CN43-1258/TP
ISSN 1007-130X
计算机工程与科学
COMPUTER ENGINEERING & SCIENCE
2010 年第 32 卷第 11 期
Vol. 32, No. 11, 2010

文章编号:1007-130X(2010)11-0114-05

微处理器体系结构级软错误易感性评估*

Evaluating the Microprocessor Architectural Vulnerability to Soft Errors

孙 岩,王永文,张民选

SUN Yan, WANG Yong-wen, ZHANG Min-xuan

(国防科学技术大学计算机学院,湖南 长沙 410073)

(School of Computer Science, National University of Defense Technology, Changsha 410073, China)

摘 要:随着集成电路特征尺寸的缩小和集成度的增加,微处理器的软错误问题越来越严重。为了提高微处理器的可靠性,设计者需要在体系结构设计时精确估算各个部件的软错误率,从而对各部件进行相应的容错设计。本文针对微处理器中的软错误问题,研究了体系结构级软错误易感性估算模型,基于该模型对超标量微处理器主要部件的软错误易感性进行定量分析,并讨论了可靠性与性能的折衷设计。实验结果对微处理器软错误的预防和保护具有一定指导意义,也为微处理器主要部件的容错设计提供了参考。

Abstract: With the reduction in IC feature size and increase in integration, the soft error problem in microprocessors is becoming more and more serious. In order to improve the reliability of microprocessors, designers need to estimate the soft error rate of various components accurately for their fault-tolerance design. In this paper, the estimation model for the microprocessor architectural vulnerability to soft errors is studied. Based on the model, major components' vulnerability to soft errors in superscalar microprocessors is evaluated. The trade-off design between reliability and performance is also discussed. The experimental results can guide microprocessors' soft error prevention and protection, and can also provide references for the fault-tolerance design of major components in microprocessors.

关键词:可靠性;软错误;结构易感因子;微处理器

Key words: reliability; soft error; architectural vulnerability factor; microprocessor

doi: 10.3969/j.issn.1007-130X.2010.11.031

中图分类号:TP302

文献标识码:A

1 引言

软错误(Soft Error)又称为瞬态错误(Transient Error),它主要由放射性粒子等因素导致,因此也称为放射诱发的软错误(Radiation-induced Soft Error)[1]。过去认为放射性粒子不会对地面的集成电路产生影响,软错误的研究主要集中在航空航天领域。但是,随着集成电路的高速发展,芯片工艺尺寸和供电电压不断降低,芯片内的节点电量也随之降低,这使得地面上的集成电路也可能由于较少的放射性粒子而发生软错误,同时单个芯片中的器件数急剧增加,也增大了发生软错误的概率[2]。另外,集成电路封装材料中也含有放射性物质,这使得集成电路的软错误问题更加严重。软错误问题随着工艺尺寸的缩小以及系统复杂性的升高将会越来越严重,影响了系统的可靠性、可用性和安全性[3]。

微处理器是电子设备的核心部件,一旦发生软错误,可能引起严重的损害。与航空航天领域不同的是,地面上的微处理器由于成本和性能的限制,很难实现规模较大的冗余(如时空三模冗余[4])或代价较高的错误保护方案(如全 ECC 保护[5])。因此,需要对微处理器各主要部件的可靠性进行评估,从而根据各部件的可靠性特点设计不同的保护方案,实现成本-可靠性的较好折衷。

微处理器的软错误易感性可以通过高能粒子辐照实验

* 收稿日期:2009-01-02;修订日期:2009-04-28
基金项目:国家自然科学基金资助项目(60703074, 60873016);国家 863 计划资助项目(2009AA01Z102)
作者简介:孙岩(1980-),男,甘肃兰州人,博士生,研究方向为微处理器与 VLSI 设计技术;王永文,副研究员,研究方向为计算机体系结构与微电子技术;张民选,教授,博士生导师,研究方向为高性能计算机系统及其实现技术、微电子技术。
通讯地址:410073 湖南省长沙市国防科学技术大学计算机学院博士生队;Tel:13973117440;E-mail:yansun.nudt@gmail.com
Address: Doctoral Brigade, School of Computer Science, National University of Defense Technology, Changsha, Hunan 410073, P.R. China

