doi:10.11887/j.cn.201605009 http://journal.nudt.edu.cn

CacheFI:基于架构级故障注入的片上缓存容错评估工具*

黄智濒[1,2],周 锋[1],马华东[1],何若愚[1]
(1. 智能通信软件与多媒体北京市重点实验室 北京邮电大学 计算机学院,北京 100876;
2. 软件开发环境国家重点实验室,北京航空航天大学 计算机学院,北京 100191)

摘 要:体系架构级缓存容错技术被认为是应对较高的永久位故障率的有效手段,但目前缓存容错机制的体系架构级评估工具较少。针对这个问题,提出CacheFI,即基于Simics的缓存故障注入工具,采用故障生成和注入分离的设计,故障生成是随机分布、模式和时序三个方面的结合,故障注入则考虑了故障可重现性和模块化的需要。在全系统模拟器Simics上,基于15个选自SPEC CPU2000的测试程序,利用CacheFI对Buddy和MAEP等典型的体系架构级缓存容错机制进行评估,展现了其弱点和典型的片上缓存容错机制存在的问题。

关键词:故障注入;片上缓存;体系结构模拟器;Simics
中图分类号:TP316 **文献标志码**:A **文章编号**:1001-2486(2016)05-052-07

CacheFI: micro-architectural-level fault injection based fault-tolerant evaluation tool for on-chip Caches

HUANG Zhibin[1,2], *ZHOU Feng*[1], *MA Huadong*[1], *HE Ruoyu*[1]
(1. School of Computer Science and Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China;
2. State Key Laboratory of Software Development Environment, School of Computer Science and Engineering, Beihang University, Beijing 100191, China)

Abstract: Architectural solutions of on-chip Cache fault tolerance are considered as effective means for high persistent failure probabilities. However, less fault injection tools for on-chip Caches are available. Therefore, the CacheFI, a fault injection tool based on the full-system simulator Simics was proposed. A separate mechanism of the fault generation and injection was employed. Fault generation consists of stochastic distribution control, failure patterns and explosive timing. Fault injection was designed to focus on the requirement of repeatability and modularity. According to the experiments based on Simics and 15 benchmarks from SPEC CPU2000, it evaluates typical micro-architectural fault mechanisms, such as Buddy, MAEP(matching access and error pattern) etc. by injecting Cache faults with CacheFI. Consequently, it presents the weakness and issues of these typical mechanisms.

Key words: fault injection; on-chip Cache; micro-architectural simulators; Simics

随着丹纳德微缩的终结,处理器芯片的功耗问题成为关键的制约因素之一。降低供电电压和动态的电压缩放是芯片的低功耗设计中的重要手段。在低电压下,工艺参数偏差会更加严重,晶体管更加脆弱和不可靠。芯片上的大型存储结构,特别是一级缓存、二级缓存等,更易受晶体管的工艺偏差和老化磨损影响,且其电压往往决定了整个芯片的供电电压。因此,缓解片上缓存对供电电压的限制,成为目前芯片功耗优化设计研究中的关键问题之一。

静态随机存储器(Static Random Access Memory, SRAM)存储阵列的位故障主要可以分为两大类:永久故障和非永久故障。永久故障与供电电压具有强关联性,据测算,每降低500 mV,永久故障率将有多于10亿倍的增加[1],出厂前简单的禁用排除所有永久错误将会导致芯片成品率过低。非永久故障主要包括软错误和不确定位故障。同样降低500 mV,软错误率的增长率只有2.5~3倍,要远小于永久故障率的增速。不确定位故障表现出与永久故障率具有一定的比例关系[1],最高可达1∶10。而且在低电压下,特别是电压降低到650 mV之后,永久故障和不确定故

* **收稿日期**:2015-05-30
基金项目:中国博士后基金资助项目(2014M550662);软件开发环境国家重点实验室资助项目(SKLSDE-2014KF-04);智能通信软件与多媒体北京市重点实验室资助项目(ITSM201303)
作者简介:黄智濒(1978—),男,湖北汉川市人,讲师,博士,E-mail:huangzb@bupt.edu.cn

