近阈值电压下可容错的末级缓存结构设计

刘 伟*①② 魏志刚① 杜 薇①② 曹广义① 王 伟③
①(武汉理工大学计算机科学与技术学院 武汉 430070)
②(交通物联网技术湖北省重点实验室 武汉 430070)
③(同济大学计算机科学与技术系 上海 200092)

摘 要: 近阈值电压技术通过降低晶体管的电源电压来降低芯片能耗和提升能效。但是,近阈值电压技术会在 Cache 中引起大量位错误,严重影响末级缓存的功能。针对近阈值电压下超过 1%的位错误率造成的 Cache 故障问题,该文提出一种基于传统 6T SRAM 单元的可容错的末级缓存结构(FTLLC)。该策略对缓存条目中的错误进行了低错纠正和多错压缩,提高了 Cache 中数据保存的可靠性。为了验证 FTLLC 的有效性,该文在 gem5 中实现了该结构,并运行了 SPEC CPU2006 测试集进行仿真实验。结果表明,对于 650 mV 电压下 65 nm 工艺的末级缓存,FTLLC 与 Concertina 压缩机制相比在 4-Byte 粒度下末级缓存可用容量增加了 24.9%,性能提高了 7.2%,末级缓存的访存缺失率下降了 58.2%,而面积和能耗开销仅有少量增加。

关键词: 近阈值电压;容错 Cache;纠错码;压缩机制
中图分类号: TP302.8 **文献标识码:** A **文章编号:** 1009-5896(2018)07-1759-08
DOI: 10.11999/JEIT170989

Fault-tolerant Last Level Cache Architecture Design at Near-threshold Voltage

LIU Wei①② WEI Zhigang① DU Wei①② CAO Guangyi① WANG Wei③
①(*College of Computer Science and Technology*, *Wuhan University of Technology*, *Wuhan* 430070, *China*)
②(*Hubei Key Laboratory of Transportation Internet of Things*, *Wuhan* 430070, *China*)
③(*Department of Computer Science and Technology*, *Tongji University*, *Shanghai* 200092, *China*)

Abstract: Near-threshold voltage computing enables transistor voltage scaling to continue with Moore's Law projection and dramatically improves power and energy efficiency. However, a great number of bit-cell errors occur in large SRAM structures, such as Last-Level Cache (LLC). A Fault-Tolerant LLC (FTLLC) design with conventional 6T SRAM cells is proposed to deal with a higher failure rate which is more than 1% at near-threshold voltage. FTLLC improves the reliability of data stored in Cache by correcting the single-error and compressing multi-errors in Cache entry. To validate the efficiency of FTLLC, FTLLC and prior works are implemented in gem5, and are simulated with SPEC CPU2006. The experiment shows that compared with Concertina at 650 mV, the performance of a 65 nm FTLLC with 4-Byte subblock size improves by 7.2% and the Cache capacity increases by 24.9%. Besides, the miss rate decreases by 58.2%, and there are little increases on area overhead and power consumption.

Key words: Near-threshold voltage; Fault-tolerant Cache; Error correction code; Compression mechanism

收稿日期:2017-10-23;改回日期:2018-04-03;网络出版:2018-05-10
*通信作者:刘伟 wliu@whut.edu.cn
基金项目:国家自然科学基金(61672384),教育部人文社科项目(16YJCZH014),湖北省自然科学基金(2016CFB466),中央高校基本科研业务费(WUT: 2016III028, 2017III028-005) , 湖北省技术创新专项重大项目(2017AAA122)
Foundation Items: The National Natural Science Foundation of China (61672384), The Ministry of Education of Humanities and Social Science project (16YJCZH014), The Natural Science Foundation of Hubei Province (2016CFB466), The Fundamental Research Funds for the Central Universities (WUT: 2016III028, 2017III028-005), Major Program of Technical Innovation Special Program in Hubei Province of China (2017AAA122)

1 引言

随着半导体工艺的提高，芯片上可集成更多的晶体管，其数量按照摩尔定律每 18 个月翻一番[1]。因此，一块芯片理论上能够集成更多的晶体管电路。但是，能量和功耗限制很大程度上阻碍了更多晶体管的正常使用，导致芯片性能难以继续提升[2]。近阈值电压技术将电源电压降低到晶体管阈值电压附近，能够大幅降低功耗和提高能效[3]，使芯片性能不再受到散热能力的约束，进一步提升了芯片性能。

近阈值电压下，由于工艺参数波动等原因，末级缓存中存在大量不能正确存取数据的 SRAM 单元。这些存在故障的单元可根据错误分为两类：硬错误和软错误。硬错误由 RDF(Random Dopant Fluctuation)引起，可以通过 BIST(Build-In Self Test)程序检测确定并且在确定电压下不会发生变化。而软错误仅在运行过程中产生，无法检测确定。由于软错误发生概率比硬错误低 5 个量级[4]，在近阈值电压下可以被忽略。

