗方面，PB算法（也即MO算法）的调度效果要胜过TALK算法，因此，PB算法具有最佳的调度效果。

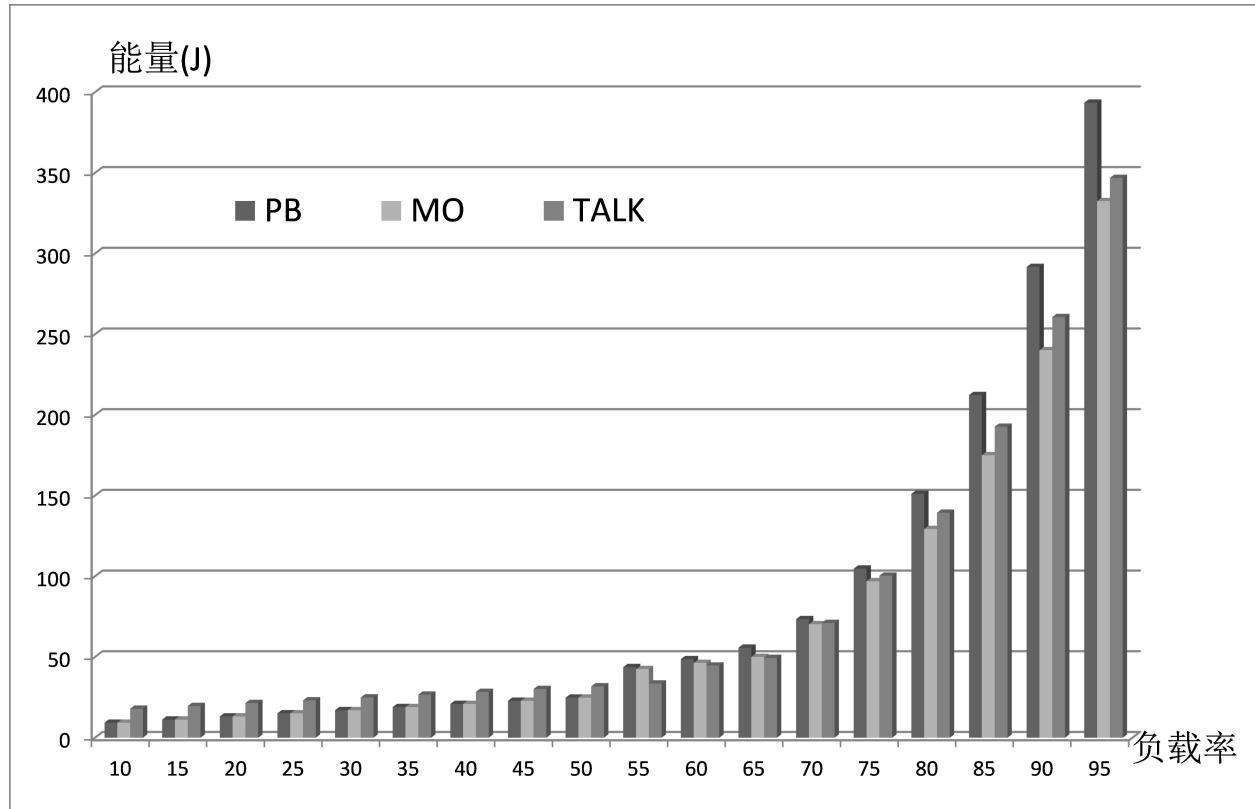


图 2.2 各种调度算法的能耗效果随负载率变化的比较

- 2) 当工作负载率 ( $W/D$ ) 处于50%—70%区段（近似值）时，三种源算法的峰值温度低于320K (47°C)，峰值温度对系统的性能与可靠性也没有影响；在系统能耗方面，TALK算法的调度效果要胜过PB和MO算法，因此，TALK算法具有最佳的调度效果。
- 3) 当工作负载率 ( $W/D$ ) 大于70%（近似值）时，三种源算法的峰值温度高于320K (47°C)，最高可超过380K (107°C)，峰值温度对系统的性能与可靠性具有明显影响，其中MO算法具有最低的峰值温度；在系统能耗方面，MO算法的调度效果要明显胜过其它两种源算法，MO算法具有最佳的调度效果。

由此可见，最优的DPTM调度算法与工作负载率有直接关系。我们将以此关系作为理论基础，用于DPTM调度原型系统的构建，即根据对实时系统工作负荷的精确预测结果，来选择效果最佳的调度算法，并对DPTM调度效果进行评价。

### 2.6.2 基于时间序列分析的工作负载预测

本文采用文献[32]中所提出的基于时间序列分析的工作负载预测模型来满足VP-TALK算法预知工作负载率的需求。具体的负载预测模型构建的公式推导和演化请参见文

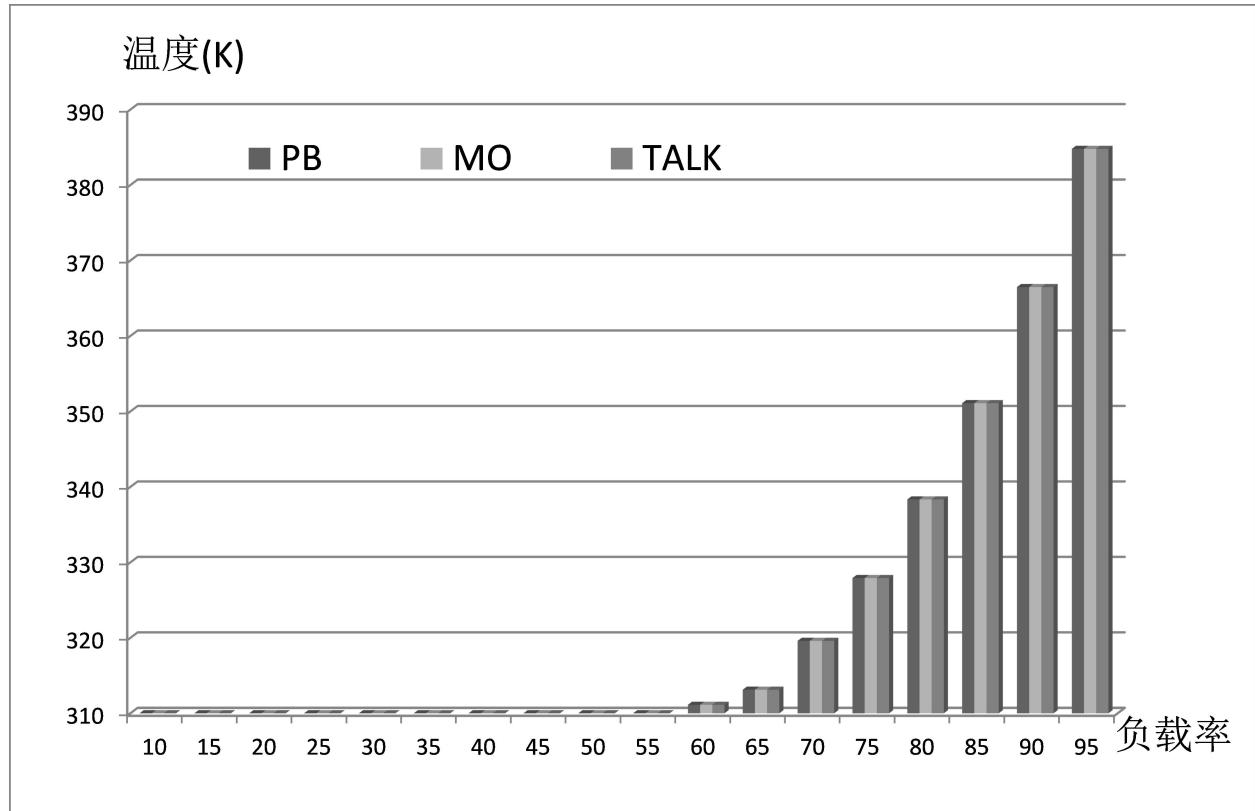


图 2.3 各种调度算法的峰值温度效果随负载率变化的比较

章[32]的相关章节。这里，主要简单介绍整个预测流程。实时系统的任务从其本质属性来讲，可以分为三种不同的成分。本文认为，工作负载 $X(t)$ 中包含

- 1) 趋势成分 $T(t)$ : 对应任务量随着时间而平稳增长、衰减的性质；
- 2) 周期成分 $P(t)$ , 对应任务量随时间变化表现出的规律性；
- 3) 随机成分 $R(t)$ , 对应任务量之中不确定的随机性。

从而可以将任务看作三者之和： $X(t) = T(t) + P(t) + R(T)$ 。基于对三种基本成分的分析，本文提出了基于灰色模型、傅里叶级数分解模型以及径向基函数（RBF）人工神经网络理论的组合预测数学模型，根据若干数量的真实的任务负载历史数据，预测片上系统下一个时间步的近似任务量。具体而言，

- 1) 趋势成分反映工作负载随着时间演变而增加或衰减的趋势。灰度模型GM(1,1)模型可以通过累加原始数据数列，弱化随机扰动因素影响，发现该序列的指数增长规律；
- 2) 对于一个周期函数，最成熟的分析方法就是用傅里叶级数展开或者傅里叶变换；
- 3) 众所周知，任何服务系统的工作任务都会有一定的随机性，称之为任务负载中的随机成分。径向基函数神经网络是通过非线性基函数的线性组合，对非线性函数关系具有良好的逼近能力，适于描述网络流量的非线性、时变性的复杂因素，并可克

服BP神经网络训练时间长及计算复杂度高的不足

### 2.6.3 基于在线评价的动态温度与功耗管理系统

根据调度算法性能与工作负载大小相关的观察，以及基于时间序列分析的工作负载预测，我们提出了在线评价、选择DPTM调度算法的调度策略，并以此构建了本文的实时功耗温度管理原型系统。整个系统由工作负载预测、调度策略选择和调度策略评价三大模块组成。在该系统工作中，其三大模块主要完成如下功能。

- 1) 工作负载预测模块：我们根据负载变化周期的长短，提出了一种组合式的负载预测方法，采用多种不同拟合方法来分别对任务的不同物理意义成分进行精确预测，以获得对复杂任务的精确预测；
- 2) 调度策略选择模块：我们综合考虑实时完成任务、温度上限、能耗最小化、漏电流与温度相关以及芯片模式切换代价等多种因素，选用不同的任务调度策略；
- 3) 调度策略评价模块：对每种策略的系统能耗与峰值温度进行评价，并将其作为实时功耗温度管理系统的反馈量，供调度策略选择模块参考。

调度策略选择的学习主要通过后期性能评价的评价值完成。假设存在 $N$ 类算法，编号为 $1, 2, \dots, N$ 。它们在某一时刻 $t$ 的得分或者权重为， $w_t = (w_{1t}, w_{2t}, \dots, w_{Nt})$ 对其中任一个权重分量 $w_{kt}$ 可以采用式2-8进行计算

$$w_{kt} = 1 - \sum_{j=t_0}^{t-1} \frac{E_{k,j}\lambda_j}{\sum_{i=1}^N E_{i,j}} \quad (2-8)$$

式2-8中 $E_{i,j}$ 代表芯片使用第 $i$ 类DPTM算法在时刻 $j$ 的负载情况下消耗的能量， $\lambda_j$ 为一可调整的参数，代表时刻 $j$ 的能耗情况对决策的影响程度， $t_0$ 是可以变动的初始值，它的取值意味着选取从何时开始的能耗情况作为以后决策的参考。在预测出工作负荷值后，则开始使用2-9进行决策：

$$DPTM_t = \arg(\max_{1 \leq k \leq N} (w_{k,t})) \quad (2-9)$$

式2-9中 $DPTM_t$ 为 $t$ 时刻选择出的动态功耗温度调度策略。考虑到温度上的限制，我们需要考察所选的调度源算法是否会超过温度上限。如果能耗的节约是在很高的峰值温度的代价下换取的，我们将放弃该源算法，而选择次优的但是有较低峰值温度的调度源算法。与三种已有源算法和本文提出的VP-TALK算法相比，本文实时功耗温度管理系统的扩展改进点在于：具有高精度的任务预测模块，为根据负载量而进行的策略选择提供前提基础；通过基于调度效果评价的机器学习，自适应地根据负载量的轻重选择调度策略。

#### 2.6.4 基于单一调度策略的动态温度与功耗管理系统

值得一提的是，如果我们选定某一种调度策略，省略机器学习模块，就构成了基于单一调度策略的实时功耗温度管理原型系统。在该实时功耗温度管理原型系统中，输入量为任务负载的历史值，通过这些历史值，利用第六节所述的任务负载预测模型，可以得到对于下一时刻任务量的预测值。进而，可以确定完成预测任务值所需要的芯片电压或者频率，并利用上文所述的某一种算法进行调度。不论是基于及其学习的实时功耗温度管理系统，还是基于单一调度策略的实时功耗温度管理系统，都要求提前预知任务负载的大小。因此，本文采用所提出的组合模型预测方法。该方法将复杂任务按频谱长短分类为随机/周期/趋势等三种成分，然后采用灰色模型/傅里叶模型/径向基函数（RBF）神经网络模型对这三种成分进行组合分析，可以得到平均相对误差低于3%、归一化方差小于0.5的任务负载预测效果[32]。

### 3 动态温度与功耗管理系统的模拟验证

#### 3.1 模拟实验环境

为了验证本文所提出的预测任务负载模型、VP-TALI调度算法以及DPTM原型系统，我们进行了三组模拟实验。实验运行平台为配有Intel Core 2 Q9550 CPU、4GB RAM的Windows7 64位操作系统，预测模型以及各DPTM算法在MATLAB[33]软件中进行仿真模拟。