障将成为主要故障源,将超越 10^{-3} 的警戒值,达到高故障率水平。片上缓存的容错机制和技术需要应对这种故障源的变化和故障率的水平变化。

因此,片上缓存容错机制的研究成为目前高效能关键任务处理器芯片研究的重要内容之一。国内外的研究人员提出了大量的缓存容错技术,大致可以分为四大类:①电路级的容错技术,主要采用多电压域或可靠性更高的存储单元电路[2],这类方法会增加面积开销和成本,会额外带来大量的设计、制造和测试工作,而且其供电电压的降低幅度有限。②编码级的容错技术,它是应对软错误的有效手段,被广泛地应用到目前的现代微处理器中。例如,Power8 处理器在 L1 缓存中使用了奇偶校验码[3],但在处理永久故障和不确定位故障时,编码方案效率较低,编码解码引起的延迟开销大。③体系结构级的缓存容错技术,主要通过修改缓存的操作路径或逻辑结构实现位故障的修补或者容忍。④混合容错技术,即上述三类方法的综合利用。

体系结构级的缓存容错技术目前研究比较活跃,而且被认为是应对高永久故障率的有效手段之一。这些容错机制往往依赖于典型的体系结构级的模拟器评估其性能,如 Simics[4] 和 Gem5 等,但是,由于缺少缓存故障注入工具,其可靠性和有效性的评估验证并不充分。针对这个问题,本文提出了 CacheFI,它是一种基于全系统模拟器 Simics 的故障注入架构,支持对片上缓存的各种容错机制和技术的评估以及压力测试。

1　故障注入的相关研究

故障注入是一种评估容错机制的有效方法,最初由 IBM 实验室在 20 世纪 70 年代提出,故障注入依据注入方式和对象大致可以分为三类:基于物理硬件的故障注入,基于软件实现的故障注入和基于仿真模拟的故障注入。

基于物理硬件实现的故障注入是指直接将故障注入硬件中,主要通过管脚或者辐射方式将故障注入目标系统。硬件故障注入控制性差,灵活性差,容易损坏系统。而且它只能在系统设计后期,原型系统成形之后实施。

基于软件实现的故障注入主要通过修改内存或者寄存器的值,模拟硬件故障对应用软件的影响。软件故障注入通常利用编译器,运行时通过环境或者二进制翻译技术[5] 实现。主要用于评估操作系统和应用软件的可靠性和容错能力,它的主要弱点是故障注入位置受限,对应用透明的结构难以覆盖,例如片上缓存。为了缓解故障注入位置受限问题,一些基于固件的故障注入方案被提出,例如秦磊等[6] 提出针对 IA64 架构的故障注入方案,利用处理器抽象层接口获取 CPU 硬件的信息,并注入故障到 CPU 中,这类方法本质上也属于软件故障注入技术。软件故障注入技术不能评估 CPU 内部硬件故障源自身的容错机制。

基于仿真模拟的故障注入依据仿真模拟的目标层次,可以分为基于硬件描述语言的仿真器故障注入和体系架构级模拟器的故障注入。前者主要是利用典型的硬件描述语言从行为模式或者结构模式仿真具体的硬件,基于寄存器级或门级实现。这类仿真器与具体的硬件模型密切联系,虽然高度可控,且模拟精度高,但仿真编译合成耗时长,而且由于硬件描述语言的限制,故障注入的覆盖范围和场景复杂度受限,对片上缓存故障注入的支持较少,主要集中在寄存器、总线、端口和内存等。为了加速仿真器的模拟过程,有些工具通过 FPGA 加速,例如 FuSE[7]。

后者主要依赖体系架构级模拟器。体系架构模拟器是体系结构研究的重要支撑工具,也是系统早期研究的重要辅助。体系架构级模拟器的故障注入成为目前故障注入研究的新热点,对评估体系结构的脆弱因子(Architecture Vulnerability Factor, AVF)和体系结构级的容错机制具有重要作用。胡倩等[8] 提出了基于 Simics 的处理器典型流水部件的故障注入,主要针对译码单元、地址生成单元、算术逻辑单元和寄存器堆。FSFI[9] 是一个基于开源全系统模拟器的故障注入工具,主要针对算术逻辑单元、解码器、寄存器堆和地址生成单元等。FIMSIM[10] 是一个基于 M5 模拟器的故障注入工具,主要针对寄存器堆、算术逻辑单元、传输后备缓冲器、绕开逻辑等。目前国内外基于体系结构模拟器的故障注入主要集中在对执行部件相关的组件上,据我们的广泛调研,片上缓存的故障注入工具尚不多见。