近年来，近阈值电压下 Cache 的可靠性问题受到广泛关注。文献[4]基于拉丁正交方码(OLSC)提出多位纠错的 MS-ECC 策略，文献[5]使用简单的纠错码保护缓存行，而文献[1,6]结合两种不同复杂度的纠错码技术提出 VS-ECC 策略和改进策略。这类基于纠错码技术的策略能够保证末级缓存的可靠性，但同时要求高昂的面积开销和延迟代价。文献[7]将比特位重映射排列，使用简单的纠错码技术实现多位错误纠正。文献[8]将保存到有错缓存条目的数据映射到无错的条目中，以充分利用末级缓存中可用的容量。文献[9]提出一种压缩机制，将非空的数据映射到缓存条目中无错的部分。这类使用映射方法的策略实现了一定的可靠性，但是在 Cache 容量和性能上有较大的牺牲。文献[10,11]通过利用计算机存储结构中天然存在的数据冗余来保证 Cache 的可靠性，这类方法带来了明显的性能损失。此外，文献[12–15]针对近阈值电压提出了不同的 SRAM 单元设计以提升 Cache 的可靠性，不过这类 SRAM 单元拥有较大的面积开销和延迟代价。

为了保证 Cache 在近阈值电压下的可靠性，本文提出一种可容错的末级缓存结构(FTLLC)。该结构采用纠错保护机制对缓存条目进行细粒度保护，使用压缩机制对数据进行压缩，保证数据正确存入末级缓存。本文针对纠错保护机制提出了部分保护策略，对纠错码技术带来的面积开销和性能代价做一定的折中。

2 FTLLC 设计与实现

本文提出了一种结合纠错码技术和压缩机制的可容错末级缓存设计，旨在保证 Cache 在较高失效率下可靠运行的同时提高缓存性能。由于前人基于纠错码技术的策略带来了较大延迟和面积代价，本文提出简单低延迟的纠错机制来纠正缓存行中的一些位错误，并设计一种保护策略以降低纠错策略实现所带来的空间代价。同时，本文对存入末级缓存的数据块进行压缩，将其保存在无错的位置，最大化 Cache 的可用容量。压缩模块和纠错模块的融合协作，极大程度降低了 SRAM 单元失效对 Cache 可靠性和性能损失的影响。

2.1 整体结构与工作流程

容错的末级缓存设计结构如图 1 所示，整个结构包括末级缓存模块、压缩模块和纠正模块。图 1 中的 LLC Tag 和 LLC Data 是末级缓存模块，压缩模块由空子块探测器、子块压缩/解压器、错误映射表(UFM)和压缩映射表(CM)这 5 部分组成，编/解码器和 ECC 阵列共同构成纠正模块。图中粗线部分表示 FTLLC 访存过程的关键路径，该路径上增加的组件会导致末级缓存访存延迟增加。

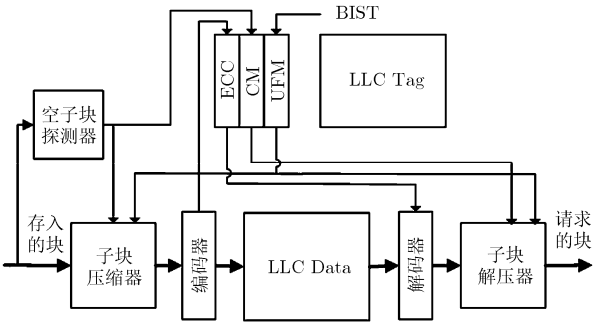

图 1 FTLLC 设计结构

如图 1 所示，对末级缓存进行数据写入操作时，FTLLC 分为 3 步操作。

第 1 步 空子块探测器分析传入的数据块，确定空子块的位置，生成表示空子块信息的二进制串，并将结果传递至 CM 和子块压缩器。

第 2 步 结合从空子块探测器得到的空子块信息和 UFM 中对应缓存条目的子条目错误信息，子块压缩器对数据块进行压缩，将非空的子块重新映射到可纠正的子条目中。

第 3 步 编码器依据本文提出的保护策略对压缩后的数据使用纠错码技术进行编码，生成的校验信息存入 ECC 阵列，再将数据存入对应的缓存条目中。

对末级缓存进行读操作时，FTLLC 同样需要经过两个模块的处理。首先，FTLLC 根据请求的地址通过 Tag 信息确定数据所在的缓存条目，将对应的

数据和 ECC 校验信息取出。然后，解码器对缓存条目中的数据正确性进行校验，若有错则进行纠正。最后，子块解压器通过对应的错误信息和数据压缩信息对数据进行解压，将‘0’填入被压缩的位置，最终得到完整的数据块。