#### 3.2 VP-TALK算法的动态温度与功耗管理效果验证

本文通过与现有的TALK、PB、MO算法进行对比，来验证本文提出的VP-TALK算法的优越性。实验中选取 $p_r$ 、 $c_r$ 取值分别为 $0.001s/V$ 和 $0.01J/V^2$ ，温度最高上限设为390K，任务周期固定为10s。支持DVFS芯片的电压值从0.6V到1.4V可调，步长0.1V。根据电压可以得到归一化的速度序列为0.574, 0.6611, 0.7324, 0.7926, 0.8446, 0.8901, 0.930, 0.9670, 1。任务负载率为5%至95%，变化步长为5%。对于4种对比算法，我们分别将能耗情况（单位为焦耳J）和峰值温度情况（单位为绝对温度K）绘制于图3.1和图3.2。

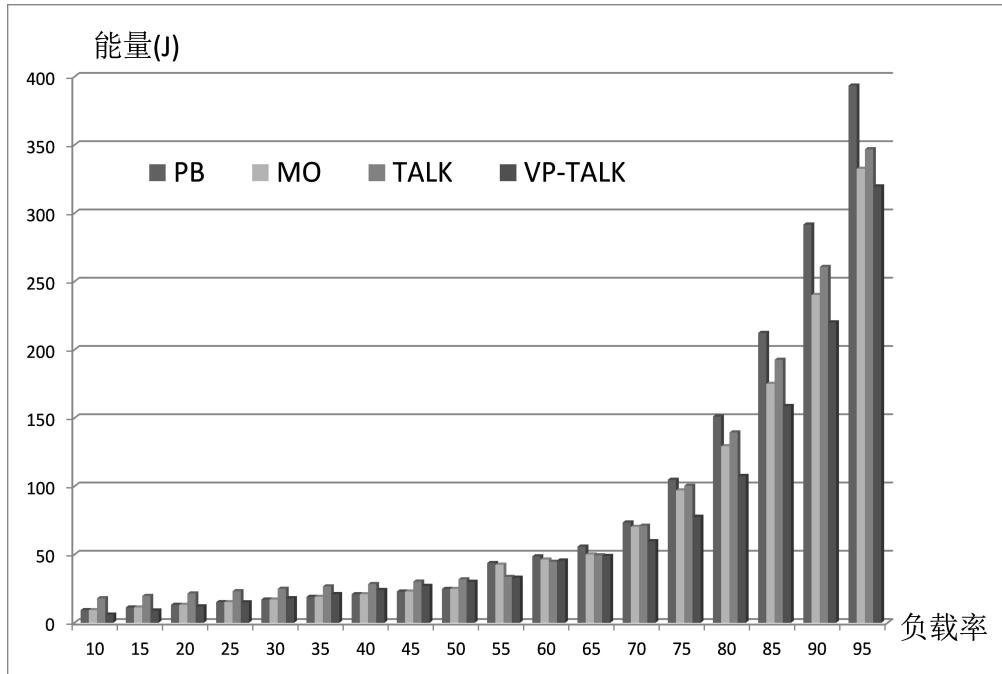


图 3.1 各种算法的能耗调度效果比较

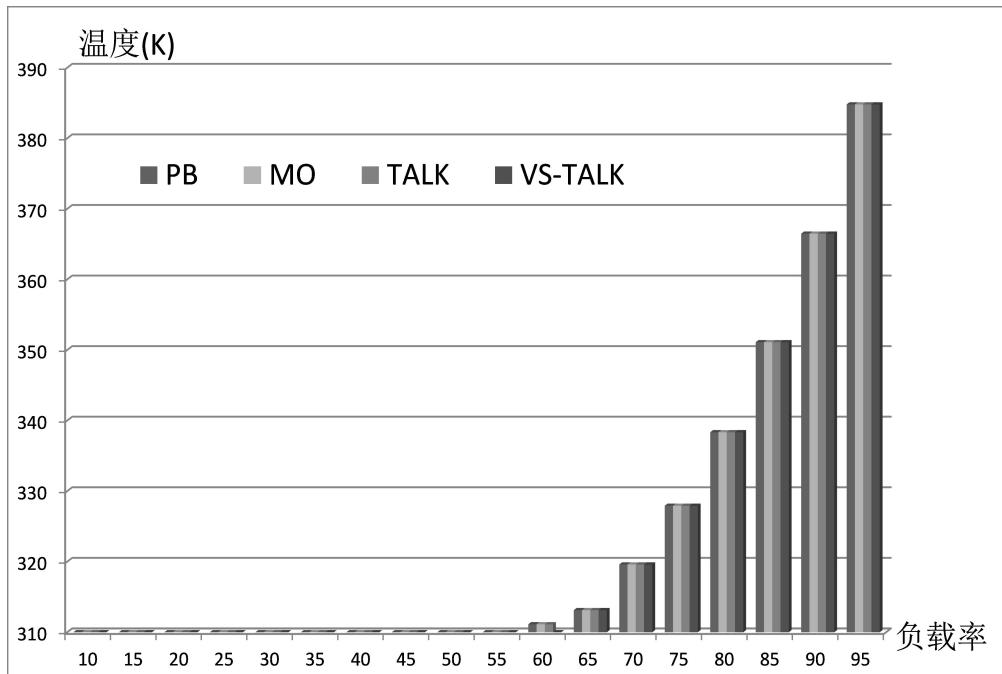


图 3.2 各种算法的峰值温度调度效果比较

从图3.1可以看出，对于能耗：当负载率小于55%时，VP-TALK优于TALK、PB与MO等价。这是因为，在负载率不高于芯片最低运行速度（0.574）

表 3.1 VP-TALK算法调度效果与已有算法的比较

算法	PB	MO	TALK	Avg
平均(%)	20.54	11.04	11.42	14.33
最大(%)	28.83	22.34	21.27	24.68

时，VP-TALK相当于使用DVS技术的TALK，PB相当于使用单一速度的MO。此时，TALK与PB的调度效果相差无几。在负载率为25%到50%时，PB略优于VP-TALK算法。原因是VP-TALK在电压和时间切换上付出了更大的代价。然而，随着负载率的增加，当负载率大于55%时，VP-TALK的调度效果全面优于其他三种算法，比PB、MO以及TALK的平均能耗分别节省大约20.54%、11.04%、11.42%。表3.1列出了当负载率大于55%时，能耗节省数据的统计信息，具体说明了VP-TALK的在能耗节约方面的优势。从图3.2中可以看出，当工作负载率小于50%时，各种算法所达到的峰值温度都不超过310K，四种调度算法在峰值温度方面的差距基本小于1K，并在最大负载率时近似共同终结于384K。

### 3.3 基于在线评价的动态温度与功耗管理系统的效果评测

为了评价本文基于机器学习的实时功耗温度管理原型系统功耗与温度的调度效果，我们做了两组实验。首先，基于[32]中所得到的每个时刻工作负载预测值，我们分别采用四种对比源算法，即TALK[22]、PB[23]、MO[25]，VP-TALK，算出每个时刻每种算法的能耗值与峰值温度；然后将四种对比源算法每个时刻的能耗值与峰值温度进行平均，得到四种对比源算法每个时刻的平均调度效果；最后使用图3.3与图3.4分别比较了本文原型系统与四种对比源算法平均调度效果的能耗（单位J）和峰值温度（单位°C）。