事实上,片上缓存的故障注入与寄存器或执行部件的故障注入有明显的不同,而且片上缓存的位故障率更高,对处理器芯片的成品率影响更大,片上缓存由 SRAM 存储单元构成,而且这些存储单元组合成了不同的逻辑单元,如标记域、数据域、缓存行、缓存组、缓存体以及复杂的缓存逻辑结构。这些层次的逻辑单元出现故障均会导致缓存不能正常工作。本文提出的 CacheFI 充分考虑了缓存各个层次逻辑单元的故障和位故障分布。

2 CacheFI 的架构

CacheFI 是基于全系统模拟器 Simics 的片上缓存故障注入工具,充分利用了 Simics 的 C 语言应用开发接口,充分考虑了缓存各个层次的逻辑单元,支持多种故障模式和多种故障分布,而且高度模块化,可以实现对各种缓存结构容错机制的评估验证和压力测试。图 1 展示了 CacheFI 的整体结构。主要由三部分组成:①基于 XML 格式的配置参数文件;②CacheFI-Gen,故障生成器;③CacheFI-Load,故障注入器。

基于 XML 格式的配置参数文件主要由五类参数构成,即:CPU 核配置、缓存架构配置、控制参数、故障模式配置和位故障水平参数,后续模块需要综合考虑这些参数。故障模式配置主要考虑故障注入的目的,CacheFI 主要用来进行故障容错机制的验证和故障容错能力的压力测试。依据目的不同,生成故障的覆盖范围和对缓存各层次逻辑结构的故障分布考虑有所不同。位故障水平参数主要依据不同的 SRAM 存储单元结构(如传统的 6T,或者 7T/14T 结构,8T 或者 10T 等)在不同的电压水平下的各种故障模式的故障率。这些数据可以依据国际半导体技术蓝图(International Technology Roadmap for Semiconductors, ITRS)的相关参数和对 SRAM 存储单元的故障模型[1,11]获得,也可以由用户直接输入指定。