2.2 压缩模块设计

在普通程序中，天然存在全部由二进制‘0’填充的数据区域，且大量分布于程序各处。例如 SPEC CPU2006 基准测试程序集[16]中各测试程序在 1-Byte, 2-Byte 和 4-Byte 粒度下都存在较高比例的空数据子块[9]，这就为压缩机制的使用提供了可能。然而，在具有较高位单元失效率的末级缓存中，缓存条目对存入的数据块要求较高的空子块比例，大量低空子块比例的数据块将带来较大的性能损失。因此，本文基于 Concertina[9]提出优化的压缩模块设计，以结合纠正模块应用。

压缩模块中的核心组件是错误映射表 UFM 和压缩映射表 CM。缓存条目中存在无法被 ECC 纠正的子条目，UFM 中保存着这些子条目位置信息，若有错子条目无法纠正，则置对应的位为‘0’，否则置为‘1’。CM 则保存着数据块的压缩信息，若一个数据子块是空子块，则用‘0’标识 CM 中对应的位，否则以‘1’标识。

如图 2 所示，以 4-Byte 大小的粒度为例，则 CM 和 UFM 中每个数据块或缓存条目对应的信息用 16 bit 位来保存，每比特位表示对应的 4-Byte 子块/子条目的状态。CM 中值为‘1’的位所对应的子块数据需要保存在 UFM 中值为‘1’的位所代表的子条目内，这个过程通过子块压缩器完成。本文的子块压缩/解压器采用 Concertina[9]中的子块压缩和重放置逻辑实现。该组件具有较低的延迟代价和极低的面积开销，相比于 UFM 和 CM 所引入的面积开销，子块压缩/解压器的实现开销可以忽略。

UFM 中数据由 BIST 生成，当芯片通电启动时，BIST 会自动检测末级缓存[7,17]，对 UFM 进行初始化，初始化的过程见表 1 的算法 1。首先，BIST 初始化 UFM 所有比特位为‘0’，再检测缓存条目中

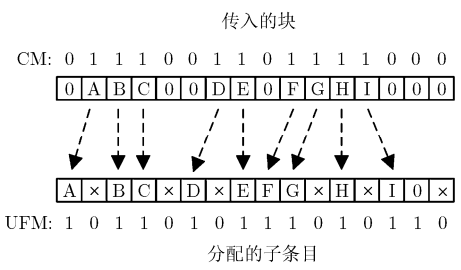

图 2 压缩数据块重放置到有错缓存条目的过程示例

表 1　UFM 初始化算法

算法 1　UFM 初始化算法
输入：空 UFM 阵列，末级缓存 LLC 阵列
输出：配置好的 UFM 阵列
(1) **for** each CacheEntry in LLC **do** //对每个缓存条目分析
(2)　**for** each subentry in CacheEntry **do**
(3)　　初始化 UFM 中每一位为‘0’;
(4)　　**if** subentry 没有单元故障 **then**
(5)　　UFM 中对应子条目状态位←1;
(6)　　**end if**
(7)　**end for**
(8)　初始化 ECC 阵列;
(9)　**for** each subentry in CacheEntry **do**
(10)　　**if** subentry 存在单元故障，且可被 ECC 纠正 **then**
(11)　　更新 UFM 中状态位为‘1’;
(12)　　**end if**
(13)　**end for**
(14) **end for**
(15) **return** UFM

每个子条目中是否存在单元故障，如果某个子条目无错，则置该子条目在 UFM 中所对应的状态位为‘1’。然后，BIST 将对 ECC 阵列进行初始化，得到各子条目的纠正信息，具体过程见表 2 的算法 2。最后，BIST 根据 ECC 阵列中的信息判断有错的子条目是否能够被 ECC 纠正，如果能够被纠正，则置对应状态位为‘1’。UFM 的初始化过程结束后，

表 2　ECC 阵列初始化算法

算法 2　ECC 阵列初始化算法
输入：空的 ECC array，缓存条目
输出：ECC array
(1) **for** each ECCentry in ECCarray **do**
(2)　Initialize ECCentry with ‘0’; //初始化 ECC 阵列
(3)　**for** each Unit in ECCentry **do**//检测纠错单元是否有错
(4)　　**if** Unit 中存在单元故障 **then**
(5)　　设置 Unit 无效;
(6)　　**end if**
(7)　**end for**
(8)　**for** each Unit in ECCentry **do** //对有错子条目进行保护
(9)　　**if** 缓存条目中存在可被 ECC 纠正的子条目 **then**
(10)　　获得可纠正的子条目;
(11)　　**if** Unit 有效并未被使用 **then**
(12)　　位置索引←子条目位置;
(13)　　设置 Unit 已使用;
(14)　　**end if**
(15)　**end if**
(16)　**end for**
(17) **end for**
(18) **return** ECCarray

UFM 中的状态信息不再需要更新，不会为 Cache 的读写操作带来额外开销。

CM 中的数据来自于空子块探测器，随着芯片工作的过程实时传入更新。探测器对要写入末级缓存的数据中每一子块进行分析，置空子块对应的标志位为‘0’。最终探测器生成一串代表空子块信息的二进制串，该串的长度与压缩粒度大小有关，图 2 中可看到 4-Byte 粒度下数据块和 CM 串的对应结果。