### 3 动态温度与功耗管理系统的模拟验证

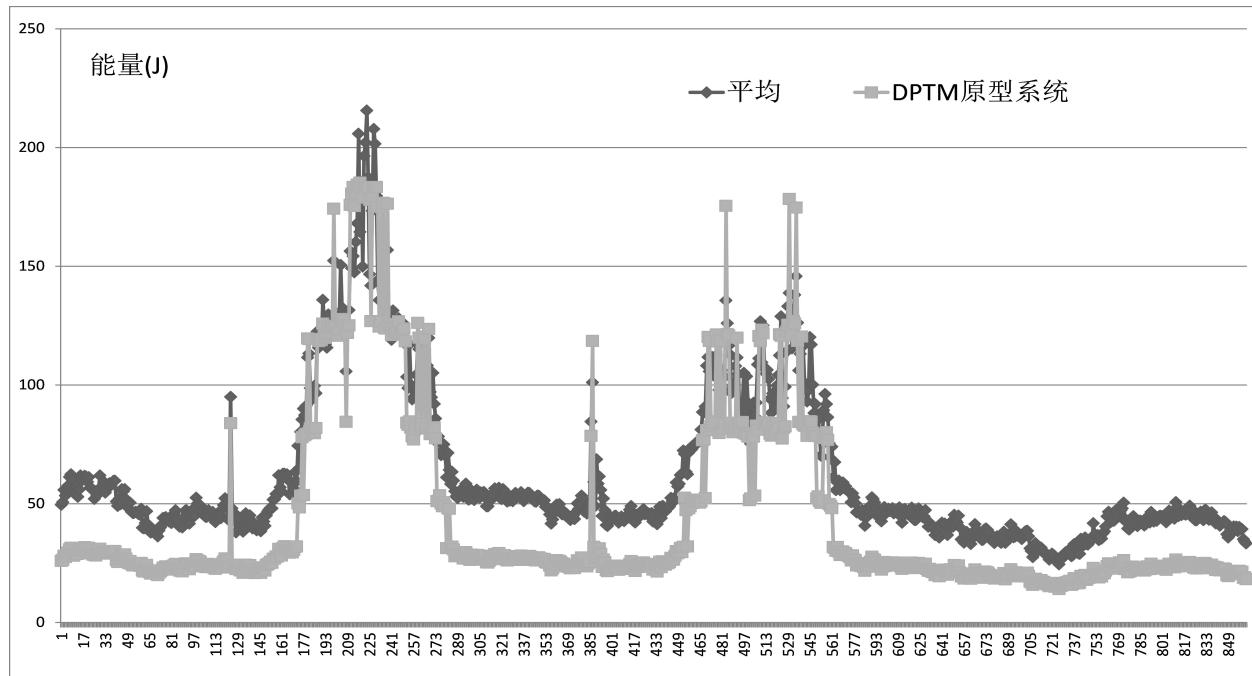


图 3.3 基于在线评价的动态温度与功耗管理系统VS各种源算法均值（能耗调度效果）

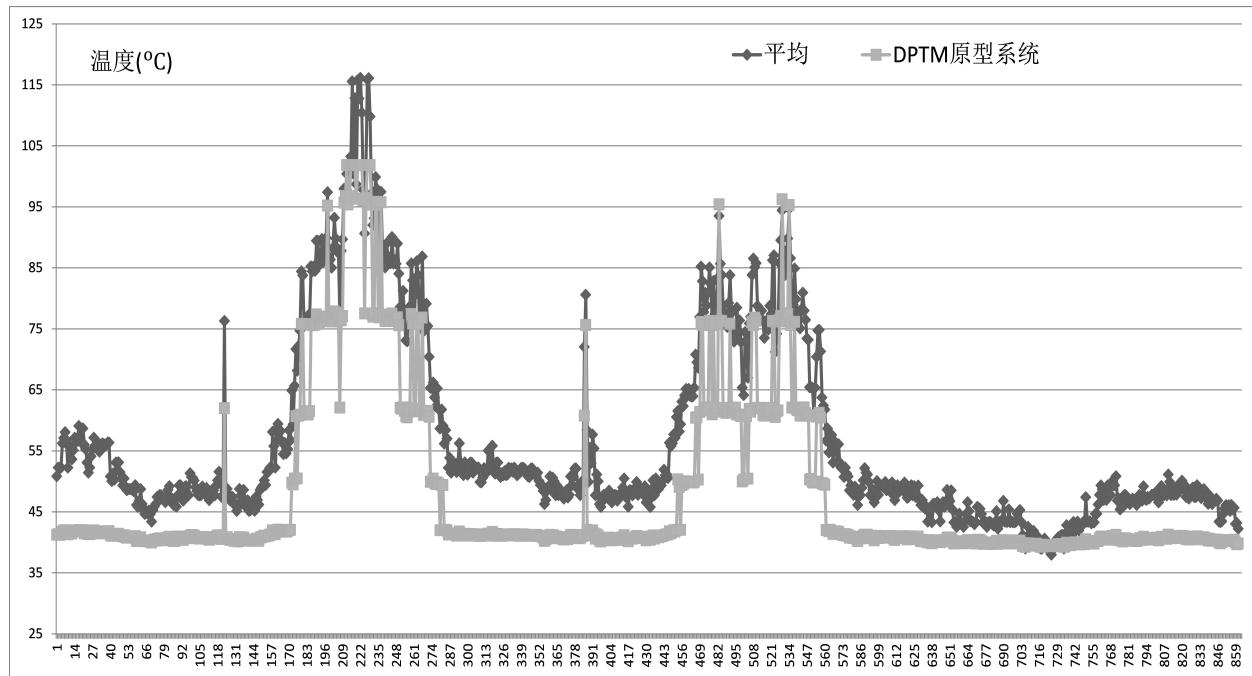


图 3.4 基于在线评价的动态温度与功耗管理系统VS源算法均值（峰值温度调度效果）

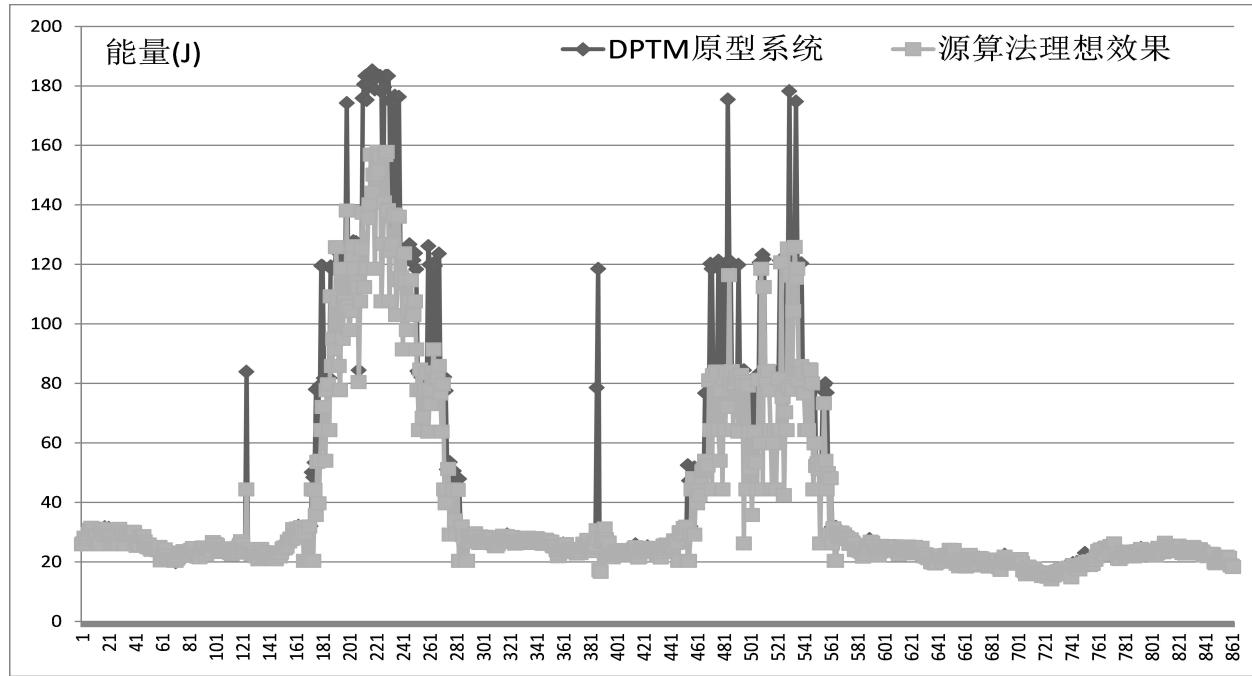


图 3.5 基于在线评价的动态温度与功耗管理系统VS理想调度效果（能耗调度效果）

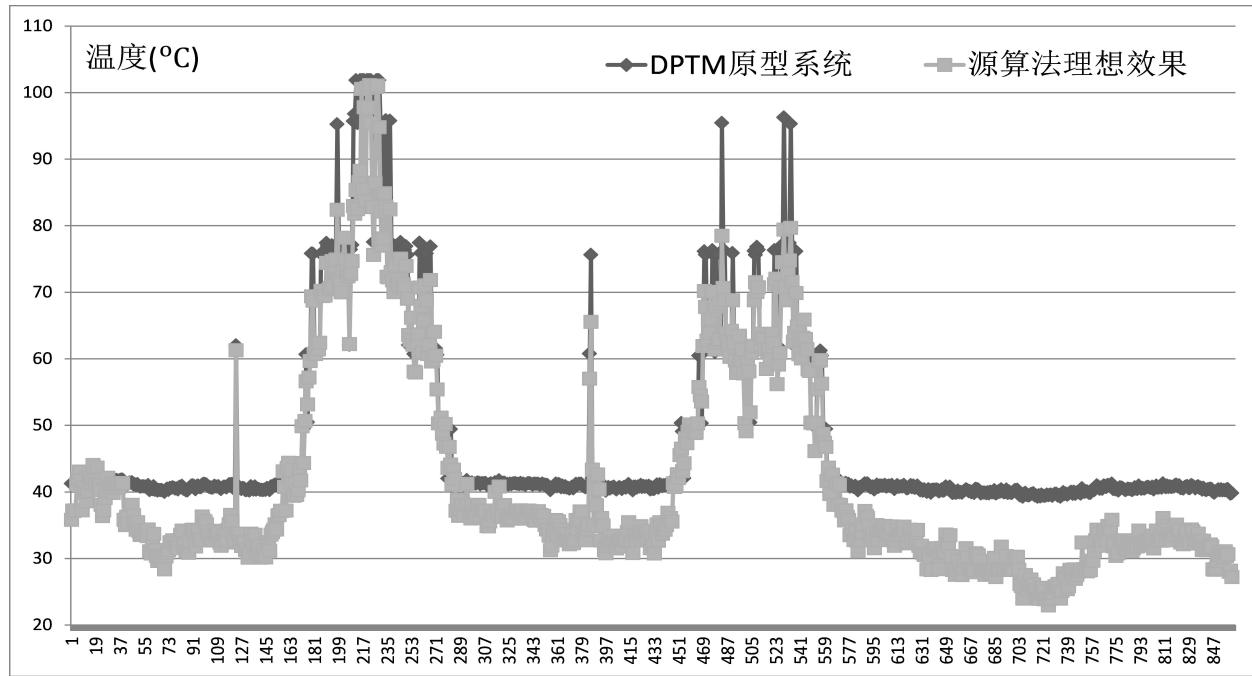


图 3.6 基于在线评价的动态温度与功耗管理系统VS理想调度效果（峰值温度调度效果）

其次，同样基于[32]中所得到的每个时刻工作负载真实值，采用四种源算法，分别算出每个时刻的能耗值与峰值温度，取其中的最优值作为实时功耗温度管理调度的理

想值；然后使用这批理想值来客观评价集成了TALK、PB、MO以及VP-TALK算法的实时功耗温度管理原型系统的调度效果；最后使用图3.5与图3.6分别比较了本文原型系统与四种对比源算法理论最优调度效果的能耗（单位J）和峰值温度（单位°C）。从图3.3与图3.4可以直观地看出：与四种源算法平均调度效果相比，本文原型系统择优式的组合DPTM算法可以明显拉低实时系统的运行能耗曲线和峰值温度曲线，这表明对于四种DPTM源算法，本文基于机器学习的实时功耗温度管理原型系统获得了“取其长、去其短”的优化效果。同时从表3的比较数据可以得出“本文原型系统可以获得近似最优的调度效果”的结论，其实验依据如下：

- 1) 与四种DPTM源算法的平均效果相比（图3.3），本文原型系统采用的择优式组合DPTM算法可获得更优的能耗优化效果，所有时间采样点能耗的累加值 $E_{TOTAL}$ （总能耗）可获得18.39%的改进，所有时间采样点中的最大值 $E_{MAX}$ （最大能耗）可获得18.77%的改进。整体改进效果非常明显。
- 2) 与四种DPTM源算法相比（图3.4），本文原型系统可以获得更优的峰值温度优化效果，所有时间采样点中的最大峰值温度 $T_{P_{MAX}}$ 是最优，所有时间采样点峰值温度的平均值 $T_{P_{AVG}}$ 是比算法均值稍弱。与四种DPTM源算法调度效果平均值相比，本文原型系统可以获得 $T_{P_{MAX}} 1.81^{\circ}\text{C}$ 的改进，但 $T_{P_{AVG}}$ 有 $-1.31^{\circ}\text{C}$ 的退化。从拉低最大峰值温度的角度来讲，改进效果较为明显。
- 3) 通过图3.5与图3.6对本文原型系统的调度效果与理想值进行的直观比较，看出本文原型系统可以获得比四种源算法均值更接近于理想值的优化效果。与表3中理想值的 $E_{TOTAL}/E_{MAX}/T_{P_{AVG}}/T_{P_{MAX}}$ 参数相比，四种DPTM源算法调度效果平均值会产生28.64%/30.09%/7.71°C/9.73°C的差距，而本文原型系统只产生了12.55%/13.93%/9.02°C/7.91°C差距。

### 3.4 本章总结

第2章深入分析与评估了已有的主流调度算法，提出了一系列调度准则和经验。基于这些对芯片工作休眠状态调度的经验准则，我们提出一种在能耗节省方面更具优势的DPTM调度算法VP-TALK，以此算法为基础，综合本文所提出的预测任务负载模型，构建了一个基于负载预测的实时功耗温度管理系统。本章中的仿真实验表明，

- 1) 本文的组合模型在负载预测方面胜过众多的相关模型及算法，平均误差仅为2.89%；
- 2) 本文所提出的VP-TALK算法在较高的工作负载率和共同的峰值温度约束下，分别

- 比Pattern-Based、M-Oscillating、TALk分别节能20.54%、11.04%、11.42%；
- 3) 本文所提出的综合四种源算法、基于机器学习的实时功耗温度管理原型系统较为接近理想值，与其 $E_{TOTAL}/E_{MAX}/T_{PAVG}/T_{PMAZ}$ 参数相比，只产生了12.55%/13.93%/9.02°C/7.91°C差距。

## 4 考虑温度对漏电流功耗影响的MPSoC结构级热分析算法

### 4.1 多核芯片热分析的研究对象建模

#### 4.1.1 芯片的散热系统

图4.1给出了完整的芯片散热系统结构，在散热片下依次是导热层2(TIM2)、扩热层、导热层1(TIM1)，导热层1下方就是芯片的内核(die)，注意：内核+导热层1+扩热层+散热片+风扇构成了芯片的主散热通道。内核下方是芯片的封装基座，封装基座下方是芯片插座，芯片插座下方则是印刷电路版(PCB)，注意：内核+封装基座+芯片插座+PCB构成了辅散热通道。主散热通道的散热能力强于辅散热通道几个数量级，所以在全芯片3D热分析中，均只讨论主散热通道，而忽略辅散热通道，本文下面将主散热通道简称散热通道。

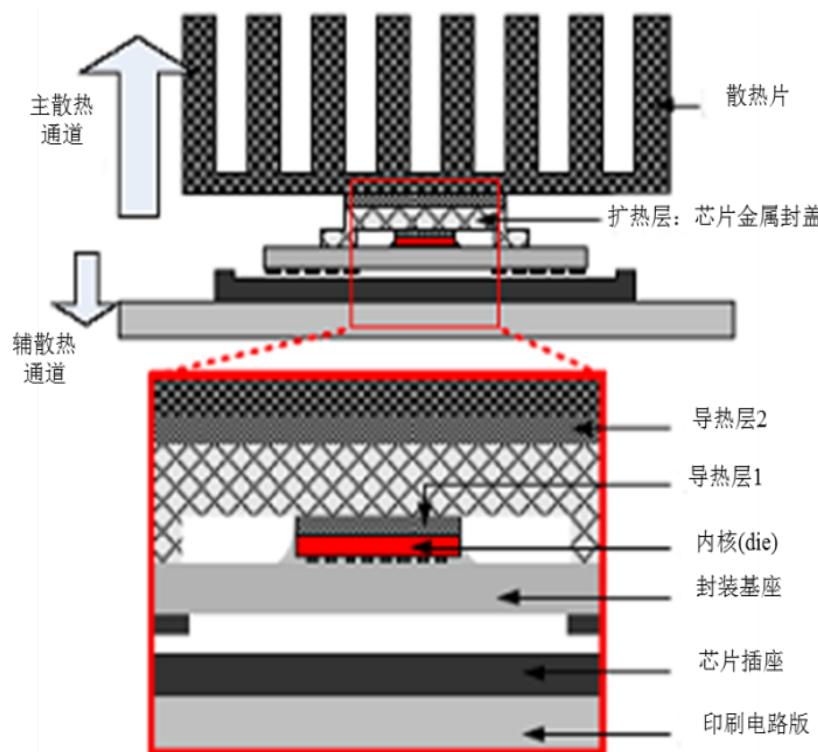


图 4.1 IC 散热系统的结构示意

从图4.1还可以看出，IC散热系统包括：内核的硅衬底、由硅胶或硅脂等软性材

表 4.1 导热材料的热设计参数

材料	$k(W/(cm \times K))$	$\rho(kg/cm^3)$	$c(J/(kg \times K))$
铜	4.00	0.00892	386.00
硅	1.00	0.00232	171.00
硅脂	0.04	0.00240	NA

料构成的导热层、由芯片金属壳构成的扩热层、及散热片等散热器件。图4.2为Intel公司FCLGA10的散热系统结构，可以看出，在封装好的芯片内，芯片金属封盖起到扩热的作用；为了保护内核，封盖与内核之间采用的是一层很薄的软性材料（多数为硅脂）。

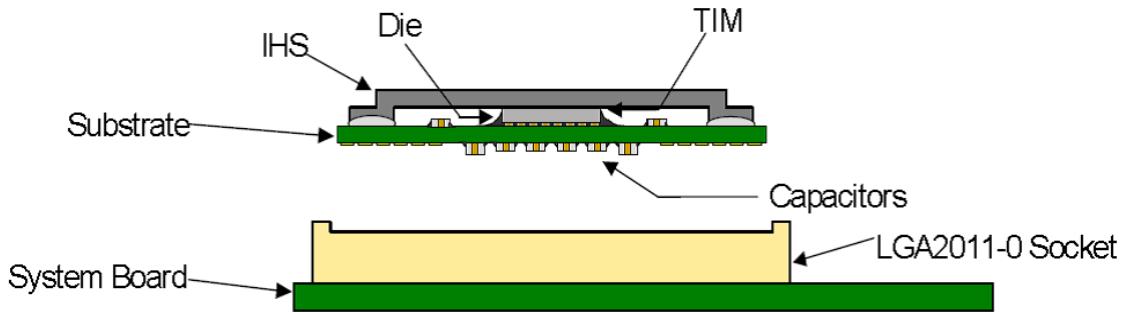


图 4.2 Intel公司FCLGA10封装的散热系统结构

一般而言，芯片散热系统所用的导热材料有：铜Cu（封盖与散热片）、硅Si（硅衬底）、硅胶或硅脂（导热层），其中铜和硅的导热能力要远高于硅胶与硅脂。因此，在散热系统中，为了降低系统的热阻，必须尽量减小导热层的厚度，但为了保证不损坏内核，金属封盖与内核之间的导热层厚度一般取0.5-0.8mm，从图4.2可以看出：Intel公司CPU芯片的导热层厚度要小于0.5mm，这主要取决于该公司的巨大技术实力。

为了对芯片温度进行稳态或者瞬态分析，就必须对各传热器件进行热阻（热导）和热容的参数提取。为此，我们将铜、硅、导热层(TIM)材料的热导率 $k$ （单位为 $W/(cm \times K)$ ）、比重 $\rho$ （单位为 $kg/cm^3$ ）、比热容 $c$ （单位为 $J/(kg \times K)$ ）列于表4.1中。对于作为导热层材料的硅脂，由于工艺不同，其参数变化较大，所以取上界。