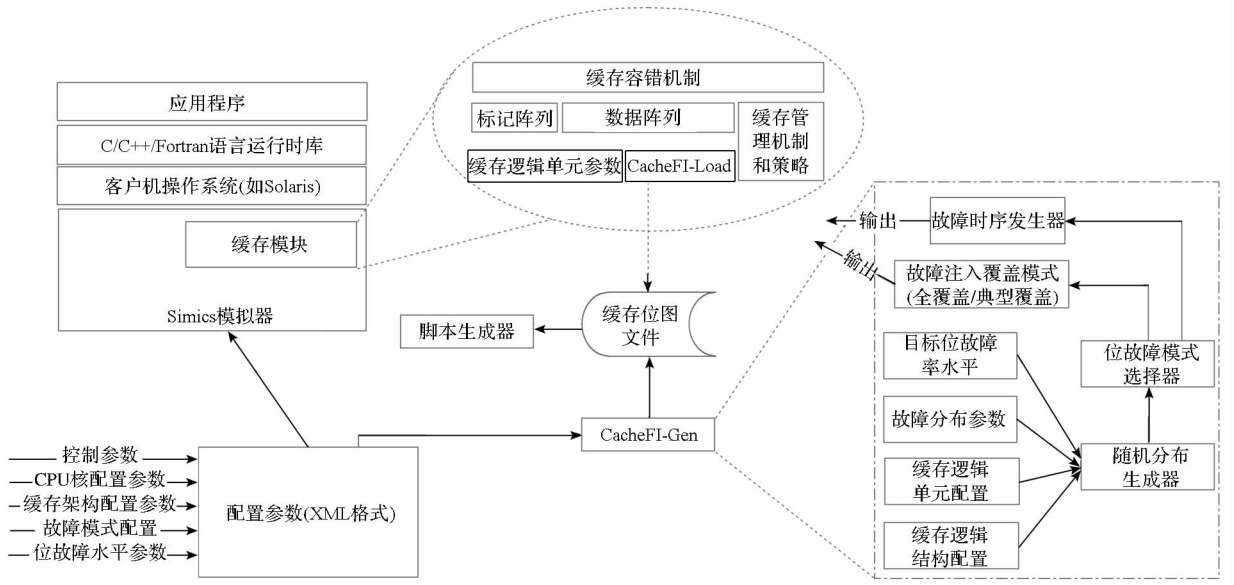

图 1 CacheFI 片上缓存故障注入工具的架构图

Fig. 1 Architecture of CacheFI

CPU 核配置、缓存架构配置和控制参数主要用来生成 Simics 全系统模拟器的运行脚本。

CacheFI-Gen 由三部分组成:随机分布生成器、位故障模式生成器和故障时序发生器。

随机分布生成器主要考虑设定的目标位故障率水平,依据故障分布参数,同时要考虑缓存的逻辑结构配置和各级缓存逻辑单元(如缓存子块、缓存行、标记域和数据域、缓存组以及其他与逻辑结构相关的单元结构等),生成故障分布。依据故障注入的目的不同,随机分布生成故障的方式也不同。如果故障注入是为了验证故障容错机制,随机分布生成依据泊松分布合理分布所有的位故障,以合理评估容错能力。如果故障注入为了故障容错能力的压力测试,则以故障覆盖为主,衍生各种极端情况的故障分布,这主要依据故障注入覆盖模式参数决定。CacheFI-Gen 通过全局表 G-Table 记录缓存各个逻辑结构和逻辑单元的位故障分布情况。

随机分布生成器以缓存组为基础,通过组编码 SID,缓存行内字编码 WID,以及字内位偏移 BID,生成故障分布。SID,WID 和 BID 由随机数函数生成,每次位故障设定前,通过查询全局表 G-Table,依据故障位所属的各级逻辑结构和逻辑单元,确定该故障位是否成立。

位故障模式生成器主要设定每一个故障位的故障发生模式,SRAM 存储单元的故障主要分为永久故障和非永久故障,非永久故障又可以分为软错误和不确定位故障。永久故障是在特定电压下,位故障确定持续存在。非永久故障也与电压密切相关,其发生具有随机性和不确定性,但故障发生之后会持续存在,直到电压升高,该位的故障状态才会发生变化。软错误是瞬时错误,如果某

位偶然发生软错误,称为瞬时故障,如果某位频繁发生软错误,则称为间歇故障。因此,将位故障分成四类:永久故障、不确定位故障、瞬时故障和间歇故障。另一方面,SRAM 存储单元的故障可以分成四类:读故障、写故障、保持故障和访问故障[11]。其中,访问故障是时序故障,由延迟引起。其他三种故障是数据故障。数据故障进一步可以分成三种类型:恒 0,恒 1,翻转。由于体系结构模拟器难以有效地模拟缓存访问流水线及其时序,因此,将访问故障模式用数据故障代替,这并不影响 CacheFI 的故障注入效果和对缓存容错机制验证的支持。总之,每一个位故障的模式包括故障发生的时间属性和值属性,具体见表 1。

表 1　故障模式的分类

Tab. 1　Classification of failure patterns

分类名称	故障时间属性	故障的值属性
永久故障	一定电压下固定发生,持续	恒 0,恒 1,翻转
不确定位故障	一定电压下随机发生,持续	恒 0,恒 1,翻转
瞬时故障	随机发生一次,瞬时	恒 0,恒 1,翻转
间歇故障	随机发生多次,瞬时	恒 0,恒 1,翻转

由于除永久故障之外的其他三类故障的发生时机都具有随机性,瞬时故障和间歇故障持续时间较短,而且间歇故障会多次随机发生,因此在 CacheFI-Gen 中专门设置了故障时序发生器。在电压频率发生改变之后,以机器周期为基本的时间单元(基于 Simics 的 API 函数 SIM_cycle_count),通过生成随机数,设定故障发生的时刻。对于瞬时故障,还需要设定故障持续的时间,如果故障持续存在的这段时间,有缓存请求访问该位,则会出现故障,否则,该位虽然出现故障,但并没有衍生成错误。

对于间歇故障会随机发生多次,每次持续时间瞬时且随机。2 次间歇故障发生时刻的时间间隔服从均匀分布、指数分布和 Weibull 分布。这些分布属性会存储到故障位图文件中。

CacheFI-Gen 会将位故障信息生成故障位图文件。该文件采用结构化的行列结构存储,每一行描述一个位故障。通过故障位图文件注入故障的好处之一是可以在基于 Simics 进行性能评估时,所有的测试程序基于同一故障分布,减少故障分布差异导致的性能差异。另一个好处是有助于实验过程的复现,进而有助于缓存容错机制的验证。