为了保证存储阵列的可靠性，本文采用 ST SRAM 单元[12]实现 UFM 和 CM。ST SRAM 单元相比 6T SRAM 会带来 100%的额外面积增加以及 60%的延迟代价。本文通过 CACTI[18]分析后发现，UFM 和 CM 使用 ST SRAM 不会影响末级缓存整体的访问延迟。UFM 和 CM 造成的面积增加将在 4.4 节分析。

2.3 纠正模块实现

前人采用的纠错码技术是基于 BCH 码设计的，例如 SECDED, DECTED 以及 4EC5ED 等。SECDED 仅能纠正数据块中的 1 个位错误，但是编/解码只需要 1 周期。4EC5ED 能够纠正 4 个错误，不过该技术具有较高的延迟代价，需要至少 7 个周期解码[1]。本文设置与压缩粒度一致的纠错粒度，采用 SECDED 技术对指定粒度下每个子条目进行保护，以此获得较低的延迟代价和较高的可靠性。然而，SECDED 技术会带来较大的面积开销。因此，本文提出了最优保护策略(FTLLC-OPT-k)，即对条目中最适合纠正的 k 个子条目进行纠正，使保护的子条目数达到最优。同时，本文提出完全保护策略(FTLLC-FULL)，以反映本文的纠正模块所能达到的最佳性能。

2.3.1 最优保护策略 对于拥有 M 个子条目的缓存条目，本文使用纠错码保护其中 $k(k<M)$个子条目，当 k 增加时，可以保护的子条目数量也随之增加。例如，在 4-Byte 粒度下缓存条目有 16 个子条目，假设$k=4$，那么对应的 ECC 阵列如图 3(a)所示。纠错标志表示用于保护的 4 个纠错单元是否已使用。每个纠错单元由有效位、位置索引和校验信息构成。有效位表示该纠错单元中所有的比特位是否是无错的，位置索引表示被保护的子条目在缓存条目中的位置，校验信息表示编码器生成的校验码信息，其长度与 SECDED 保护的粒度大小有关。4-Byte 粒度的子条目需要 7 个比特位，因此，FTLLC-OPT-4 的 ECC 阵列使用 52 个比特位保护一个缓存条目。若要纠正 k 个子条目，则每个缓存条目需要 $13k$ 个比特位保存信息。通过分析失效模型，本文设定 k 值范围在 4 到 7 之间。为了保证 ECC 阵列中数据的可靠性，本文使用 ST SRAM 单元存储这些重要数据。

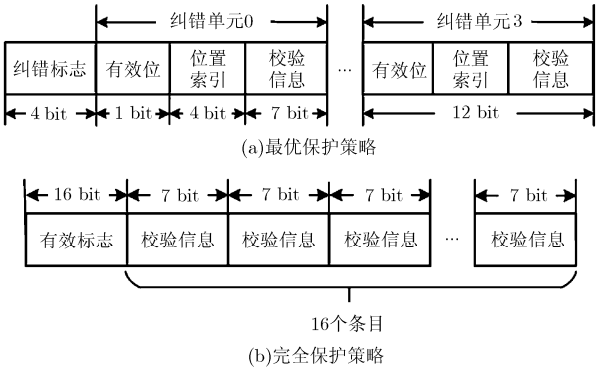

图 3 最优保护策略和完全保护策略的 ECC 阵列结构

ECC 阵列内容的生成分为两个阶段：启动阶段和运行阶段。启动阶段，即 BIST 检测阶段，该阶段过程可见算法 2。首先，BIST 将 ECC 阵列中所有比特位初始化为‘0’。然后，BIST 对 ECC 阵列中的 SRAM 单元进行检测，若纠错单元中没有 SRAM 单元故障，那么该纠错单元对应的有效位将被置‘1’，否则置为‘0’。最后，BIST 对每个子条目中有错的单元进行检测统计，将有错且可被纠正的子条目的位置存入未使用的纠错单元中，然后设置该纠错单元已使用。当每一个 Cache Set 中的子条目都已被检测或所有纠错单元都已使用，则 BIST 检测完成。运行阶段中，每次有数据将要被写入末级缓存中时，压缩后的数据通过编码器生成校验信息，根据 ECC 阵列中保护的子条目，将对应部分的校验信息保存。

2.3.2 完全保护策略 完全保护策略对缓存条目中每个子条目都进行保护，该方式可以达到保护策略的最佳性能效果，但是会引入极高的面积开销。本文以 4-Byte 大小的纠错粒度为例，其 ECC 阵列的结构如图 3(b)所示。从图 3(b)可以看到，ECC 阵列由 2 部分构成，一部分为有效标志，另一部分是校验信息。校验信息表示对应子条目通过编码器生成的校验码信息，16 个子条目就有 16 块校验信息区域。有效标志表示 16 个校验信息条目是否是正确的，同时使用 ST SRAM 实现 ECC 阵列以提高可靠性。因此，4-Byte 粒度下的 ECC 阵列需要额外 128 比特位来保护一个缓存条目，粒度越细，额外要求的开销越大。