#### 4.1.2 多核架构及其电热分布

目前多核CPU普遍采用同质架构。即每个核心（core）拥有相同的逻辑功能模块（computing unit）、容量相同的专享缓存(exclusive cache)，占有相同的内核面积，同时共享最后一级缓存(last level cache,LLC)缓存、I/O等功能模块[20,34]。每个核心具有相同数

量的工作模式，不同的工作模式意味着消耗不同程度的能量。一般来说，每个核心除具有一个全速高能模式外，还具有多种耗能程度不同的节能模式[14]。在每个核心内，逻辑功能模块具有最大的功耗密度，该功能模块对应的指令L1缓存和数据L1缓存功耗密度次之，L2缓存具有的功耗密度相较而言最小。由于注入的热量大，每个核的热点（温度最高点）出现在逻辑功能模块，所以在物理设计中，理论上说，要将逻辑功能模块布放在散热条件好的芯片边沿处，而散热条件最差的芯片中央布放功耗密度最小的LLC，从而降低芯片的热点温度。图4.3为Alpha 21264芯片的物理布局[34]，

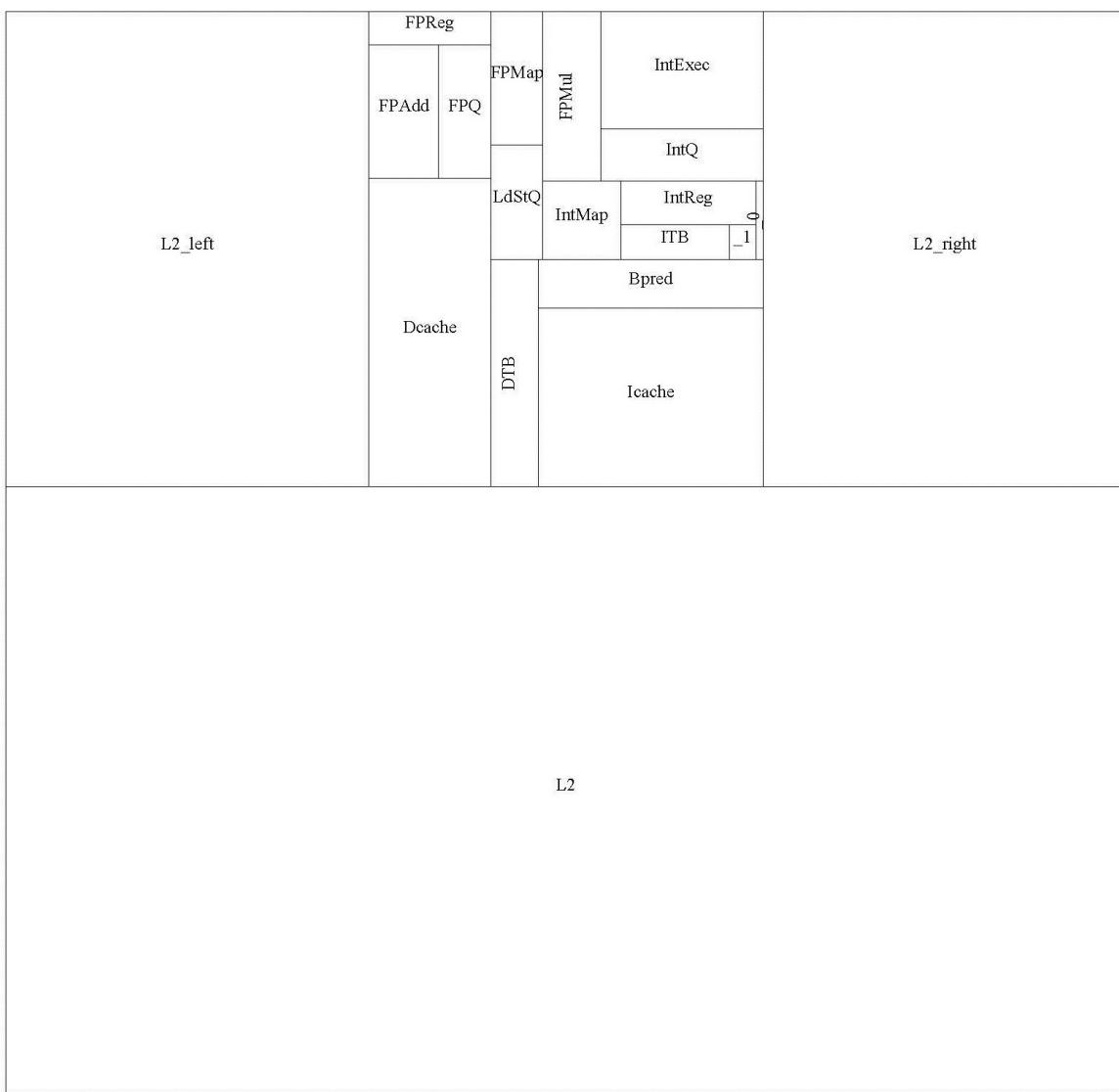


图 4.3 Alpha21264芯片的物理布局

#### 4.1.3 芯片热分析及HotSpot模块级模型

在MPSoC结构级热分析中，一般采用已有的电热等效模型，分析稳态温度分布，以降低计算复杂度[7,8]。对于稳态热分析而言，将芯片的功耗分布作为注入的热流向量 $\mathbf{P}$ ，对芯片的物理结构进行离散化建模后，可以获得节点之间的热导矩阵 $\mathbf{G}$ ，目前多采用如下的稳态热分析方程计算节点温度分布向量 $\mathbf{T}$ ：

$$\mathbf{G} \times \mathbf{T} = \mathbf{P} \quad (4-1)$$

对于多核实时功耗温度管理(dynamic power and temperature management, DPTM)研究[13, 35]，目前广泛采用Skadron等人提出的HotSpot热分析模型（软件）[34]构建热导矩阵 $\mathbf{G}$ ，并采用式4-1进行计算。HotSpot采用基于等效热导的电路模型，将体系结构级的功能模块作为分析热点的对象。除了模块级别的热分析模型，HotSpot仍然提供了更为复杂的网格级(grid mode)热分析模型与方法。本文所指的HotSpot模型及其计算结果均指模块模式(block mode)的热分析模型与结果。一种直观的对应芯片以及热封装的物理结构的具体建模例子如图4.4所示。电路模型在垂直热传导方向上有3层：内核(die)层、扩热(heat spreader)层、与散热片(heatsink)层，另外加入第4层热对流(heat convector)层，即与环境温度的接口。内核层根据芯片的几何布局被分为块；扩热层分为5块：与内核层完好对应的 $R_{sp}$ 以及4块呈梯形状的环绕块；散热片层按照与扩热层相似的划分方法，分为 $R_{hs}$ 以及4个环绕块；最后，从热封装到外界环境的热对流层由 $R_{convection}$ 表示[14,18]。层与层之间模型刻画由内核直至封装及外界环境的热流；层内水平模型刻画相邻模块间的热扩散。内核层产生的功耗等效为每个模块中心的电流源[18,20]。建模完成后，通过RC瞬态与稳态电路分析，可得到芯片的温度分布。用HotSpot计算分析Alpha 21264芯片的温度分布如图4.5所示。

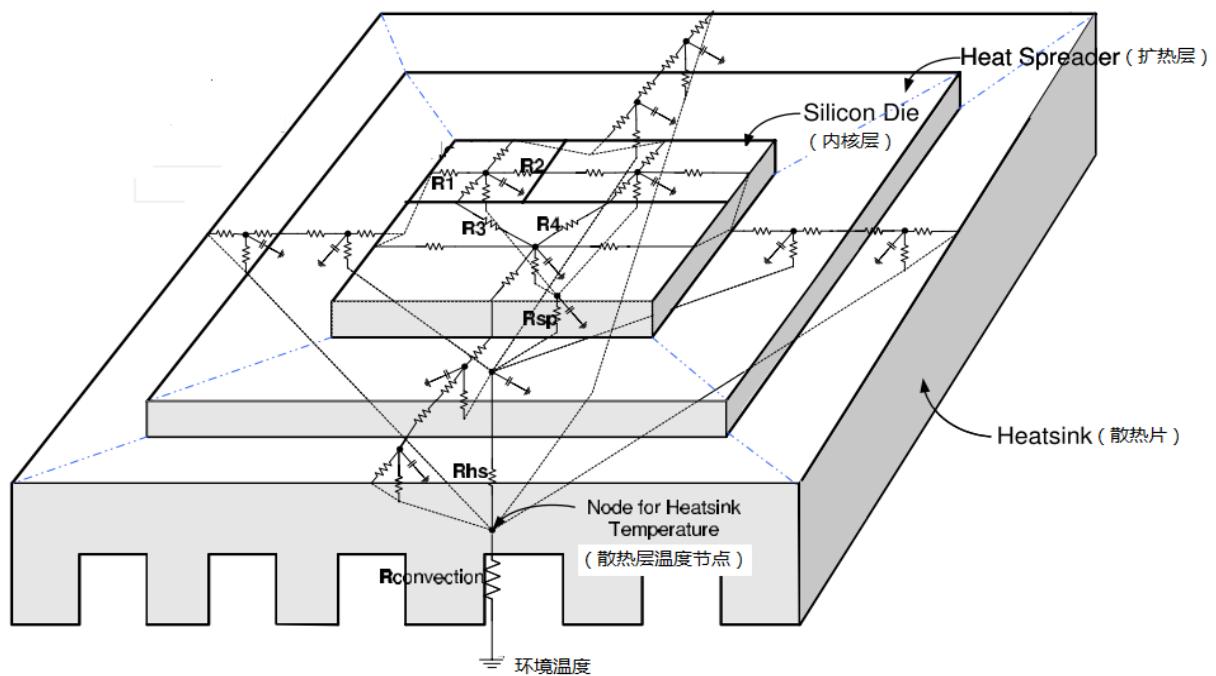


图 4.4 HotSpot 对芯片的电路等效热分析模型

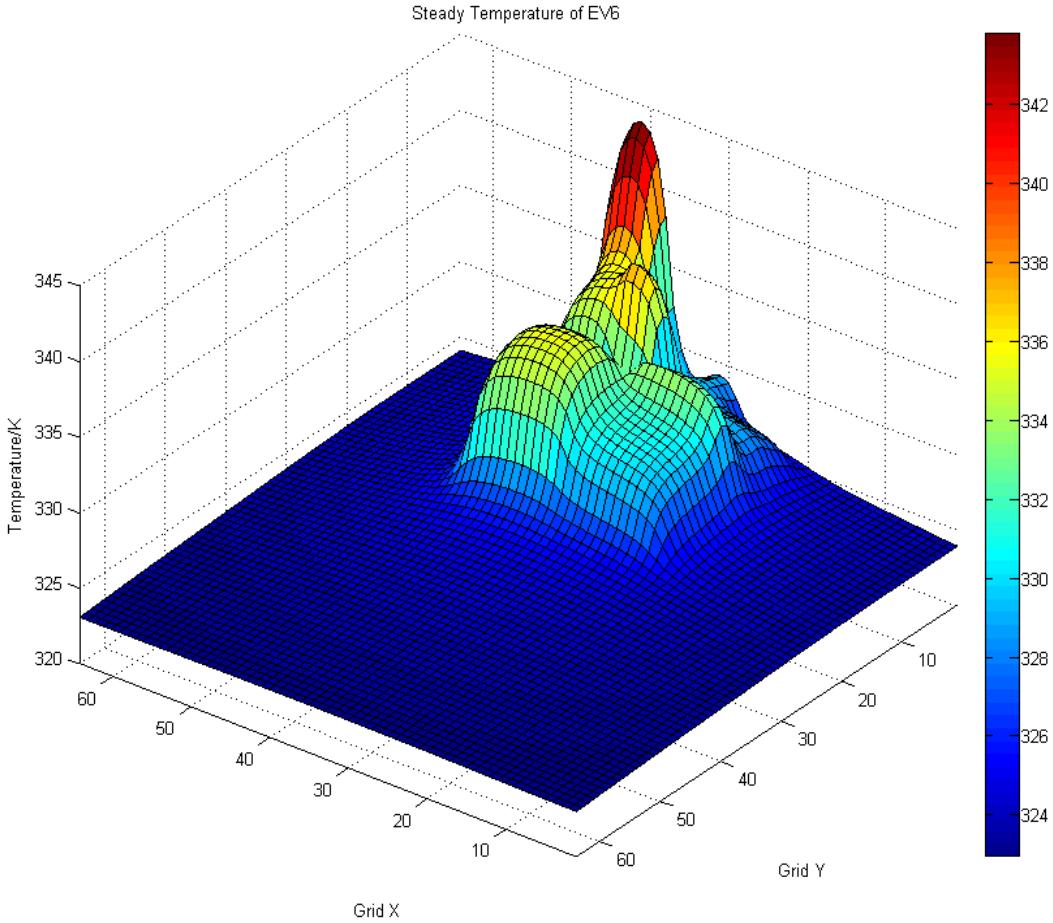


图 4.5 Alpha21264芯片的温度分布

#### 4.1.4 电热耦合效应：温度对漏电流功耗的影响

正如2.3节中，式2-4、式2-6所定性与定量地描述那样，芯片功耗由动态功耗 $P_{dynamic}$ 与静态功耗 $P_{leakage}$ 两部分组成，随着芯片制作工艺的提高， $P_{leakage}$ 已成为芯片功耗的主要贡献者。而工作温度 $T$ 的升高可以明显增大 $P_{leakage}$ 。这就是所谓的芯片内部电热耦合效应。在考虑电热耦合效应的热分析过程中，需要对 $T - P$ 采用迭代方法计算，如4.6所示，对于一个16核CPU的测例（具体的实验参数设置见第5章），采用迭代算法来逼近最终的精确解。