CacheFI-Load 是内置于 Simics 缓存模块的机制,主要用于读取故障位图文件,并解析位故障属性,设定对应的数据阵列和标记阵列的故障。CacheFI-Load 主要由文件解析模块、分布函数生成模块以及位故障注入模块构成。CacheFI-Load 以 Simics 的 G-cache 模块为基础,深度修改了其中的 struct cache_line 结构和 struct generic_cache 结构,并对缓存访问过程进行了全面的修改,主要包括标记域比对函数 lookup_line()、缓存访问处理函数 gc_operate()、读访问函数 handle_read() 和写访问函数 handle_write() 以及缓存管理策略的初始化函数 update_config() 等。

通过将故障的生成过程和故障实际注入过程分离,CacheFI 不仅有效地实现了各种故障模式的生成,并通过有效的保存故障位图来支持实验的可重现性,而且可以有效地支持各级缓存逻辑单元和结构的故障注入,充分验证缓存容错机制。

3　实验评估

基于时钟周期的全系统模拟器 Simics[4],模拟了一个 16 核的多核处理器。每一个处理器核基于 UltraSPARC IV + 的配置,具体参数见表 2。模拟器的操作系统是 Solaris10,主要的测试程序选自 SPEC CPU2000。

表 2　实验环境的基准配置

Tab. 2　Baseline configuration

项	属性值
处理器核	16 核,每核均为 1 个硬件线程,且顺序执行;时钟频率 1.5 GHz;每核配置有私有的 L1 数据缓存和 L1 指令缓存,容量均为 32 KB,缓存行大小为 64 字节,均为 4 路,命中延迟为 3 个周期,采用 LRU 替换算法管理
共享的 L2 缓存	容量 2 M,缓存行大小为 64 字节;8 路组相连,命中延迟为 11 个周期,采用 LRU 替换算法管理。采用 MESI 缓存一致性协议
内存	访问延迟为 200 个周期
测试程序	164.gzip,168.wupwise,171.swim,172.mgrid,173.applu,175.vpr,177.mesa,179.art,181.mcf,183.equake,188.ammp,197.parser,200.sixtrack,255.vortex,256.bzip2

CacheFI 集成到 Simics 模拟器环境中,并基

于 G-cache 模块，实现了 ZerehCache[12] 和 Archipelago[13]，Buddy[14]（由于 Salvage[15] 缓存机制与 Buddy 类似，因此没有选作后续实验），Macho[16] 以及 MAEP（Matching Access and Error Patterns）[17] 缓存容错机制。

ZerehCache[12] 利用图染色算法充分利用冗余行列，Archipelago[13] 利用一种自适应的最小团覆盖算法，试图最小化牺牲缓存行，但是，这两种方法都是 NP 难问题，难以找到精确优化解，只能寻求近似算法 IBSC（Incomplete Backtracking Sequential Coloring）和 DSATUR 求解。Buddy[14]，Salvage[15] 和 Macho[16] 是基于缓存组级的无冲突子块（chunk）修补的缓存容错机制。MAEP[17] 是容忍位故障的方案，通过调整地址映射，使得请求的数据能放置到无故障子块中。

为了测试 CacheFI，主要做两方面的实验：①CacheFI对缓存容错架构的验证能力，并对典型的缓存容错架构容错能力的分析，指出这些容错架构的问题；②CacheFI 与 Simics 集成环境下，这些缓存容错架构的性能评估。

进行实验评估时，设置了三种永久位故障率（*pFail*）的场景，即永久位故障率分别为 0.001，0.002 和 0.004，依据文献[1]的位故障模型与供电电压 *Vcc* 的关系，*Vcc* 分别对应为 470 mV，450 mV和 420 mV。这与文献[1]和文献[17]的实验场景类似。不确定位故障率与永久位故障率设定比例关系，按高比例关系，设定为 1∶10，即每发生 10 个永久位故障会发生 1 个不确定位故障[1]。瞬时故障率为 10^{-6}，为了简化实验过程，假设瞬时故障率保持不变。间歇故障与瞬时故障率一致。依据这些参数，CacheFI 实现故障的生成和注入。

首先，对缓存架构的容错能力进行实验验证。当位故障率较高（大于 0.002）时，ZerehCache 的 Group 以及 Archipelago 的自治岛（islands）的会显著降低，牺牲缓存行的数量明显增多。当永久位故障率为 0.004 时，ZerehCache-Group 的平均值约为 12.4 个缓存行，Archipelago-Islands 的大小平均约为 16.7 个缓存行。而且还存在一些被禁用的缓存行，ZerehCache 的禁用比例约 23.6%，Archipelago 的禁用比例约为 14.9%。