来进行评估，但这种方法需要实际的微处理器芯片，无法在设计初期进行，不能指导设计。高能粒子辐照实验还需要昂贵的设备，成本很高。因此，需要通过模拟的方法分析芯片的软错误率。由于软错误在体系结构级有很多屏蔽效应，在体系结构级还可以更好地开发程序负载特性，另外软错误的概念本身就在体系结构级定义，因此在体系结构级对芯片的软错误易感性进行评估，具有重要的理论意义和实用价值[6]。

本文针对微处理器体系结构级软错误易感性进行研究，分析了体系结构级软错误易感性评估指标和估算模型，并使用该模型对超标量处理器主要部件的软错误易感性进行评估和分析。实验结果对微处理器软错误的预防和保护具有一定指导意义，也对微处理器主要部件的设计提供了参考和评估手段。

2 软错误易感性评估指标

当前，系统的可靠性主要由平均失效时间 $MTTF$（Mean Time To Failure，简称 MTTF）和失效率 FIT（Failures In Time，简称 FIT）来衡量，这已成为工业界和研究界公认的标准。$MTTF$ 是指系统平均发生一次失效的时间，它近似等于平均失效间隔时间。而 FIT 是指系统每运行 10^9 小时平均发生的失效次数，也就是系统的失效率。$MTTF$ 与 FIT 为反比关系，当 $MTTF$ 为 1 000 年时，大约相当于 FIT 为 114，这也是商用服务器可靠性的一个典型目标。

系统总的失效率 FIT_{system} 是系统各个组成部件失效率 FIT_i 之和，即：

$$FIT_{system} = \sum_i FIT_i \tag{1}$$

而对于某个部件 i，其失效率可表示为：

$$FIT_i = IER_i \times VF_i \tag{2}$$

其中，IER 为部件的本征错误率（Intrinsic Error Rate，简称 IER），它与生产工艺、电路结构等因素有关；VF 为部件的软错误易感因子（Vulnerability Factor，简称 VF），它是时序易感因子 TVF（Timing Vulnerability Factor，简称 TVF）和结构易感因子 AVF（Architectural Vulnerability Factor，简称 AVF）之积，即：

$$VF = TVF \times AVF \tag{3}$$

TVF 表示部件容易发生软错误的时间窗口占整个运行时间的百分比，对于 SRAM 单元，TVF 为 1，对于锁存器和动态逻辑，TVF 近似为 0.5，而对于静态逻辑，在当前的工艺水平下 TVF 较小，可近似认为是 0[1]；AVF 表示部件发生的错误引起系统级错误的概率。因此，系统总的失效率可表示为：

$$FIT_{system} = \sum_i (IER_i \times TVF_i \times AVF_i) \tag{4}$$

系统在 $[0, t]$ 阶段的可靠性（Reliability）定义为在该阶段中系统正确运行的概率。在 t 时刻系统的可靠性可表示为：

$$Reliability_{system}(t) = e^{-t \times FIT_{system}} = e^{-\frac{t}{MTTF_{system}}} \tag{5}$$

对于给定的结构，本征错误率和时序易感因子可以近似认为是常数。因此，要评估微处理器的可靠性，最重要的是得到微处理器各个部件的结构易感因子 AVF。

3 软错误易感性估算模型

3.1 估算方法

AVF 反映了微处理器中的位发生错误是否会对程序的结果产生影响，如果得到微处理器各个部件的 AVF，我们就可以比较容易地算出整个处理器的 $MTTF$ 和可靠性。AVF 可以通过 ACE（Architecturally Correct Execution，简称 ACE）分析来估算。ACE 位是为了保证程序正确执行而必须保持正确的位。在某个部件中，ACE 位所占的比重就是该部件的结构易感因子 AVF。ACE 分析可以集成到体系结构模拟器中，对于某种给定的结构和测试程序，只需运行一次就可得到 AVF。由于速度较快，可以运行较多测试程序，从而提高精度。ACE 分析方法由 Mukherjee 等人于 2003 年首次提出[7]，用于计算基于指令结构的 AVF，之后又扩展到基于地址结构的存储部件上[8]。现在被学术界广泛使用。本文将采用 ACE 方法来估算微处理器的软错误易感性。