与不考虑电热耦合效应的初始解相比，芯片最高温度与静态功耗都有了明显的增加，这表明在芯片的温度分析中、必须考虑温度对静态功耗的影响，否则，将会产生较大的分析误差。同时，与不考虑电热耦合效应的温度分析算法相比，由于考虑电热耦合

效应的温度分析算法需要采用7次迭代计算才能获得精确解，所以其算法复杂度是对比算法的7倍；因此降低考虑电热耦合效应的温度分析算法复杂度就具有非常重要的研究意义。

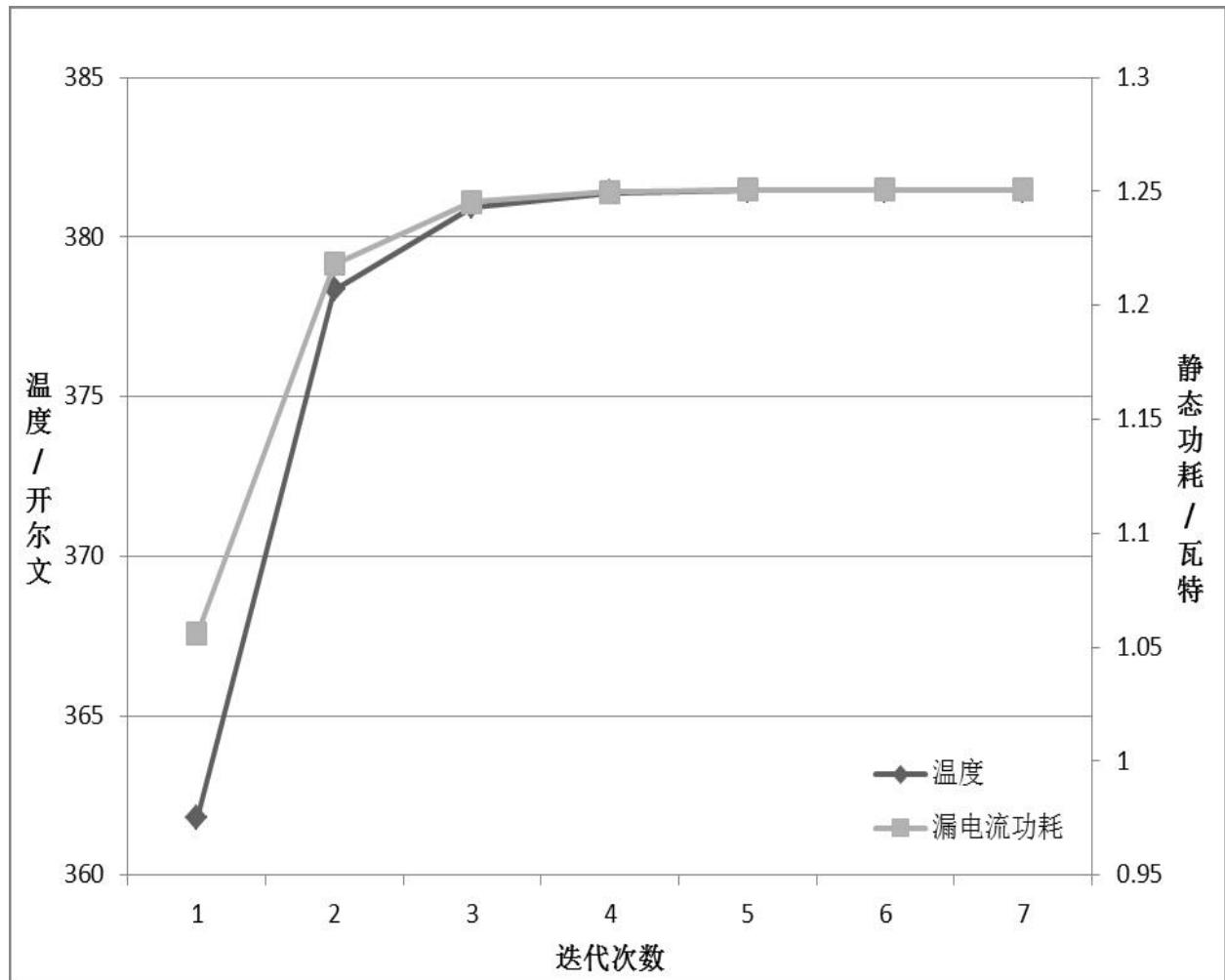


图 4.6 考虑电热耦合效应的多核芯片最高温度与静态功耗的迭代求解

## 4.2 3种MPSoC结构级热分析算法

### 4.2.1 模块级热分析算法（简称BlockTAM算法）

为了对功能模块进行热分析建模，按照式4.4对HotSpot的多核芯片热分析模型，采用等效电路的方法对其进行进一步简化，为此本文假设[18]：每个功能模块内的功耗与温度分布是均匀的，以该模块中心点的温度作为该模块的温度，并将功耗密度乘以面积作为该模块的功耗，加于模块中心点。在本文工作中，将模块 $u$ 的功耗 $P_{u,u}$ 作为注入热源。对

于多核芯片，如果仅在模块 $u$ 加上幅值为 $P_{u,u}$ 的阶跃激励，其他模块均不加激励，则可以使用HotSpot模拟器获得所有模块的温度 $T_{v,u}$ 响应曲线。先根据 $T_{u,u}$ 的最终收敛值 $\bar{T}_{u,u}$ 可以计算出模块 $u$ 的等效热阻 $R_{u,u}$ ，计算公式为4-2

$$R_{u,u} = \frac{\bar{T}_{u,u}}{P_{u,u}} \quad (4-2)$$

再根据等效热阻 $R_{u,u}$ 以及模块 $v$ 功耗 $P_{v,v}$ 的阶跃激励作为单一注入热源所得到的 $\bar{T}_{u,v}$ ，可以采用式4-3计算出反映 $P_{v,v}$ 对模块 $u$ 温度作用关系的等效热阻 $R_{u,v}$ ，

$$R_{u,v} = \frac{\bar{T}_{u,v}}{P_{v,v}} \quad (4-3)$$

最后根据所获得的参数 $R_{u,u}$ 与 $R_{u,v}$ ，可以计算 $P_{v,v}$ 对模块 $u$ 温度计算产生影响的等效热源 $P_{u,v}$ ：

$$P_{u,v} = \frac{\bar{T}_{u,v}}{\bar{T}_{u,u}} P_{u,u} = \frac{R_{u,v} \times P_{v,v}}{R_{u,u} \times P_{u,u}} P_{u,u} = \frac{R_{u,v}}{R_{u,u}} P_{v,v} \quad (4-4)$$

因此，按照图4.7中所给出的单模块温度分析模型，可以列出如下热分析表达式：

$$T_{u,u} = R_{u,u} \sum_{v=1}^N P_{u,v} = R_{u,u} \widehat{P}_u \quad (4-5)$$

其中 $N$ 为多核芯片中的功能模块数目， $\widehat{P}_u = \sum_{v=1}^N P_{u,v}$ 为模块 $u$ 的等效热源，由于一个模块只有一个等效热源，可见式4-5提供的单模块热分析模型兼容了经典的单核（单模块）热分析模型[20]。从式4-5可知：BlockTAM方法的算法时间复杂度与空间复杂度均为 $O(N^2)$ ，即算法复杂度为 $O(N^2)$ 。

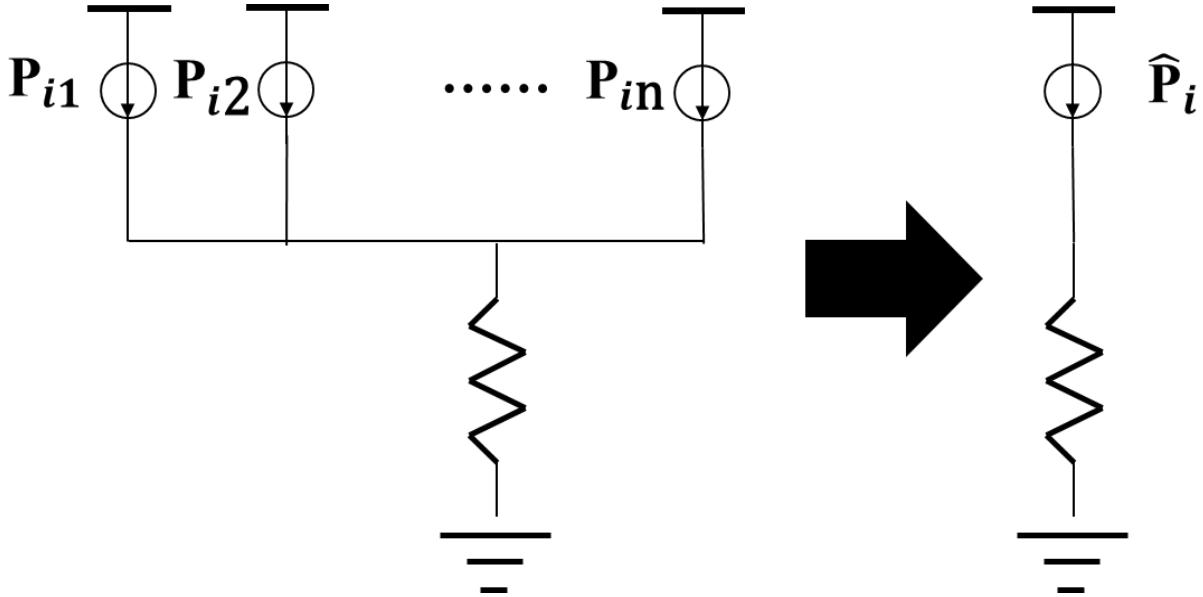


图 4.7 多核实时功耗温度管理的单模块与单核热分析简化等效电路模型

#### 4.2.2 核级热分析方法（简称CoreTAM算法）

为了对处理器核进行热分析建模，可以基于图4.4中HotSpot的多核芯片热分析模型，采用等效电路的方法对其进行进一步简化，为此，进行如下的假设[18]：

- 1) 在处理器核多个功能模块中，逻辑模块（包括ALU等器件）产生最高的工作温度，将逻辑模块的温度作为处理器核的温度；为了对其他核产生影响，将处理器核中所有功能模块的功耗加到逻辑模块中心。
- 2) 为了计算温度对漏电流功耗的影响，将逻辑模块的静-动态功耗比例、作为处理器核中所有功能模块的静-动态功耗比例，也就是：随着工作温度的变化，变的是静-动态功耗比例，不变的是核内所有模块功耗的比例。

基于如上假设，将核p内的所有模块均加上符合核内功耗比例要求的阶跃激励，其他核均不加激励，则可以使用HotSpot模拟器获得所有核的温度 $T_{t,s}$ 响应曲线。先根据 $T_{s,s}$ 的最终收敛值 $\bar{T}_{s,s}$ 可以计算出核s的等效热阻 $R_{s,s}$ ，计算公式如下：

$$R_{s,s} = \frac{\bar{T}_{s,s}}{P_{s,s}} \quad (4-6)$$

式4-6中 $P_{s,s}$ 是核s内所有模块的功耗之和。再根据等效热阻 $R_{s,s}$ ，以及核t功耗 $P_{t,t}$ 的阶跃激励作为单一注入热源所得到的 $\bar{T}_{s,t}$ ，采用如下公式计算出反映 $P_{t,t}$ 对核s温度作用关系的等

效热阻 $R_{s,t}$ ,

$$R_{s,t} = \frac{\bar{T}_{s,t}}{P_{t,t}} \quad (4-7)$$

之后根据以上所获得的参数，可以计算 $P_{t,t}$ 对核 $s$ 温度计算产生影响的等效热源 $P_{s,t}$ :

$$P_{s,t} = \frac{\bar{T}_{s,t}}{\bar{T}_{s,s}} P_{s,s} = \frac{R_{s,t} \times P_{t,t}}{R_{s,s} \times P_{s,s}} P_{s,s} = \frac{R_{s,t}}{R_{s,s}} P_{t,t} \quad (4-8)$$

最后，按照图4.7中所给出的热分析模型，可以列出如下热分析表达式:

$$T_{s,s} = R_{s,s} \sum_{t=1}^n P_{s,t} \quad (4-9)$$

式4-9中 $n$ 为多核芯片中的核数。从式4-9可知：CoreTAM方法的算法复杂度为 $O(n^2)$ 。从计算机系统结构可知，核数 $n$ 远小于模块数 $N$ ，因此，多核芯片的核级温度分析复杂度要远小于模块级。

#### 4.2.3 考虑核内模块相互影响的改良核级热分析方法（简称BlockInsideCoreTAM算法）

在核级热分析方法研究中，假设了核内各个模块对核内逻辑模块中心温度具有相同的相关热阻，并且每个核内仅需计算一个点的温度；核内各模块均使用该温度来刷新模块的漏电流功耗，会使整个核的漏电流功耗增大。为了使漏电流计算更为精确，必须计算出各自模块的温度，为此，下面给出一个考虑核内模块相互影响的改良核级热分析方法：BlockInsideCoreTAM。在BlockInsideCoreTAM方法中，先进行如下的假设[18]:

- 1) 核 $t$ 的功耗 $P_{t,t}$ 对核 $s$ 内所有模块均产生相同的等效热量影响为 $P_{s,t} = \frac{R_{s,t}}{R_{s,s}} P_{t,t}$ ，其中 $R_{s,t}$ 为两个核内所有模块之间相关热阻的平均值，即

$$R_{s,t} = \frac{1}{m^2} \sum_{u \in BS_s} \sum_{v \in BS_t} R_{u,v} \quad (4-10)$$

式4-10中 $BS_x$ 是核 $x$ 内所有模块的集合。

- 2) 在核 $s$ 内，模块 $v$ 对模块 $u$ 的等效热量影响为 $P_{u,v} = \frac{R_{u,v}}{R_{u,u}} P_{v,v}$ ，式中 $u, v \in BS_s$ 。