通过实验发现，基于缓存组级的无冲突子块（chunk）修补的缓存容错架构，如 Buddy，Salvage 和 Macho 存在较多的问题，特别是缓存组关联度较小的时候。下面以 Buddy 缓存容错机制的实验进行说明。

通过 CacheFI 注入故障，构建 100 种不同的故障分布的缓存，并利用 Buddy 容错方案修复的缓存故障（缓存组的设定关联度为 4）。针对这些情况，对每个缓存中所有缓存组的实际关联度的统计发现，在 *pFail* = 0.001 时，只有 1 个可用完好缓存行的缓存组的比例为 1.16%，拥有 2 个缓存行的缓存组占比为 23.77%，拥有 3 个缓存行的缓存组占比为 63.12%，所有缓存行均完好的缓存组的比例为 11.95%。没有出现组内缓存行均被禁用的故障缓存组，所有缓存组关联度的几何平均值为 2.86，如图 2 所示。当 *pFail* = 0.002 时，这些对应的缓存组的比例分别为 9.44%，56.80%，31.98%，1.36% 和 0.42%。也就是说，存在 0.42% 的缓存组，其组内所有的缓存行均被禁用，这些缓存组是故障缓存组，包含这些故障缓存组的缓存并不能正常工作。所有缓存组关联度的几何平均值为 2.24。当 *pFail* = 0.004 时，这些对应的缓存组的比分别例 5.78%，42.43%，49.27%，2.51% 和 0.01%。故障缓存组的比例显著增加到 5.78%。所有缓存组关联度的几何平均值为 1.49。

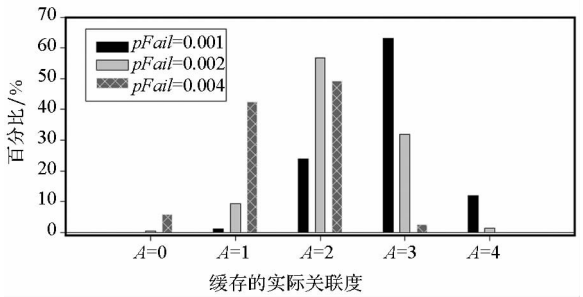

图 2 Buddy 缓存中缓存组实际缓存行数量的分布情况

Fig. 2 Practical Associativity of Cache sets in Buddy

图 3 展示了不同位故障率水平，统计 100 种不同的故障分布时，基于 Buddy 修补机制能正常工作的缓存的比例。从图 3 中可以看出，当 *pFail* = 0.001 时，所有的 Buddy 缓存均能正常工作，当 *pFail* = 0.002 时，只有 48.28% 的 Buddy 缓存可以正常工作，当 *pFail* = 0.004 时，所有的 Buddy 缓存均

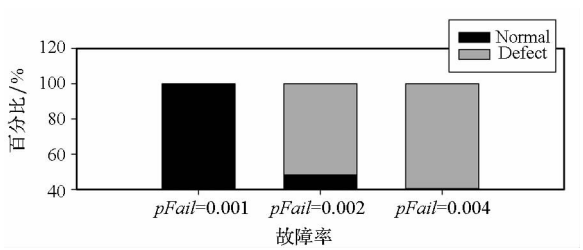

图 3 100 种不同的故障分布时正常的 Buddy 缓存比例

Fig. 3 Ratio of normal Buddy Cache for 100 failure distributions

不能正常工作,每个缓存中均包含故障缓存组。需要处理这些故障缓存组,Buddy 缓存才能正常工作。

Buddy 架构中出现的这些问题,Salvage 和 Macho 架构中也同时存在,特别是故障缓存组的问题。通过基于 CacheFI 故障注入的评估,发现这些缓存容错机制存在明显的缺陷和问题。

其次,基于故障注入的 Simics 模拟器环境,评估这些缓存容错架构的性能,主要挑选了无故障的基于最近最久未使用(Least Recently Used, LRU)算法管理的缓存,Buddy 缓存和 MAEP 缓存三种架构,并应用到 L1 数据缓存和 L1 指令缓存进行性能评估,如图 4 和图 5 所示。由于 Buddy 缓存在位故障率大于 0.002 时,会出现故障缓存组,因此,在性能评估时,使用的是修改的 Buddy 架构,将所有对故障缓存组的缓存请求映射到相邻的缓存组。