2.4 替换策略

由于压缩技术会将传入的数据压缩为一个长度不定的数据块，而同一 Cache Set 中各缓存条目中

可容纳的数据大小也不一致，于是传统的 LRU 替换策略将产生大量数据大小和缓存条目容量不匹配的问题。为了解决该问题，本文使用一种基于 LRU 改进的管理策略 —— Fit-LRU[9]。该策略在选择缓存条目时，会结合每个缓存条目的可用容量和压缩后的数据大小进行判断，先找出能够容纳所传入数据的缓存条目，再从其中选出 LRU 的条目作为最终选择的缓存条目。

在这个过程中，很有可能出现一种情况：传入数据对应的 Cache Set 中没有能够容纳该数据块的缓存条目，即所有缓存条目能够存储的数据大小都小于所传入数据块压缩后的数据大小。此时，FTLLC 直接对该数据进行写回处理，将其保存到内存中，以保证数据的可靠性。

3　评估方法

本文提出的 FTLLC 采用 gem5 模拟器进行实验[19]，模拟的系统架构基本参数如表 3。CPU 的电源电压设定为 650 mV，该电压下 65 nm SRAM 的位错误率为 1.1%[2]，故障单元相互独立且随机分布于 Cache 中[1,4]。本文针对末级缓存提出容错设计，因此假设一级缓存在近阈值电压下是鲁棒的。该实验架构中二级缓存即为末级缓存，LLC Data 使用传统的 6T SRAM 单元，而 LLC Tag 使用 ST SRAM[12]以保证可靠性。

表 3 模拟架构的参数

架构参数	内含
处理核	8 核，In-Order，单线程，1 GHz@650 mV
一级缓存	32 kB×2, 4 路组相联，私有，访问延迟 2 周期，LRU
二级缓存(末级缓存)	1 MB, 16 路组相联，共享 Cache，访问延迟 8 周期，压缩/解压阶段额外增加 1 周期延迟，码/解码阶段额外增加 1 周期延迟，Fit-LRU
一致性协议	MOESI, Full-map，基于目录协议
内存	2 GB×2，访问延迟 50 ns
片上网络	Mesh 网络，链路延迟 1 周期

本实验选择 SPEC CPU2006 作为测试集[16]，从测试集中随机选择测试程序，每选择 8 个测试程序组成一组程序集，最终随机生成 10 组程序集。对于每组程序集，本文指定系统运行 10×10^8 条指令，其中前 3×10^8 条指令用于 Cache 中数据填充，作数据收集准备，后 7×10^8 条指令用于收集运行数据。

本文提出的 FTLLC-OPT-k 策略性能与保护的子条目数目 k 有关，本文设定 k 值范围 $k\in\{4,5,6,7\}$。实验对不同 k 值的 Cache 结构进行比较，同时选取以下两种策略设计的 Cache 结构与 FTLLC 进行比较：

(1)Concertina[9]。该结构基于压缩机制设计，在压缩/解压阶段会造成 1 周期的时间延迟代价。

(2)SECDED+Disabled[5]。该结构以整个缓存条目为纠错粒度，使用 SECDED 技术对 Cache 进行保护，同时将错误较多的缓存条目无效化。由于原架构仅适用于低位错误率，本文调整以更细的粒度进行保护。

此外，本文引入了理想的 Cache，即假设传统的 Cache 结构在近阈值电压下保持高可靠性。

实验将从 Cache 容量可用率、末级缓存访存缺失率、性能、面积开销和功耗 5 个方面进行比较，同时以理想的 Cache 作为基准，将实验结果进行标准化处理。

4　结果与分析

4.1 Cache 容量可用率

在近阈值电压下，传统末级缓存的可用容量接近于 0%，而可靠的末级缓存结构设计能够大幅提高 Cache 的可用容量。本文的末级缓存容量可用率 ($\mathrm{CAR_{LLC}}$) 的计算公式如式(1)，式中 Size_{ij} 表示缓存条目 i 的子条目 j 的大小，F_{ij} 表示对应子条目是否有错，若无错则 $F_{ij}=1$，整个公式表示末级缓存中所有可用容量占总容量的比值。

$$\mathrm{CAR_{LLC}}=\frac{\sum_{i=0}^{n\mathrm{Entry}}\sum_{j=0}^{n\mathrm{Subentry}}\mathrm{Size}_{ij}\times F_{ij}}{\sum_{i=0}^{n\mathrm{Entry}}\sum_{j=0}^{n\mathrm{Subentry}}\mathrm{Size}_{ij}} \tag{1}$$