按照以上假设，可以列出如下BlockInsideCoreTAM表达式:

$$\frac{T_{u,u}}{R_{u,u}} = \sum_{t=1, t \neq s}^n P_{s,t} + \sum_{v=1}^m P_{u,v} \quad (4-11)$$

从式4-11可知：BlockInsideCoreTAM方法的算法复杂度为 $O(n^2 + (m^2 - 1)n)$ 。BlockInsideCoreTAM的计算复杂度介于CoreTAM和BlockTAM之间，当 $m \ll n$ 时，BlockInsideCoreTAM的算法复杂度近似等于CoreTAM。

## 5 MPSoC结构级热分析算法的验证实验

### 5.1 验证实验平台

本文选择热分析软件HotSpot（版本5.02）作为验证基准，构建3个测例进行温度-功耗数据对比分析，从而检验所提出的模块级、核级、改良核级这3种热分析方法的精度与效率。HotSpot软件以及三种热分析方法的运行平台为配有Intel酷睿i7，8G内存，运行Ubuntu 11.04 LTS操作系统的PC机。

### 5.2 验证实验所用到的测例

测例1采用文献[17]策略，将4个Alpha21264核心组合为一个4核处理器。每个Alpha21264的处理核心核配由两块独享缓存L2\_Right、L2\_Left，4个核共享最后一级缓存L3。芯片的布局为：将处理核心放置于芯片的四个角落，每个核的独享缓存处于核的周围，而将共享缓存区置于芯片中心（这样做不仅考虑了处理单元的片上散热问题，也有利于通过功率较低的共享缓存，隔断功率较大的多个核相互之间的热交换影响）。

测例2采用Intel Sandy Bridge-E架构的8核高端CPU芯片[36]。该芯片核心面积为435平方毫米（长宽尺寸分别为20.9毫米、20.8毫米，基本呈正方形）。本文中，我们按照Sandy Bridge-E i7-3960X的布局原则模拟生成了一种较为简略的芯片布局结构，整体平面图如图5.1所示，主要具有的模块为：双精度数据输入输出模块（DDR IO），最后一级缓存模块（LLC），核心模块，快速互连通道模块（QPI），外设成员互连通道模块（PCI-E），功耗控制单元模块（PCU）等。每个核心内的布局如图5.2所示，主要组成单元为：执行单元模块（Execution Unit），第一级数据缓存模块（L1 Data Cache），指令获取和第一级指令缓存（Instruction Fetch & L1 Cache），内存序列与执行模块（Memory Ordering & Execution），分页单元模块（Paging），乱序执行调度与退化模块（Out-of-Order Scheduling & Retirement），指令解码模块（Instruction Decode & Microcode），分支预测模块（Branch Prediction）。

## 5 MPSoC结构级热分析算法的验证实验

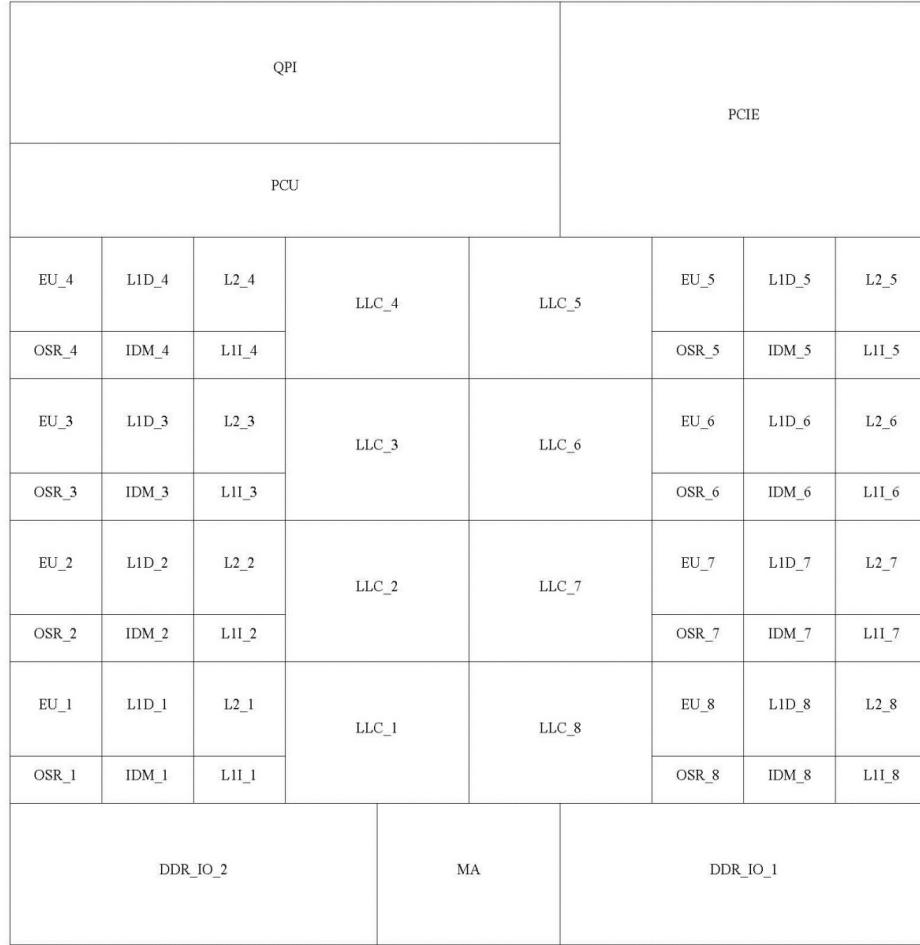


图 5.1 测例二i7-3960X芯片布局

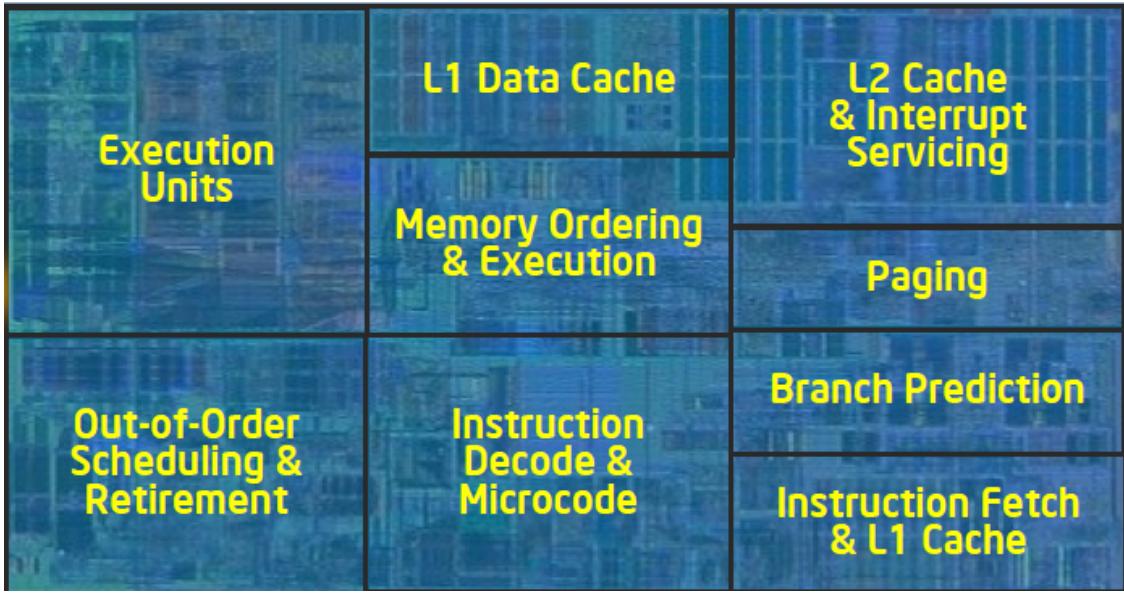


图 5.2 测例二i7-3960X核内的布局

对于测例1和测例2，进行类似的实验方案。该方案过程如下：根据表5.1中所示功率参数分配设置方案，首先重复地随机产生功耗分布；然后，在考虑电热耦合效应的情况下，分别采用BlockTAM、CoreTAM和BlockInsideCoreTAM方法，获得每个功率分布的温度稳定分布；最后将得到的结果与HotSpot的分析结果进行对比。

测例3基于测例1中采用过的Alpha21264核心，构建多个对于16核的虚拟芯片的布图规划。为了模拟温敏布局规划过程，本文对各个布图方案进行热分析，在模拟过程中，测试本文方法的运行耗时或效率。

### 5.3 针对MPSoC结构级热分析算法精度的实验数据与分析

正如前文所提到的，本文按照表5.1中设定的功率参数[18]，随机产生100个功率输入向量；然后采用HotSpot、BlockTAM、BlockInsideCoreTAM、CoreTAM这4种热分析方法，测试各个核与功能模块的温度与静态功耗。其中，一个核的静态功耗值等于所属模块的静态功耗值之和，这个核的局部热点温度是它的Core\_x模块温度。本文以HotSpot的分析结果值作为基准，来计算其余3种方法的相对误差。由于芯片的工作温度均是相对环境温度 $T_{Amb}$ 而言，所以使用式5-1来计算温度误差 $E_T$ ：

$$E_T = \left| \frac{T - T_{HotSpot}}{T_{HotSpot} - T_{Amb}} \right| \quad (5-1)$$

其中 $T_{HotSpot}$ 是HotSpot的测值， $T$ 是本文方法的温度计算值。同时，采用式5-2来计算静态

功耗误差 $E_{Plk}$ :

$$E_{Plk} = \left| \frac{Plk - Plk_{HotSpot}}{Plk_{HotSpot}} \right| \quad (5-2)$$

式5-2中 $Plk_{HotSpot}$ 是HotSpot的静态功耗测值,  $Plk$ 是本文方法的静态功耗计算值。

3种方法的温度误差被列入表5.2[18]。从表中不难看到, BlockTAM的精度最高, 平均误差小于1.6%, 最大误差小于5%; BlockInsideCoreTAM的精确效果次之, 平均误差小于4.2%, 最大误差小于15%; CorTAM的精度最低, 平均误差达到33%, 最大误差达到45%。所以, BlockTAM和BlockInsideCoreTAM这2种方法均可以提供满意的热分析精度。

3种方法的静态功耗误差如表5.3所示[18]。BlockTAM的精度最高, 最大误差小于0.5%; BlockInsideCoreTAM的精度次之, 平均误差小于1.2%, 最大误差小于3.7%; CorTAM的精度最低, 平均误差达到25%, 最大误差达到35%。因此, BlockTAM和BlockInsideCoreTAM 2种方法均可以提供满意的静态功耗分析精度。

为了考察本文方法静态功耗与温度分析的误差来源, 图5.3和图5.4[18]给出了本文3种方法对测例1中的各个模块, 所产生的静态功耗与温度相对误差。对本文3种方法的误差分析如下:

- 1) BlockTAM方法的误差分析: 由于BlockTAM是根据HotSpot软件的分析结果进行的自下而上的建模, 所以具有与HotSpot软件最近分析精度。如图5.3所示, 各模块的静态功耗分析误差小于0.3%; 如图5.4所示, 各模块的稳态温度分析误差小于4%。
- 2) CorTAM方法的误差分析: 在测例1中的一个核内, 具有高功耗(高发热量)的Core\_x模块自然是高温模块, 在其总功耗中, 静态功耗所占的比例较高; 而其他3个模块(L2\_x、L2\_Left\_x、L2\_Right\_x)是具有低功耗的低温模块, 其静态功耗所占的比例也较低。如图5.3所示, 产生局部热点的Core\_x模块温度明显高于其他核内模块。CorTAM模型假设核中所有模块的温度均等于核的局部热点温度, 这与事实有着较大的出入, 导致其所计算出来的低温模块静态功耗被明显拉高, 故而产生了11%-15%的较大误差。另一方面来说, 高温模块的温度就是核内的局部热点温度, CorTAM所计算出来的高温模块静态功耗也就理所应该较为精确(其相对误差小于2%)。由于低温模块的静态功耗计算误差明显偏大, 以核内所有模块功耗总和来计算核的局部热点温度就会产生较大的误差; 如图5.4所示, 其所产生的温度误差处于6%-8%之间, 明显大于其他2种方法。
- 3) BlockInsideCoreTAM方法的误差分析: 该方法正视了核内模块温度之间具有着明显差别, 所以, 仅仅将核外模块作为一个影响整体进行考虑。由于仅将热点之间的相

表 5.1 本文测例各功能模块的面积与功耗参数设定

测例	功能模块	面积 ( $mm \times mm$ )	动态功率(w)
1	Core_x	$3 \times 3$	10-20
	L2_left_x	$5 \times 3$	1-2.5
	L2_x	$5 \times 5$	2-5
	L2_right_x	$3 \times 5$	1-2.5
2	DDR_IO_2	$3 \times 8$	2-6.5
	MA	$3 \times 4$	2-6.5
	DDR_IO_1	$3 \times 8$	2-4.5
	OSR_x	$1 \times 2$	0-1
	IDM_x	$1 \times 2$	0-1
	L1I_x	$1 \times 2$	0-2
	EU_x	$2 \times 2$	1-8
	L1D_x	$2 \times 2$	0.5-3
	L2_x	$2 \times 2$	0.5-4.5
	LLC_x	$4 \times 3$	2-6.5
	PCU	$2 \times 8$	2-7.5
	QPI	$3 \times 12$	2-7.5
	PCIE	$3 \times 8$	2-7.5

表 5.2 多核芯片核内局部热点的温度计算误差对比

	BlockTAM		CoreTAM		BlockInsideCoreTAM	
	Avr. $E_T$	Max. $E_T$	Avr. $E_T$	Max. $E_T$	Avr. $E_T$	Max. $E_T$
测例1	1.361%	1.574%	6.573%	12.851%	2.223%	2.259%
测例2	1.447%	4.864%	33.063%	45.840%	4.185%	14.598%