3.2 基于指令的结构

对于指令队列等基于指令的结构，通常使用 Little's Law 来计算 AVF，它可简单地表示为：

$$N = B \times L \tag{6}$$

其中，N 表示一个结构的平均位数，B 表示每周期进入结构的平均带宽，L 表示每个对象通过结构的平均延时。对于某个结构，其 AVF 就是该结构中所有位中 ACE 位的比例。只要得到了进入该结构的 ACE 位平均带宽 B_{ACE}、ACE 位在该结构中的平均延时 L_{ACE} 以及该结构的总位数，就可以通过 Little's Law 计算出该结构的 AVF：

$$AVF_{structure} = \frac{B_{ACE} \times L_{ACE}}{N_{structure}} \tag{7}$$

确定某位为非 ACE 比确定为 ACE 更容易。Mukherjee 等人总结了 4 种微体系结构和 5 种体系结构的非 ACE 位[7]。除了这些非 ACE 位外，其余位是否为 ACE 还需要进一步判断。由于少数位无法判断是否 ACE，属于未知状态，因此通过 ACE 方法计算 AVF 通常得到 AVF 的上限，虽然不太精确，但还是能反映微处理器的软错误易感性趋势，对于微处理器的设计仍有指导意义。另外，Biswas 等人指出，只需要增加少量的处理器细节信息，ACE 方法就可分析出更加精确的 AVF[9]。

3.3 存储器数据阵列

存储器等基于地址的结构占微处理器芯片面积的绝大部分，这些结构也最易受软错误攻击。为了得到精确的 AVF，对于基于地址的结构采用专门的 ACE 分析方法。其中，数据阵列和标识阵列由于结构不同，其 ACE 方法也完全不同。本节介绍数据阵列的 ACE 分析方法，3.4 节介绍标识阵列的 ACE 分析方法。

对于存储器数据阵列使用生命周期（Lifetime）的 ACE 分析方法。生命周期分析方法根据存储器的操作，将存储器每一位的生命周期分为几种阶段，如 idle、fill-read、read-read 等，如图 1 所示。由于没有数据被读出或写回到下一

级存储层次中，因此 idle、read-write、write-write 和 evict-fill 为非 ACE 状态。而 fill-read、read-read、write-read 和 read-evict 是否 ACE 还需要进一步判断，因为在这些阶段虽然有数据读或写到处理器及其它存储层次，但不一定会引起程序的执行错误，如该位的数据用于前瞻操作。

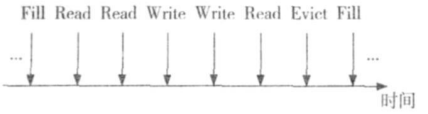

图 1　存储器生命周期

待定的四种情况是否 ACE 取决于 read 是否 ACE，以及其结构特点。若 read 本身为 ACE，则相应元素为 ACE。Read 是否 ACE 可由指令是否 ACE 得知。Read-evict 是否 ACE 由结构决定：若为写直达 Cache，则为非 ACE；若为写回 Cache，则需要根据 Cache 中的数据是否脏数据来进一步判断。

在得到数据阵列某段执行的所有位 ACE 时间之和 t_{ACE} 以及执行时间 t_{total} 和结构的总位数 $N_{structure}$ 后，就可计算出该数据阵列的 AVF：

$$AVF = \frac{t_{ACE}}{t_{total} \times N_{structure}} \tag{8}$$

3.4　存储器标识阵列

对于存储器的标识阵列(Tag Array)，使用海明距一(Hamming-Distance-One)的 ACE 分析方法。存储器标识阵列的主要操作是将输入的数据与 CAM 中的数据进行匹配。如果标识阵列中的数据发生软错误，则可能发生两种情况：(1)假阳性(False Positive)：输入数据与 CAM 中数据原本不匹配，但由于软错误变为匹配状态；(2)假阴性(False Negative)：输入数据与 CAM 中数据原本应当匹配，但是由于软错误成为不匹配状态。

对于假阳性，使用海明距一分析。在该方法中，假设只会发生一位错误。在这种情况下，当且仅当输入数据和 tag 中数据只有一位不同时，才有可能发生假阳性，这时只有不同的那一位是 ACE 位。对于假阴性，只有当输入数据和 tag 中数据相同时，才有可能发生，这时所有位都同时为 ACE 或非 ACE 位，这取决于错误发生后是否引起错误执行。