从图 4 可以看出细粒度的块保护策略能够有效地提高 Cache 的可用容量。在 4-Byte 粒度下，Concertina 策略的 Cache 容量可用率达到了 70%，而 SECDED+Disabled 策略仅有 46.7%。FTLLC-OPT-k 策略随着 k 值的增加，其容量可用率也随之提高，相比 Concertina 策略至少增加了 20.9%。此外，FTLLC-OPT-7 的容量可用率与 FTLLC-FULL 极为接近。不同情况下，FTLLC 的可用率都高于 90%，表明了本文提出的 FTLLC 对于 Cache 容量恢复具有极好的效果。

4.2 访存缺失率分析

Cache 访存缺失率是衡量 Cache 性能的重要指标，其计算方法是访存缺失数与总访存请求数的比值。图 5 展示了不同策略的末级缓存在不同粒度下的缺失率对比，SECDED+Disabled 策略的缺失率增加了 5%到 21%, Concertina 策略在各粒度下都拥有最高的缺失率。FTLLC-OPT-k 策略的缺失率增加不超过 9.4%，且随着保护数 k 增加，逐渐趋近于

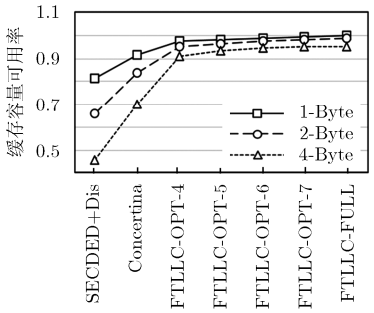

图 4 不同的末级缓存在不同粒度下的容量可用率

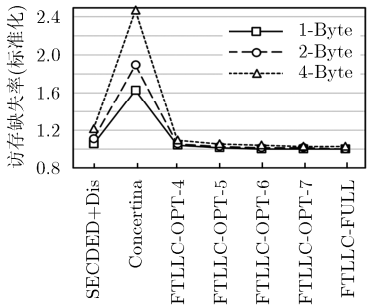

图 5 不同的末级缓存在不同粒度下的缺失率

理想 Cache 的缺失率，明显优于其它的策略。同时，作为对比的 FTLLC-FULL 策略的缺失率仅比 FTLLC-OPT-7 低 0.33%，表明本文所选取 k 值范围下最优保护策略的效果与该策略最佳效果极为接近。

4.3 性能分析

本文使用 IPC(Instructions Per Cycle)作为衡量处理器性能的指标，分析不同的末级缓存设计对 CPU 性能的影响。图 6 中展示了不同策略的末级缓存设计在不同粒度下的性能表现，随着压缩和纠错的粒度减小，末级缓存的性能衰减越小。SECDED+Disabled 策略在不同粒度下的性能衰减在 1%到 2.6%之间，而 Concertina 具有明显的性能衰减。FTLLC-OPT-k 最高仅有 1.4%的性能衰减，相对于 Concertina 策略在 4-Byte 粒度下拥有 6.7%~7.2%的性能提升。与 FTLLC-OPT-7 相比，FTLLC-FULL 策略最高仅有 0.05%的性能提升，这表明部分保护策略在性能上与完全保护策略接近。

4.4 面积开销分析

本文使用 CACTI[18]分析各种情况下 FTLLC 以及其它策略的面积开销，面积开销以传统的 6T SRAM 单元构成的末级缓存结构为基准进行标准化。从图 7 中可以看到，Concertina 在各粒度下都具有最低的面积开销。FTLLC-OPT-k 的面积开销随着保护数 k 的增加而增加，面积增加最低达到 27.2%，最高可达 70.54%，但依然低于同粒度下 SECDED+Disabled 的面积，而 FTLLC-FULL 却有 51.5%~165.8%的面积增加。FTLLC-OPT-k 拥有与 FTLLC-FULL 接近的性能，却只有较低的面积开销，表明了 FTLLC 在性能和面积开销上达到了良好的均衡。

4.5 能耗分析

本文采用 McPAT[20]来评估本文策略的能耗，该能耗同时包括末级缓存能耗和片外内存的访问能耗[9,17]。本文以正常电压下的传统末级缓存能耗为基准，将近阈值电压下各策略的能耗进行标准化。从图 8 中可以看到，近阈值电压下末级缓存的能耗仅有正常电源电压下的 23%~36%。在同一粒度下，随着 FTLLC-OPT-k 的保护数 k 从 4 增加到 7，能耗仅有少量增加。而 SECDED+Disabled 和 FTLLC-FULL 策略具有较高的能耗，这是由于大量增加的辅助电路带来的能耗增加。Concertina 策略拥有最小的额外电路增加和面积开销，但是由于较高的末级缓存访问缺失率带来了较高地内存访问能耗，从而呈现出略高于 FTLLC-OPT-k 的能耗。与理想的 Cache 相比，FTLLC 的能耗最低仅提升了 3%，低于其它策略在同等条件下的能耗开销。