关热阻作为核间影响的系数，所以该方法对于热点温度和静态功耗，可以提供较高的精度（如图5.3所示，各模块的静态功耗误差小于0.7%）；但其对于核内低温模块的温度计算精度则较差（如图5.4所示，产生热点的高温模块的温度误差小于2.2%，但低温模块的温度误差则达到了5%-11%）。

表 5.3 多核芯片各核静态功耗的计算误差对比

	BlockTAM		CoreTAM		BlockInsideCoreTAM	
	Avr. $E_{Plk}$	Max. $E_{Plk}$	Avr. $E_{Plk}$	Max. $E_{Plk}$	Avr. $E_{Plk}$	Max. $E_{Plk}$
测例1	0.314%	0.327%	6.857%	10.816%	0.495%	0.570%
测例2	0.144%	0.317%	25.123%	35.084%	1.196%	3.687%

表 5.4 1000组热分析各算法计算耗时及加速倍数X对比

分析算法	HotSpot	BlockTAM	CoreTAM	BioCorTAM
$T_{Anls}/s$	61.301	1.216	0.414	0.927
$T_{Totl}/s$	61.301	4.663	4.374	4.374
$X_{Anls}$	BASE	50.416	147.962	66.100
$X_{Totl}$	BASE	13.147	15.876	14.014

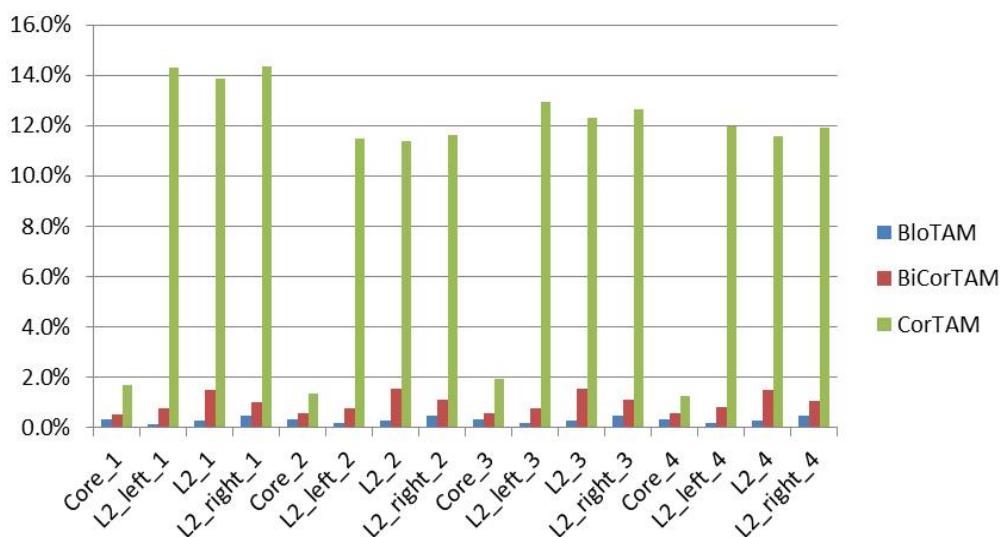


图 5.3 对测例1中各模块的静态功耗计算误差

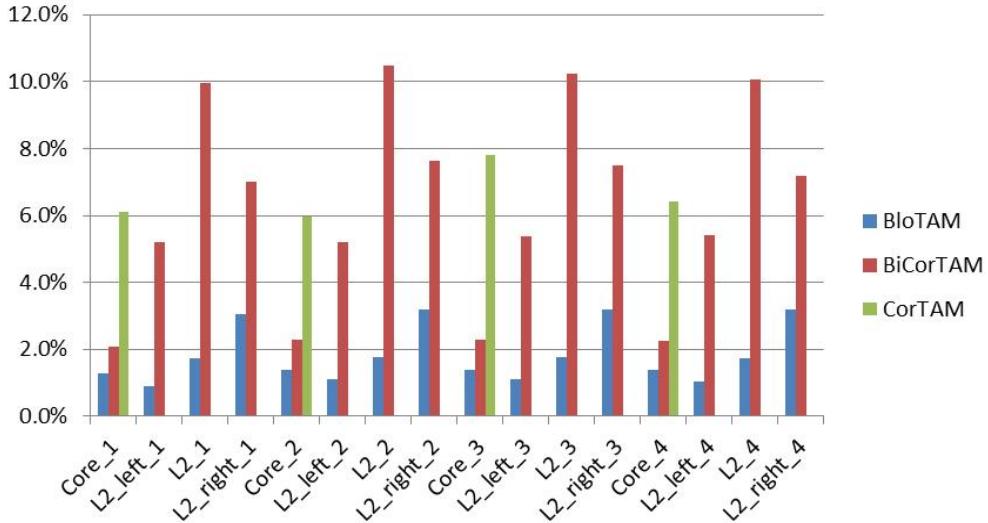


图 5.4 对测例1中各能模块峰值温度的计算误差

#### 5.4 针对MPSoC结构级热分析算法速度的实验数据与分析

本文采用16核的测例3（大测例）对其进行计算速度的验证。3种方法需要采用HotSpot软件对相关热阻的参数提取，由于芯片中有16个核、每核有4个模块，所以本文方法需要进行64次HotSpot模拟，这部分耗时被称为建模时间 $T_{Model}$ 。

实验测试结果表明， $T_{Model} = 3.447$  秒（S）。建模之后，本文方法基于这些参数，采用式4-5、4-9、式4-11 对1000个输入功耗进行温度分析与静态功耗计算，这部分耗时被称为热分析时间 $T_{Anls}$ ；建模时间与热分析时间两部分的总和被称为总耗时 $T_{Totl} = T_{Model} + T_{Anls}$ 。本文方法所提供的加速倍数 $X$ 是HotSpot的耗时与本文方法耗时之比，其中，不计入建模时间而只计入热分析计算耗时的加速比被称为热分析加速倍数 $X_{Anls}$ ，而总耗时加速倍数为 $X_{Totl}$ 。

表5.4列出了耗时与各种加速比的计算结果。可以看出，本文方法的总耗时主要消耗在建模时间上；只要提取出了模型参数，由于本文的模型复杂度很低，采用式4-5、4-9、式4-11 计算温度与静态功耗就是一个非常快速的计算过程（即 $T_{Anls}$ 非常小，只占 $T_{Totl}$ 很小的部分）。与HotSpot软件相比，本文3种方法分析计算的加速比 $X_{Anls}$ 分别达到50、147、和66；但考虑到 $T_{Anls}$ 只占 $T_{Totl}$ 很小的部分，所以本文3种方法总耗时的加速比只能达到13、15、和14。

从表5.2-表5.4的算法精度与复杂度的比较结果[18]可以看出：与HotSpot软件相比，本文的BlockTAM和BlockInsideCoreTAM在满足精度要求的前提下（热点温度误差小

于2.2%），获得了满意的加速效果，总耗时的可以达到13倍和14倍的加速效果。

## 5.5 小结

第4章中，采用自下而上的建模方法对MPSoC结构级热分析方法进行了探索，同时提出了3种具有不同算法复杂度与精度的热分析方法：模块级方法BlockTAM、核级方法CorTAM、考虑本核内模块相互影响的改良核级方法BlockInsideCoreTAM。这三种算法之所以能在保证分析计算精度的前提下、获得比较好的加速效果，归根结底是因为对整个芯片的热分析模型从结构级进行了合理适度的等效简化。本章对第4章中所提出的各种算法进行了大量的实验验证。实验数据数据表明：对核数较多MPSoC进行热分析的时候，BlockTAM和BlockInsideCoreTAM均具有算法复杂度低和精度高的优点：平均相对增量误差最大不超过3%，同时可以获得14倍左右的运算加速效果。

## 6 研究总结与展望

### 6.1 本文总结

面向复杂应用的高性能片上系统为了规避和减轻功耗墙问题，延续摩尔定律，采用了实时温度功耗管理与多核并行计算结构两种主要的技术手段。本文对这两种技术手段中的一些技术问题做了较为深入的研究。第2章-第3章对温度敏感的实时功耗调度和多核芯片的热分析方法这两个不同领域，分别做了较为深入的研究，并取得了如下成果。

首先，为了构建一个高效的实时功耗温度管理系统，本文不仅提出了一种具有高精度的组合式任务预测方法，而且还提出一种新的实时功耗温度管理任务调度算法VP-TALK，并进一步集成了一个基于负载预测的实时功耗温度管理原型系统，该系统主要包括工作负载预测、任务实时调度两大模块。基于多种调度算法的实时调度模块：先根据对工作负载率的精确预测值、计算出最优工作状态的电压/频率理想值，再从系统的电压/频率对的实际设定值中选取相邻的两个工作状态，最后考虑系统实时性、温度上限限制、静态功耗与温度的敏感关系以及芯片模式切换代价等多种因素，利用机器学习的方法，选择一种最优的调度策略。大量的模拟实验表明：

- 1) 在负载预测方面，本文实时功耗温度管理系统所采用的组合任务预测方法胜过众多的相关模型及算法，平均误差仅为2.89%；
- 2) 在节能效果方面，当负载率高于55%时，基于相同的峰值温度约束，本文所提出的VP-TALK算法分别比Pattern-based、M-oscillating和TALK对比算法节能约20.5%、11.0%、11.5%；
- 3) 本文实时功耗温度管理原型系统的调度效果接近于理想调度效果。

第4章-第5章中，采用自下而上的策略，使用HotSpot提取MPSoC功能模块之间的热相关系数，建立了模块级热分析方法BlockTAM；仅依靠热点之间的热相关系数、建立一个算法复杂度非常低的核级热分析方法CoreTAM；结合BlockTAM与CoreTAM两种方法，进一步提出了考虑本核内模块相互影响的改良核级方法BlockInsideCoreTAM。与现有的结构级热分析算法相比，本文所提出的三种方法均具有简单、高效、与现有简化模型兼容、易于扩展、考虑LDT影响等优点，可以满足温敏MPSoC设计对高效、精确的结构级热分析方法的需求。与HotSpot软件的实验结果相比，本文方法的实验数据表明：

- 1) 对核数较多MPSOC进行局部热点温度分析的时候，BlockTAM和BlockInsideCoreTAM只产生2%、3%以下的温度增量平均误差，是高精度的结构级热分析方法。
- 2) 在采用电压频率调节的温敏16核CPU布图规划研究中，在包含参数提取时间的情况下，BlockTAM和BlockInsideCoreTAM可以提供50倍左右的计算加速。
- 3) 从总体效果来看，在本文所提出三种建模分析方法中，BlockTAM和BlockInsideCoreTAM方法可以提供满意的分析精度与计算加速，是较为理想的MPSOC结构级热分析方法。

## 6.2 未来工作展望

第2章所构建的动态温度与功耗管理原型系统还只能解决实时嵌入式单CPU系统上的高效任务调度。然而，移动互联网的迅猛发展，已经推动大多数的嵌入式、移动便携式设备也广泛开始采用多核心的体系架构[37,38]，更有一些芯片制造商将GPU迁入这些移动设备[39]。在这些设备上，虽然CPU核心的计算能力与GPU核心的渲染能力被提高了2-8倍，但不幸的是，耗电量必然地也有了巨大的增长。故而，针对这些多核心的移动设备与便携设备，可能会存在如下一些亟待解决的问题：

- 1) 以最大化吞吐量为目标：当有了更多地计算资源，尤其是可进行并行处理的计算资源后，如何将任务分配至可用计算资源上；或者如何对任务预处理，分解为独立的子任务后，从而最大程度的提高所有计算资源利用率
- 2) 以最小化耗电量为目标：在保障任务被实时处理的前提下，如何才能做到将能耗、温度、耗电量这些计算所产生的副作用降到最低，从而提高设备的单次使用时长和整体使用寿命

虽然第4章所提出的稳态热分析算法能够对多核芯片温敏调度中的分析环节带来明显的加速，从而进一步降低调度或者设计过程中的耗时，但是事实上这些算法仍然存在着以下一些问题：

- 1) 预提取时间过长：不论是BlockTAM还是BlockInsideCoreTAM分析算法，都需要在分析之前，根据芯片的布局规划利用HotSpot软件进行参数提取。从第章的实验结果看来，这部分时间是所占的比例相当大，但仍然可以接受。然而，特别的，如果所要分析的芯片的布局不断变化，那么重复多次地利用HotSpot进行参数提取可能会造成很严重的分析耗时。于是，必须找到一种不依赖于芯片布局规划的参数提取办法
- 2) 支持瞬态热分析：第4章中所讨论的所有算法都是稳态热分析算法。如果能提供对某

## 6 研究总结与展望

---

个功耗变化过程的瞬态热分析的话，可能对实际的芯片设计与分析的意义和帮助更大。比如,HotSpot就提供的瞬态热分析的能力。但是，瞬态热分析的复杂度和对分析速度的需求是不成比例的。是否能够扩展本文中结构级热分析算法，使其在保障热分析速度的前提下支持瞬态热分析？这需要更进一步的研究探索

## 参考文献

1. S. Borkar. Thousand Core Chips: A Technology Perspective[C]. Proceedings of the 44th Design Automation Conference, San Diego, USA, 2007. 746–749.
2. G. Moore. Progress in Digital Integrated Electronics[C]. Proceedings of IEDM Tech Digest, Washigdon D.C., USA, 1975. 11–13.
3. B. Zhai. Theoretical and Practical Limits of Dynamic Voltage Scaling[C]. Proceedings of the 40th Design Automation Conference, San Diego, USA, 2004. 868–873.
4. R. Jejurikar, C. Pereira, R. Gupta. Leakage Aware Dynamic Voltage Scaling for Real-Time Embedded Systems[C]. Proceedings of the 40th Design Automation Conference, San Diego, USA, 2004. 275–280.
5. T. Chantem, R. P. Dick, X. S. Hu. Temperature-aware Scheduling and Assignment for Hard Real-time Applications on MPSoCs[C]. Proceedings of Design, Automation & Test in Europe, Munich, Germany, 2008. 288–293.
6. M. Santarini. Thermal Integrity: A Must for Low-power-IC Digital Design[J]. Sustainable Computing, Informatics and Systems, 2011, 4:286–293.
7. Z. G. Fu, C.S. Sun, Z. Y. Luo. A Task Scheduling Algorithm of Real-time Leakage Power and Temperature Optimization[C]. Proceedings of Computer Aided Design and Computer Graphics, Yellow Mountain City, China, 2009. 484–491.
8. Berkeley BSIM3 Device Models[Z]. [EB/OL]. <http://www.device.EECS.Berkeley.edu/bsim3>.
9. H. Sanchez, B. Kuttanna, T. Olson. Thermal Management System for High Performance Power PC Microprocessors[C]. Proceedings of International Conference on Technologies for the Information Superhighway, San Jose, USA, 1997. 325–330.
10. W. Huang, M. R. Stan, K. Sankaranarayanan. Many-core Design from a Thermal Perspective[C]. Proceedings of the 45th Design Automation Conference, Anaheim, California, USA: New York: ACM Press, 2008. 746–749.
11. M. B. Healy, H. H. S. Lee, G. H. Loh. Thermal Optimization in Multi-granularity Multi-core Floorplanning[C]. Proceedings of Asia and South Pacific Design Automation Conference. Piscataway, NJ: IEEE Press, 2009. 43–48.
12. K. Michael, R. Sherief. Frequency and Voltage Planning for Multi-core Processors under Thermal Constraints[C]. Proceedings of International Conference on Computer Design, Cancun, Mexico: Los Alamitos: IEEE Computer Society Press, 2008. 463–470.