假阳性和假阴性发生后对程序执行的影响是不同的，对于写直达 Cache、DTLB 和 Store 缓冲三种结构，在假阳性发生后都会发生错误，这是因为错误的匹配会引起错误的数据操作；而假阴性发生后，只有 Store 缓冲有可能发生错误，这是因为不匹配会导致重新访问下一级存储层次，对于多级存储层次的结构，在下一级存储层次中存有副本，因此不会发生错误[8]。

4　实验结果与分析

4.1　实验方法

Sim-SODA 是美国佛罗里达大学的研究人员基于 Alpha-21264 结构开发的体系结构级可靠性分析工具，它覆盖了大部分微处理器部件，采用细粒度的分析方法，并使用 Cooldown 技术解决边缘效应，能够实现较精确的分析[10]。我们使用 Sim-SODA 工具，在 SPEC2000 中选择 12 组作为测试激励，其中整数和浮点各 6 组，使用 reference 输入集。在模拟时，我们跳过前 500M 条指令，模拟 100M 条指令。对于 Cache 等存储结构，我们设置 100M 条指令作为 Cooldown 阶段。模拟的处理器基准配置如表 1 所示。

表 1　模拟器基准配置

项目	配置
流水线深度	7 级
指令流出队列	20 项
再定序缓冲	80 项
整数/浮点部件	4 组/1 组
牺牲缓冲	8 项
Load/Store 队列	32 项/32 项
寄存器文件	80 项
L1 指令 Cache	64KB，2 路组相联，块大小 64Byte
L1 数据 Cache	64KB，2 路组相联，块大小 64Byte
ITLB/DTLB	128 项，全相联/128 项，全相联
L2 Cache	2 048KB，直接映射，块大小 64Byte

4.2　微处理器软错误易感性概貌

图 2 表示了主要部件 AVF 的平均值。可以看到，AVF 较大的部件有 L1 数据 Cache 标识阵列、寄存器文件、DTLB 数据阵列和标识阵列、L1 数据 Cache 数据阵列等，都在 25%以上，甚至接近 50%。而 Load 队列、Store 队列和牺牲缓冲的 AVF 则相对较小，都在 10%以下。指令队列、再定序缓冲和功能单元的 AVF 居中，约在 10%和 20%之间。整数和浮点测试程序的 AVF 也有一定不同，对于多数部件，浮点测试程序比整数测试程序的 AVF 略高或近似，但 DTLB 的数据阵列、标识阵列及指令队列，浮点测试程序的 AVF 却低很多，约为整数测试程序 AVF 的一半。这说明 AVF 不仅与微处理器的结构有关，而且与应用特点也具有相关性。在进行高可靠性处理器设计时，需要优先对 AVF 较高的部件进行可靠性设计优化或容错设计，从而用最小的代价换取最大的可靠性提升。另外，在进行可靠性设计时，还要考虑处理器主要的应用环境，根据具体的应用来进行设计优化，从而实现可靠性与其它设计要素的最佳折衷。

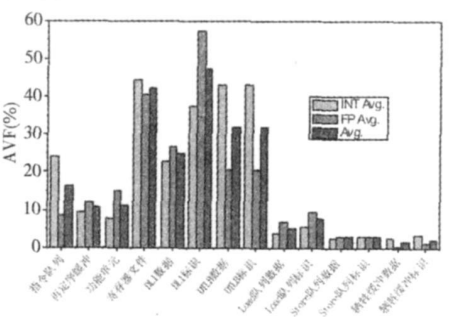

图 2　微处理器软错误易感性概貌

4.3　AVF 敏感性分析

微处理器部件的数量不仅对性能有影响，对软错误易感性也有影响。一方面，增加部件将增大易感面积，这使得部件更容易发生软错误；另一方面，增加部件可以加快执行速度，降低了易感时间，这又有利于降低软错误。为了分析软错误与部件数量与软错误易感性的关系，我们专门对多