综合前面所有指标，Concertina 虽然拥有最低的面积开销，但其性能的衰减也是最大的，同时也带来了较大能耗开销。而 SECDED+Disabled 虽然

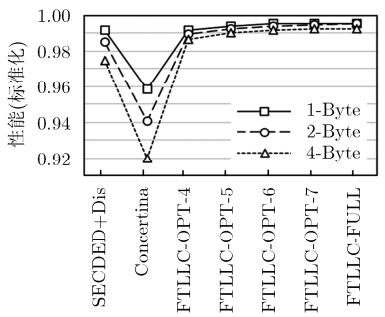

图 6 不同的末级缓存在不同粒度下的性能

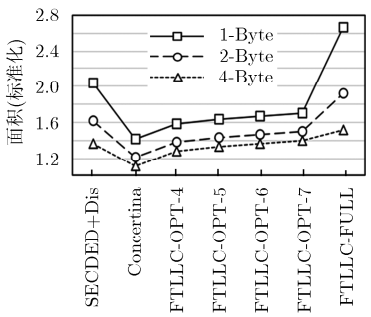

图 7 不同的末级缓存在不同粒度下的面积开销

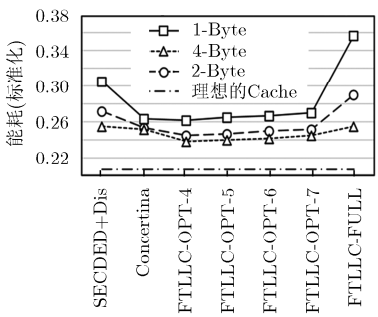

图 8 不同的末级缓存在不同粒度下的能耗开销

有较低的性能衰减，但是造成了巨大的面积和能耗开销。FTLLC 虽然在面积开销上也有所增加，但是拥有最低的能耗增加，同时在性能上具有较好的表现。与其它策略相比，本文的 FTLLC 能够以较低的面积增加实现较高的可靠性、较好的性能提升和最小的能耗增加。

5　结论

本文主要研究近阈值电压下具有较高错误率的末级缓存所面临的可靠性和性能问题，提出了一种可容错的末级缓存设计 —— FTLLC。FTLLC 使用压缩策略对传入末级缓存中的数据进行压缩，减少了对缓存条目中无错子条目的需求，同时使用纠错码技术对有错的缓存子条目进行纠错保护，增加了缓存条目中可用的子条目数量。本文提出了最优保护策略 FTLLC-OPT-k，对部分可纠正的子条目进行保护，在保证性能的基础上降低了面积开销。相比于压缩策略 Concertina, FTLLC 可以实现最低 6.7%的性能提升，以及最低仅 16.8%的面积增加，同时在提高 Cache 容量可用率和降低访存缺失率上具有明显的提升效果。FTLLC 与其它的策略相比拥有较好的能耗控制并实现了较高的可靠性和性能提升。

参 考 文 献

[1] ALAMELDEEN A R, WAGNER I, CHISHTI Z, *et al*. Energy-efficient cache design using variable-strength error-correcting codes[C]. Proceedings of the 38th Annual International Symposium on Computer Architecture, New York, 2011: 461–472. doi: 10.1145/2000064.2000118.

[2] DRESLINSKI R G, WIECKOWSKI M, BLAAUW D, *et al*. Near-threshold computing: Reclaiming Moore's Law through energy efficient integrated circuits[J]. *Proceedings of the IEEE*, 2010, 98(2): 253–266. doi: 10.1109/JPROC.2009.2034764.

[3] 张永欢, 姜岩峰. 近阈值电压电路研究进展[J]. 微电子学, 2016, 46(1): 107–112. doi: 10.13911/j.cnki.1004-3365.2016.01.024.
ZHANG Yonghuan and JIANG Yanfeng. Research progress of near threshold voltage circuits[J]. *Microelectronics*, 2016, 46(1): 107–112. doi: 10.13911/j.cnki.1004-3365.2016.01.024.

[4] CHISHTI Z, ALAMELDEEN A R, WILKERSON C, *et al*. Improving cache lifetime reliability at ultra-low voltages[C]. Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture, New York, 2009: 89–99. doi: 10.1145/1669112.1669126.

[5] HIJAZ F, SHI Qingchuan, and KHAN O. A private level-1 cache architecture to exploit the latency and capacity tradeoffs in multicores operating at near-threshold voltages [C]. IEEE 31st International Conference on Computer Design, Asheville, 2013: 85–92. doi: 10.1109/ICCD.2013.6657029.