13. K. Sankaranarayanan, B. H. Meyer, M. R. Stan. Thermal Benefit of Multi-core Floorplanning: A Limits Study[J]. Sustainable Computing, Informatics and Systems, 2011, 4:286–293.
14. V. Hanumaiah, R. Rao, S. Vrudhula. Throughput Optimal Task Allocation under Thermal Constraints for Multi-core Processors[C]. Proceedings of the 46th Design Automation Conference. New York: ACM Press, 2009. 776–781.
15. Y. Ge, Q. R. Qiu. Task Allocation for Minimum System Power in a Homogenous Multi-core Processor[C]. Proceedings of International Green Computing Conference. Los Alamitos: IEEE Computer Society Press, 2010. 299–306.
16. C. L. Lung, Y. L. Ho, D. M. Kwai. Thermal-aware Online Task Allocation for 3D Multi-core Processor Throughput Optimization[C]. Proceedings of Design Automation & Test in Europe, Grenoble, France: New York: ACM Press, 2011. 1–6.
17. D. C. Juan, D. Marculescu. A Learning-based Autoregressive Model for Fast Transient Thermal Analysis of Chip-multiprocessors[C]. Proceedings of Asia and South Pacific Design Automation Conference, Sydney, Australia: Piscataway, NJ: IEEE Press, 2012. 597–602.
18. 闫佳琪, 骆祖莹, 唐亮, et al. 考虑温度对漏电流功耗影响的MPSoC结构级热分析方法[J]. 计算机辅助设计与图形学学报, 2013, 25(11):1767–1774.
19. K Skadron, T. Abdelzaher, R. M. Stan, et al. Control-Theoretic Techniques and Thermal-RC Modeling for Accurate and Localized Dynamic Thermal Management[C]. Proceedings of High Performance Computer Architecture, Boston, USA, 2002. 17–28.
20. K. Skadron, R. M. Stan, K. Sankaranarayanan. Temperature-aware Microarchitecture: Modeling and Implementation[J]. ACM Transactions on Architecture and Code Optimization, 2004, 1(1):94–125.
21. J Srinivasan, S. V. Adve. Predictive Dynamic Thermal Management for Multimedia Applications[C]. Proceedings of International Conference on Supercomputing, San Francisco, USA, 2003. 109–120.
22. L. Yuan, S. Leventhal, G. Qu. Temperature-aware Leakage Minimization Technique for Real-time Systems[C]. Proceedings of International Conference on Computer Aided Design, San Jose, USA: Piscataway, NJ: IEEE Press, 2006. 761–764.
23. C. Y. Yang, L. Thiele, T. W. Kuo. Energy-efficient Real-Time Task Scheduling with Temperature-Dependent Leakage[C]. Proceedings of International Conference on Computer Aided Design, Dresden, Germany, 2010. 9–14.
24. M. Bao, A. Andrei, P. Eles. Temperature-aware Idle Time Distribution for Energy Optimization with Dynamic Voltage Scaling[C]. Proceedings of Design, Automation & Test in Europe, Dresden, Germany, 2010. 21–27.

25. H. Huang, G. Quan. Leakage Aware Energy Minimization for Real-Time Systems under the Maximum Temperature Constraint[C]. Proceedings of Design, Automation & Test in Europe, Grenoble, France, 2011. 479–484.
26. Y. Zhan, B. Goplen, S. S. Sapatnekar. Electrothermal Analysis and Optimization Techniques for Nano-scale Integrated Circuits[C]. Proceedings of Asia and South Pacific Design Automation Conference, Yokohama, Japan: Piscataway, NJ: IEEE Press, 2006. 219–222.
27. W. Huang, S. Ghosh, S. Velusamy. HotSpot: A Compact Thermal Modeling Methodology for Early-stage VLSI Design[J]. IEEE Transactions on Very Large Scale Integration Systems, 2006, 14(5):501–513.
28. M. Janicki, J. H. Collet, A. Louri. HotSpots and Core-to-core Thermal Coupling in Future Multi-core Architectures[C]. Proceedings of the 26th IEEE Semiconductor Thermal Measurement and Management Symposium. Los Alamitos: IEEE Computer Society Press, 2010. 205–210.
29. L. Thiele, S. Chakraborty, M. Naedele. Real-time Calculus for Scheduling Hard Real-time Systems[C]. Proceedings of International Symposium on Circuits and System, Geneva, Switzerland, 2000. 101–104.
30. V. Chaturvedi, H. Huang, G. Quan. Leakage-aware Scheduling on Maximal Temperature Minimization for Periodic Hard Real-time Systems[C]. Proceedings of International Conference on Embedded Software and System, Bradford, UK, 2010. 1802–1809.
31. D. Rai, H. Yang, I. Bacivarov. Worst-Case Temperature Analysis for Real-Time Systems[C]. Proceedings of Design, Automation & Test in Europe, Grenoble, France, 2011. 631–636.
32. 闫佳琪, 骆祖莹, 赵国兴. 基于任务精确预测的实时功耗温度管理[C]. Proceedings of 第十七届全国计算机辅助设计与图形学学术会议暨第九届全国智能CAD与数字娱乐学术会议论文集, 北京: 清华大学出版社. 513–516.
33. MATLAB[Z]. [EB/OL]. <http://www.mathworks.com/products/matlab/>.
34. K. Skadron, M. R. Stan, W. Huang. Temperature-aware Microarchitecture[C]. Proceedings of International Symposium on Computer Architecture. Los Alamitos: IEEE Computer Society Press, 2003. 2–13.
35. W. P. Liao, L. He, K. M. Lepak. Temperature and Supply Voltage Aware Performance and Power Modeling at Microarchitecture Level[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and System, 2005, 24(7):1042–1053.
36. Intel Sandy Bridge-E架构的8核高端CPU[Z]. [OL]. <http://ark.intel.com/products/63696>.
37. 采用多核结构的iPhone移动设备[Z]. [OL]. <http://www.apple.com/cn/iphone/compare/>.
38. 采用多核结构的iPad移动设备[Z]. [OL]. <http://www.apple.com/cn/ipad/compare/>.
39. 手机GPU芯片图形性能跑分横向对比[Z]. [OL]. <http://www.dashi.com/article/1148.html>.

## 学术成果

1. Jiaqi Yan, Zuying Luo, Liang Tang: 《Accurate Architecture-level Thermal Analysis Methods for MPSoC with Considering Leakage Power Dependence on Temperature》 ISQED2013: 178-183. (EI 收录, 检索号: 20132716473440)
2. 闫佳琪, 骆祖莹, 唐亮, 赵国兴: 《考虑温度对漏电流功耗影响的MPSoC结构级热分析方法》 计算机辅助设计与图形学学报, 2013, 24 (11) : 1767-1774 (EI源刊)
3. 闫佳琪, 骆祖莹, 赵国兴: 《基于任务精确预测的实时功耗温度管理》, 中国图形学进展(2012)- 第十七届全国计算机辅助设计与图形学学术会议(CAD/CG‘2012)暨第九届全国智能CAD与数字娱乐学术会议(CIDE‘2012)论文集, 2012,7:513

## 致 谢

人生无根蒂，飘如陌上尘。  
分散随风转，此已非常身。  
落地为兄弟，何必骨肉亲。  
得欢当作乐，斗酒聚比邻。  
盛年不重来，一日难再晨。  
及时当勉励，岁月不待人。

首次接触陶渊明的这首杂诗，还是在14年前的书法课上，徐伟老师每节课一句地教授我们如何眷写。转眼十年过去，我曾在首都博物馆与徐老师有过一面之缘，那时，徐老师已经是北京书画艺研会副会长，并兼管首都博物馆的陈列展览事宜，而我也就要本科毕业，并决定继续留校读研究生。徐老师早已认不出我这个并非正规的书法弟子，我也只能尴尬地寒暄了几句。想想确实是“人生无根蒂，破如陌上尘”。

紧接着便是研究生入学后结识一些新面孔，当然由于各种原因，本科时很多不太熟悉的旧面孔成了每天更为亲近的面孔。前者的一个典型例子就是沉迷游戏、并最终会靠游戏发家致富的严智同学；后者的典型例子就是与我通一个导师的唐亮同学和最后不是和我一个导师的张儒少同学。我们四个人组成了601寝室。每日在学习与科研之余，可以熄灯泡脚夜聊，可以香蕉甜橙西瓜，可以啤酒炸鸡英超，可以泡面火锅日昌，真所谓“落地为兄弟，何必骨肉亲。得欢当作乐，斗酒聚比邻”。

转眼间，研究生生涯行将结束，也确实每天劳苦奋战，也确实让身体和身材都大不如前，白发多了，肚子圆了。虽然也能在简历里写入些许学术成果，但是仍然不免感叹光阴荏苒，还有很多事情可以做、很多事情值得做，而这些事情却来不及在这三年里触及。也正应了这最后两句诗：“盛年不重来，一日难再晨。及时当勉励，岁月不待人”。

在此，我要感谢这些在我的研究生生涯里给予了我无限鼓励、帮助的人：

感谢我的父母。虽然我的母亲由于历史原因只能完成高中，但是她却有着胜过学历的人生智慧。她的人生虽然经历了很多困难挫折，她的身心也承受着很多痛苦，但是她却没有让我以一种消极的态度对待人生，反而用自己的行动告诉了我应该乐观豁达的对待生活，并且鼓励我自己闯出一番事业，不像很多同龄人一样虚度青春。我也要感谢我的父亲。纵使在很多人看来，他并不是一个值得称赞的人，但是没有他每天的辛勤工作，没有他的供养，我的求学之路也许早已经结束。总之，没有我的父母对我学习上的

## 致谢

---

支持与生活上的关爱，我绝不可能做出今天的研究成果。

感谢我的研究生导师骆祖莹副教授。骆老师为人宽厚和蔼，却又很懂得激励我们在科研上投入足够的时间与精力。我研究生的所有成果都是在骆老师的指导下完成的，并且骆老师因为我需要出国的原因，把所有的论文第一作者的位置都让给了我。其实多数的想法都是源自于骆老师，最后完整的idea和实现是我和骆老师无数次的交谈、讨论和验证中实现的。感谢在我一筹莫展的时候，骆老师能为我的科研之路指明方向。在我需要一个科研伙伴时，我永远能够找到骆老师，我相信，国内太多的研究生导师都没有骆老师一半地对学生认真负责。

感谢同实验组的赵国兴老师。赵老师是我们组内不可多得的人才，可谓十八般武器样样精通。他扎实的数学功底，严谨的逻辑推理思维，让我们在耳濡目染下，获得了许多科研上的意外收获。更难得的是，赵老师经常和我们打成一片，探讨各种工程与科学问题：小到软件破解、硬件升级，大到社会现状、人生感悟，而且往往是边聊边请我们吃了一顿大餐。三年下来，欠下了赵老师无数顿饭钱，在此不知如何感谢。

感谢高性能计算实验组曾经与现在的同学，他们是杨旭、黄琨、唐亮、李晓怡、王红蕊、王嘉琪、邹甜、唐传高等同学。特别感谢我的研究伙伴唐亮，三年来我同他一起在实验室奋斗或者打游戏。感谢杨旭与黄琨师兄在我刚进实验室之后对我的照顾。感谢李晓怡与王红蕊为我们报销。感谢邹甜经常能找到好地方为我们改善伙食。有他们陪伴的研究生三年是我一生中最珍贵的时光。

感谢虚拟现实实验室主任与信息科学与技术学院院长周明全教授。周老师虽然每日忙于各种项目事宜，但在每个学生需要他的时候，却并没有任何院长的架子，总是慷慨地给予帮助。感谢虚拟现实实验室的全体老师和同学们平日里的的帮助和支持！

此外，感谢北京师范大学天文系的余恒老师，虽然他并不认识我，但是一定有很多像我一样的人会由衷的感谢他的工作：因为他制作维护的北京师范大学学位论文模板极大的方便了 $\text{\LaTeX}$ 用户的论文写作。

最后，本文所有工作都承蒙国家自然科学基金资助，特此致谢。

闫佳琪  
2014年4月