个部件的 AVF 进行敏感性分析，实验结果如图 3 所示。

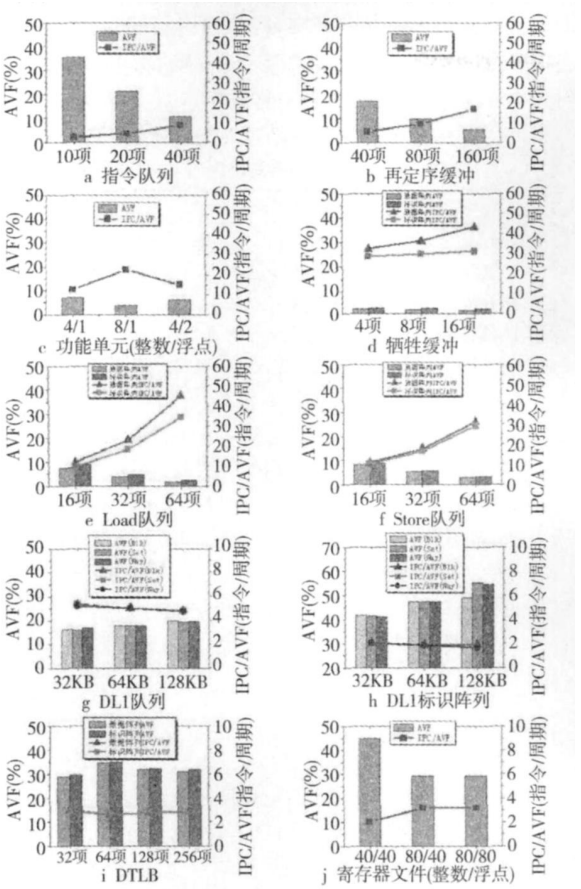

图 3　微处理器主要部件 AVF 及 IPC/AVF

对于多数部件，增加部件数量可以降低 AVF，也就是说，增加部件数量导致易感时间的降低比易感面积的增加更显著。这类部件有指令队列、再定序缓冲、Load 队列和 Store 队列，这些部件的易感性对部件数量的变化非常敏感，当部件数量增大为原来的两倍时，AVF 有显著降低，为原来的 50%～62%。再定序缓冲的可靠性对部件数量的变化相比之下不太敏感，当增大再定序缓冲为原来的两倍时，数据阵列和标识阵列的 AVF 平均降低为原来的 86% 和 95%。

L^1 数据 Cache 数据阵列和标识阵列的 AVF 随着 Cache 容量的增大而略有增加。为了实验各种可能的情况，分别通过改变 L^1 数据 Cache 的块大小、组大小和相联度来改变 Cache 大小，在图 3 中分别以 Blk、Set 和 Way 表示这三种情况。我们实验了 L^1 数据 Cache 大小为 32KB、64KB 和 128KB 的情况，结果显示，虽然 Cache 的 AVF 随容量增大而增加，但其敏感性较小。当容量增大为原来的两倍时，Cache 数据阵列和标识阵列 AVF 平均为原来的 109%和 112%。这与文献[11]和文献[12]的结果吻合。

还有一些部件情况略复杂，如功能单元、寄存器文件、DTLB 的数据阵列和标识阵列。当整数功能单元增为原来的两倍，AVF 显著降低，为原来的 56%；而保持整数功能单元不变，将浮点功能单元加倍后，AVF 降低为原来的 83%，因此整数功能单元 AVF 的敏感性更强。将整数寄存器文件加倍后，AVF 降为 65%，但增加浮点寄存器对 AVF 没有影响。对于 DTLB，我们分别对 32 项、64 项、128 项和 256 项进行了实验。实验显示，DTLB 由 32 项增加为 64 项时，AVF 有所增加，约为 32 项 AVF 的 119%，继续加倍 DTLB 的项数时，AVF 开始略微下降，分别为 92% 和 97%。

4.4　性能-可靠性联合评估

改变部件的参数一方面会影响部件的 AVF，另一方面还会影响处理器的性能。因此，在改变部件参数降低 AVF 的同时还要考虑对性能的影响，做到性能-可靠性的较好折衷。

$MTTF$ 表征了系统的可靠性，但没有考虑性能问题，无法反映性能-可靠性的折衷。为了解决该问题，文献[13]提出平均无失效指令 $MITF$(Mean Instruction To Failure，简称 MITF)的概念。$MITF$ 表征微处理器在发生失效前可以执行的平均指令条数，该度量标准可表示为：

$$MITF = IPC \times f \times MTTF = \frac{IPC \times f}{IER \times TVF \times AVF} = \frac{f}{IER \times TVF} \times \frac{IPC}{AVF} \tag{9}$$