[6] 赵彩, 丁永林, 陈志坚. 基于混合纠错码的可容错性高速缓存研究[J]. 计算机应用研究, 2016, 33(2): 444–446. doi: 10.3969/j.issn.1001-3695.2016.02.029.
ZHAO Cai, DING Yonglin, and CHEN Zhijian. Fault-tolerance cache research based on mixed ECC[J]. *Application Research of Computers*, 2016, 33(2): 444–446. doi: 10.3969/j.issn.1001-3695.2016.02.029.

[7] DUWE H, JIAN Xun, PETRISKO D, *et al*. Rescuing uncorrectable fault patterns in on-chip memories through error pattern transformation[C]. Proceedings of the 43rd International Symposium on Computer Architecture, Seoul, 2016: 634–644. doi: 10.1109/ISCA.2016.61.

[8] WANG Jing, LIU Yanjun, ZHANG Weigong, *et al*. Exploring variation-aware fault-tolerant cache under near-threshold computing[C]. 45th International Conference on Parallel Processing, Philadelphia, 2016: 149–158. doi: 10.1109/ICPP.2016.24.

[9] FERRERÓN A, SUÁREZ-GRACIA D, ALASTRUEY-BENEDÉ J, *et al*. Concertina: Squeezing in cache content to operate at near-threshold voltage[J]. *IEEE Transactions on Computers*, 2016, 65(3): 755–769. doi: 10.1109/TC.2015.2479585.

[10] WANG Ying, HAN Yinhe, LI Huawei, *et al*. VANUCA: Enabling near-threshold voltage operation in large-capacity cache[J]. *IEEE Transactions on Very Large Scale Integration Systems*, 2016, 24(3): 858–870. doi: 10.1109/TVLSI.2015.2424440.

[11] JUNG D, LEE H, and KIM S W. Lowering minimum supply voltage for power-efficient cache design by exploiting data redundancy[J]. *ACM Transactions on Design Automation of Electronic Systems*, 2015, 21(1): 1–24. doi: 10.1145/2795229.

[12] CALHOUN B H and CHANDRAKASAN A P. A 256-kb 65-nm sub-threshold SRAM design for ultra-low-voltage operation[J]. *IEEE Journal of Solid-State Circuits*, 2007, 42(3): 680–688. doi: 10.1109/JSSC.2006.891726.

[13] 杨堃. 近阈值低功耗 SRAM 研究设计[D]. [硕士论文], 上海交通大学, 2011.
YANG Kun. Low power SRAM research and design under near-threshold voltage supply[D]. [Master dissertation], Shanghai Jiao Tong University, 2011.

[14] 齐蓓蓓. 近阈值绝热静态存储器设计[D]. [硕士论文], 宁波大学, 2015.
QI Beibei. The design of near-threshold adiabatic SRAM[D]. [Master dissertation], Ningbo University, 2015.

[15] 于雨情, 王天琦, 齐春华, 等. 65 nm 近阈值 SRAM 稳定性分析[J]. 微电子学与计算机, 2017, 34(1): 26–29. doi: 10.19304/j.cnki.issn1000-7180.2017.01.006.

YU Yuqing, WANG Tianqi, QI Chunhua, *et al*. The analysis of the stability of 65nm SRAM at near-threshold region[J]. *Microelectronics & Computer*, 2017, 34(1): 26–29. doi: 10.19304/j.cnki.issn1000-7180.2017.01.006.

[16] HENNING J L. SPEC CPU2006 benchmark descriptions[J]. *ACM SIGARCH Computer Architecture News*, 2006, 34(4): 1–17. doi: 10.1145/1186736.1186737.

[17] DUWE H, JIAN Xun, and KUMAR R. Correction prediction: Reducing error correction latency for on-chip memories[C]. IEEE 21st International Symposium on High Performance Computer Architecture, California, 2015: 463–475. doi: 10.1109/HPCA.2015.7056055.

[18] MURALIMANOHAR N, BALASUBRAMONIAN R, and JOUPPI N P. Optimizing NUCA organizations and wiring alternatives for large Caches with CACTI 6.0[C]. Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture, Chicago, 2007: 3–14. doi: 10.1109/MICRO.2007.33.

[19] BINKERT N, BECKMANN B, BLACK G, *et al*. The gem5 simulator[J]. *ACM SIGARCH Computer Architecture News*, 2011, 39(2): 1–7. doi: 10.1145/2024716.2024718.

[20] LI Sheng, AHN J H, STRONG R D, *et al*. McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures[C]. Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture, New York, 2010: 469–480. doi: 10.1145/1669112.1669172.

刘 伟： 男，1978 年生，博士，副教授，主要研究方向为低功耗系统结构、云计算与服务计算.

魏志刚： 男，1993 年生，硕士生，研究方向为低功耗系统结构、近阈值计算.

杜 薇： 女，1978 年生，博士，副教授，主要研究方向为绿色计算、云计算与服务计算.

曹广义： 男，1984 年生，博士，讲师，主要研究方向为云计算、绿色计算.

王 伟： 男，1979 年生，博士，副教授，主要研究方向为绿色计算、云计算.