当使用 CacheFI 注入故障时,Buddy 和 MAEP 主要处理永久故障,其他的位故障都依赖纠错码检错和纠错。这些故障会增加一定的访问延迟。从图 4 和图 5 中可以看出,缓存容错机制由于需要处理位故障,导致几乎所有的测试程序的缺失率都有明显的增加,而且当位故障率越大时,缺失率增加越多。Buddy 修补位故障机制的性能要明显优于 MAEP 容忍位故障的机制。在 15 个测试程序中,当 $pFail = 0.001$ 时,L1 数据缓存中,除了 188.ammp 外,在 Buddy 中其他 14 个测试程序的缺失率都要低于 MAEP 缓存的情况。当 $pFail = 0.002$ 时,除了 177.mesa,179.art 和 188.ammp 之外,其他 12 个测试程序在 Buddy 中表现更优。

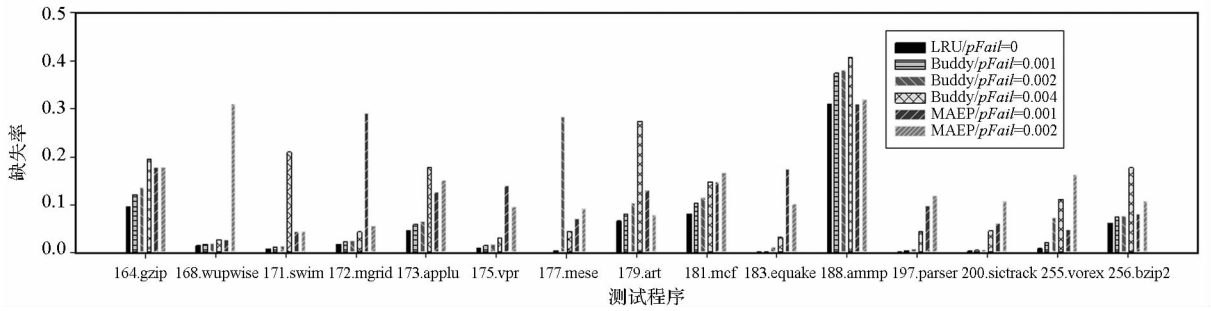

图 4　不同缓存容错架构的 L1 数据缓存的缺失率

Fig. 4　Miss rates of different L1 data Cache architectures

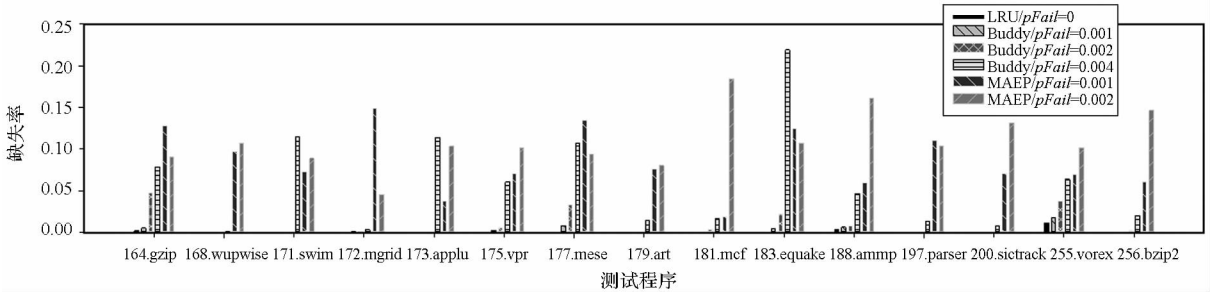

图 5　不同缓存容错架构的 L1 指令缓存的缺失率

Fig. 5　Miss rates of different L1 instruction Cache architectures

4　结论

在低电压下,特别是电压降低到 650 mV 之后,永久故障和不确定故障将成为主要故障源,将超越 10^{-3} 的警戒值,达到高故障率水平。片上缓存的故障是影响芯片成品率的主要原因之一,而且体系架构级的缓存容错技术被认为是应对永久位故障率较高的有效手段。但是体系结构级的片上缓存容错机制的验证和测试工具尚不多见,影响了对体系架构级的片上缓存容错机制的有效验证和评估。针对这个问题,本文提出了 CacheFI,它是一个基于全系统模拟器 Simics 的片上缓存的故障注入工具,可以有效地支持对片上缓存的各种容错机制和技术的评估以及压力测试。

参考文献(References)

[1] Chishti Z, Alameldeen A R, Wilkerson C, et al. Improving Cache lifetime reliability at ultralow voltages [C]// Proceedings of the 42th Annual IEEE/ACM International Symposium on Micro-architecture, 2009: 89 - 99.