其中，f 为处理器频率，IER 和 TVF 分别为本征错误率和时序易感因子。对于确定的微处理器部件，上述三者可认为是常数，因此 $MITF$ 与 IPC/AVF 成正比。从而，IPC/AVF 可以作为性能-可靠性联合评估的指标。在图 3 中，我们用折线图表示了各种情况下 IPC 与 AVF 的比值，从而更全面反映处理器参数改变对性能和可靠性的综合影响。部件的 IPC/AVF 越大，则部件的性能和可靠性的综合指标越好。

如图 3 所示，多数部件的 IPC/AVF 与 AVF 的倒数基本具有类似的趋势和幅度，如指令队列、再定序缓冲、功能单元、DTLB、Load 队列、Store 队列和牺牲缓冲。这说明，对于多数部件，单一部件参数的改变对 IPC 的影响不大，因此 AVF 的变化占主导地位。但是，对于寄存器文件、L^1 数据 Cache 数据阵列和标识阵列，IPC/AVF 与 AVF 的倒数趋势类似，但变化幅度更大。这说明这些部件参数变化对 IPC 的影响较大。这是因为寄存器文件和 L^1 数据 Cache 的位数很多，并且处于影响处理器性能的关键部分，需要对其频繁访问，因此对性能有较大的影响。增大这些部件，性能-可靠性的综合提升较大。

各个部件 IPC/AVF 值相对部件参数的变化情况，可以为微处理器各部件的参数设计提供理论支持和设计参考。例如，增大指令队列、再定序缓冲、牺牲缓冲、Load 队列和 Store 队列的大小可显著提高其 IPC/AVF 值，得到很好的性能-可靠性综合提升；对于 DTLB，基准设计已经是最佳情况，性能-可靠性折衷最好；对于 L^1 数据 Cache 的数据阵列和标识阵阵列，虽然改变 Cache 大小对其 AVF 影响不是特别大，但一是由于 Cache 绝对数量较大，二是其对性能影响很大，因此必须仔细考虑。从图 3 可以发现，改变 Cache 块大小比改变组大小和相联度的效果略好。

5　结束语

软错误易感性是现代微处理器设计需要考虑的重要问题。随着集成电路工艺的不断提高，由软错误引起的可靠性问题将越来越突出，应用于军事、航空航天以及科学计算

领域的微处理器必须考虑高能粒子引起的软错误问题，以确保系统的可靠性。

本文主要在体系结构级对微处理器的软错误易感性进行研究，分析软错误易感性的评估模型，对微处理器重要部件的软错误易感性进行量化研究和分析，从而对微处理器的容错设计进行有效的指导。

在未来的工作中，我们将进一步完善模拟平台，提高分析精度，并结合处理器的性能和功耗等因素提高微处理器的可靠性。

参考文献：

[1] Mukherjee S S, Emer J S, Reinhardt S K. The Soft Error Problem: An Architectural Perspective[C]// Proc of HPCA-11, 2005: 243-247.

[2] Shivakumar P, Kistler M, Keckler S W, et al. Modeling the Effect of Technology Trends on the Soft Error Rate of Combinational Logic[C]// Proc of DSN'02, 2002: 389-398.

[3] Mukherjee S. Architecture Design for Soft Errors[M]. Burlington: Morgan Kaufmann Publishers, 2008.

[4] 赖鑫，戴葵，刘芳，等. 高可靠 8051 微处理器的设计与实现[J]. 计算机工程与科学，2009，31(1)：117-120.

[5] Lee K, Shrivastava A, Issenin I, et al. Partially Protected Caches to Reduce Failures due to Soft Errors in Mission-Critical Multimedia Systems[R]. Technical Report TR-08-06, Center for Embedded Computer Systems, University of California, 2008.

[6] Li X, Adve S V, Bose P, et al. SoftArch: An Architecture-Level Tool for Modeling and Analyzing Soft Errors[C]// Proc of DSN'05, 2005: 496-505.

[7] Mukherjee S S, Weaver C, Emer J S, et al. A Systematic Methodology to Compute the Architectural Vulnerability Factors for a High-Performance Microprocessor[C]// Proc of Micro-36, 2003: 29-42.

[8] Biswas A, Racunas P, Cheveresan R, et al. Computing Architectural Vulnerability Factors for Address-Based Structures[C]// Proc of ISCA-32, 2005: 532-543.

[9] Biswas A, Racunas P, Emer J, et al. Computing Accurate AVFs Using ACE Analysis on Performance Models: A Rebuttal[J]. IEEE Computer Architecture Letters, 2008, 7(1): 21-24.

[10] Fu X, Li T, Fortes J A B. Sim-SODA: A Unified Framework for Architectural Level[C]// Proc of Workshop on Modeling, Benchmarking and Simulation, 2006.

[11] Asadi G, Sridharan V, Tahoori M B, et al. Balancing Performance and Reliability in the Memory Hierarchy[C]// Proc of ISPASS'05, 2005: 269-279.

[12] Cai Y, Schmitz M T, Ejlali A, et al. Cache Size Selection for Performance, Energy and Reliability of Time-Constrained Systems[C]// Proc of ASP-DAC'06, 2006: 923-928.

[13] Weaver C, Emer J S, Mukherjee S S, et al. Techniques to Reduce the Soft Error Rate of a High-Performance Microprocessor[C]// Proc of ISCA-31, 2004: 264-275.

(上接第 113 页)

法比以往方法精确 7%～13%，映射结果和通讯参数设计更优。本文所述方法虽然是面向 Mesh 结构的 NoC 映射问题，但同样适用于其他结构的 NoC 映射问题。本文目前所设计的通讯参数还比较有限，下一步希望能对更多 NoC 参数进行自动化设计，如路由器缓存等。

参考文献：

[1] Hemani A, Jantsch A, Kumar S, et al. Network on a Chip: An Architecture for Billion Transistor Era[C]// Proc of the IEEE NorChip Conf, 2000: 166-173.

[2] Benini L, Micheli G D. Networks on Chips: A New SoC Paradigm[J]. IEEE Computer, 2002, 35(1): 70-78.

[3] Hu J, Marculescu R. Energy and Performance-Aware Mapping for Regular NoC Architectures[J]. IEEE Trans on CAD of Integrated Circuits and Systems, 2005, 24(4): 551-562.

[4] Ascia G, Catania V, Palesi M. Multi-Objective Mapping for Mesh-Based NoC Architectures[C]// Proc of the 2nd IEEE/ACM/IFIP Int'l Conf on Hardware/Software Codesign and System Synthesis, 2004: 182-187.

[5] Lei T, Kumar S. A Two-Step Genetic Algorithm for Mapping Task Graphs to a Network on Chip Architecture[C]// Proc of the Euromicro Symp on Digital Systems Design, 2003: 180-187.

[6] 周干民，尹勇生，胡永华，等. 基于蚁群优化算法的 NoC 映射[J]. 计算机工程与应用，2005，41(18)：7-11.

[7] 杨盛光，李丽，高明伦，等. 面向能耗和延时的 NoC 映射方法[J]. 电子学报，2008，36(5)：937-942.

[8] 常政威，谢晓娜，桑楠，等. 片上网络映射问题的改进禁忌搜索算法[J]. 计算机辅助设计与图形学学报，2008，20(2)：155-160.

[9] LI Guangshun, WU Junhua, MA Guangsheng. Mapping of Irregular IP onto NoC Architecture with Optimal Energy Consumption[J]. Tsinghua Science and Technology, 2007, 12(1): 146-149.

[10] 岳培培，刘建. NoC 映射问题中的列举路径分配算法[J]. 电子科技大学学报，2008，37(1)：54-57.

[11] 肖晓强，姜玉琴，金士尧，等. BWR——带缓冲的虫孔路由技术[J]. 计算机学报，2001，24(1)：78-83.

[12] Garey M R, Johnson D S. Computers and Intractability: A Guide to the Theory of NP-completeness[M]. Freeman and Company, 1979.

[13] Dick R P, Rhodes D L, Wolf W. TGFF: Task Graphs for Free[C]// Proc of the 6th Int'l Workshop on Hardware/Software Co-design, 1998: 97-101.

[14] The Network Simulator - NS-2[CP/OL]. [2008-10-30]. http://isi.edu/nsnam/ns/.