[2] Kulkarni J P, Kim K, Roy K. A 160 mV robust schmitt trigger

based subthreshold SRAM[J]. IEEE Journal of Solid-State Circuits, 2007, 42(10): 2303 – 2313.

[3] Starke W J, Stuecheli J, Daly D M, et al. The Cache and memory subsystems of the IBM POWER8 processor[J]. IBM Journal of Research and Development, 2015, 59(1): 1 – 13.

[4] Magnusson P S, Christensson M, Eskilson J, et al. Simics: a full system simulator platform[J]. Computer, 2002, 35(2): 50 – 58.

[5] Li D, Vetter J S, Yu W. Classifying soft error vulnerabilities in extreme-scale scientific applications using a binary instrumentation tool[C]// Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC), 2012: 1 – 11.

[6] 秦磊, 庞东贺, 张展, 等. IA64 固件层处理器故障注入技术[C]//第六届中国测试学术会议论文集, 2010.
QIN Lei, PANG Donghe, ZHANG Zhan, et al. IA64 firmware implemented processor fault injection technique [C]// Proceedings of 6th China Test Conference, 2010. (in Chinese)

[7] Jeitler M, Delvai M, Reich S. FuSE-A hardware accelerated HDL fault injection tool [C]// Proceedings of Southern Conference on Programmable Logic, 2009: 89 – 94.

[8] 胡倩, 王超, 王海霞, 等. 基于 Simics 的系统级故障注入平台[J]. 计算机工程, 2015, 41(2): 57 – 62,75.
HU Qian, WANG Chao, WANG Haixia, et al. Simics-based system level fault injection platform [J]. Computer Engineering, 2015, 41(2): 57 – 62,75. (in Chinese)

[9] Chao W, Fu Z C, Chen H S, et al. FSFI: a full system simulator-based fault injection tool[C]//Proceedings of the 1st International Conference on Instrumentation, Measurement, Computer, Communication and Control (IMCCC'11), 2011: 326 – 329.

[10] Yalcin G, Unsal O S, Cristal A, et al. FIMSIM: a fault injection infrastructure for micro-architectural simulators [C]//Proceedings of the IEEE 29th International Conference on Computer Design (ICCD), 2011: 431 – 432.

[11] Mukhopadhyay S, Mahmoodi H, Roy K. Modeling of failure probability and statistical design of SRAM array for yield enhancement in nanoscaled CMOS[J]. IEEE Transaction on Computer-Aided Design Integrated Circuits, 2005, 24(12): 1859 – 1880.

[12] Ansari A, Gupta S, Feng S G, et al. Zerehcache: armoring Cache architectures in high defect density technologies[C]// Proceedings of the 42th Annual IEEE/ACM International Symposium on Micro-architecture (MICRO – 42), 2009: 100 – 110.

[13] Ansari A, Feng S G, Gupta S, et al. Archipelago: a polymorphic Cache design for enabling robust near-threshold operation[C]// Proceedings of the IEEE 17th International Symposium on High Performance Computer Architecture (HPCA – 17), 2011: 539 – 550.

[14] Koh C K, Wong W F, Chen Y R, et al. Tolerating process variations in large set associative Caches: the buddy Cache[J]. ACM Transactions on Architecture and Code Optimization (TACO), 2009, 6(2): 116 – 123.

[15] Koh C K, Wong W F, Chen Y R, et al. The salvage Cache: a fault-tolerant Cache architecture for next-generation memory technologies[C]// Proceedings of the IEEE International Conference on Computer Design, 2009: 268 – 274.

[16] Mahmood T, Hong S, Kim S. Ensuring Cache reliability and energy scaling at near-threshold voltage with Macho[J]. IEEE Transactions on Computers, 2015, 64(6): 1694 – 1706.

[17] Choi Y, Yoo S, Lee S, et al. MAEPER: matching access and error patterns with error-free resource for low Vcc L1 Cache[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2013, 21(6): 1013 – 1026.