

基于 BCH 算法的高速缓存纠检错方案研究

王珍珍¹, 杨志涛², 王 晶^{1,3}, 张伟功^{1,4}

(1 首都师范大学 信息工程学院 北京 100048; 2 唐山现代工控技术有限公司, 河北 唐山 063000;

3 北京成像技术高精尖创新中心, 北京 100048; 4 电子系统可靠性技术北京市重点实验室, 北京 100048)

摘 要: 针对抗辐射微处理器的容错需求, 特别是在高速缓存中由单粒子效应造成的多位翻转故障, 设计了一种高速缓存纠检错方案, 以 LEON2 处理器为研究平台, 分别对指令 Cache 和数据 Cache 的标记存储器和数据存储器设计并行的 BCH 编码器和译码器, 实现多达 4 位错误的检测。针对检测到的多位错误, 利用存储层次特征, 实现对故障的有效规避。以 LEON2 处理器为研究平台的仿真结果证明, 该方案能够有效应对高速缓存中单粒子效应所引发的多位错误, 并且方案的资源开销和性能开销都比较小, 为未来纳米级微处理器的容错设计提供了支撑。

关键词: BCH 码; 单粒子效应; 并行; 高速缓存; 检错

中图分类号: TP302.8

文献标识码: A

文章编号: 1000-7180(2016)11-0124-05

Research of Cache Error Detection and Correction Based on BCH Algorithm

WANG Zhen-zhen¹, YANG Zhi-tao², WANG Jing^{1,3}, ZHANG Wei-gong^{1,4}

(1 College of Information Engineering, Capital Normal University, Beijing 100048, China;

2 Tangshan Modern Control Co., Ltd, Tangshan 063000, China

3 Beijing Advanced Innovation Center for Imaging Technology, Beijing 100048, China;

4 Beijing Key Laboratory of Electronic System Reliability and Prognostics, Beijing 100048, China)

Abstract: A solution of cache error detection and correction based on LEON2 processor was brought forward in this paper, to meet the fault tolerance requirements of radiation hardened microprocessor, especially for Multi-Bit Upset caused by single event effect in cache. Rather, parallel BCH (Bose-Chaudhuri-Hocquenghem) encoder and decoder were designed for tag memory and data memory of instruction cache and data cache, which could achieve error detection up to four bits. Also it could handle the faults effectively after checking out them. Experimental results based on LEON2 showed that this design could efficiently handle multi-bit error due to single event effect in cache, and its resource and performance overhead was acceptable, thus it would support for fault tolerance design of the future nanoscale microprocessor.

Key words: BCH codes; single event effect; parallel; cache; error detection

1 引言

单粒子效应^[1]是空间应用发生故障和工作异常的重要诱因之一。随着半导体器件特征尺寸的减小、工作频率的上升和节点电压的降低, 单个粒子引起

的故障表现形式更为复杂, 由此引发的多位翻转 (Multiple Bit Upset, MBU) 通常会导致 2-4 位错误^[2], 最多 8 位随机翻转错误。可见 MBU 比起 SEU 引发的单比特错误, 对空间应用的电子系统危害更大。

收稿日期: 2016-05-20; 修回日期: 2016-07-06

基金项目: 国家自然科学基金项目(61472260, 61402302, 61502321); 北京市属高等学校创新团队建设与教师职业发展计划项目(IDHT20150507)

空间环境中,规整的存储部件,尤其是高速缓存(Cache)极易受到单粒子效应的影响,引发危害严重的 SEU 和 MBU 故障,因此空间处理器必须对 Cache 进行容错加固设计. 现有方案大多只能处理 Cache 的 1-2 位错误,而对于 MBU 引发的 2-4 位错误缺乏行之有效的容错方案. 本文提出基于 BCH 算法的高速缓存纠错检错方案,能够处理 Cache(包括指令 Cache 和数据 Cache)的标记存储器(Tag memory)和数据存储器(Data memory)可能出现的 1-4 位错误,对故障进行有效规避.

2 相关工作

2.1 高速缓存的容错现状

目前国内外的 Cache 容错方案不能检测相邻或分散的 1-4 位错,为了缓解我国航天领域对国外技术的依赖,必须探索新的 Cache 容错机制. 本文提出的基于 BCH 算法的 Cache 纠错检错方案,可以有效解决上述问题. 与国内现有方案相比,它的创新之处在于能够检测出随机的 1-4 位错误,容错能力明显提高.

2.2 BCH 码概述

BCH 码能检测并纠正多位随机错误,并且容易根据纠错能力直接确定码的构造,是一类应用广泛的差错控制码^[3],在实际中应用最广泛的是码元取自 GF(2)的二进制 BCH 码.

对于任意的正整数 $m(m \geq 3)$ 和 $t(t \leq 2^m - 1)$,一定存在一组二进制 BCH 码,其参数满足表 1,根据实际需要确定以上各参数,即可找到对应的 BCH 码.

表 1 BCH 码参数表

码长	$n = 2^m - 1$
可纠正的最大错误位数	t
信息位个数	k
校验位个数	$n - k \leq mt$
最小汉明距离	$d_{min} = 2t + 1$

近年来也出现一些针对 FLASH 存储器设计的 BCH 编译码算法,主要是应用于 NAND FLASH 芯片^[4]校验纠错而研究的并行的 BCH 编译码器. 相比之前的串行结构,并行结构大大提高了 BCH 的编解码速度. 例如,我国北斗卫星导航系统^[5]就使用了并行的 BCH 解码器,能在一个周期内完成解码,但它只是针对一位错误. 相比之下,国外对 BCH 的并行化和如何实现快速的 BCH 编译码等研究更

加先进. 而在国内,特别是针对空间处理器高速缓存多位错误的容错设计方面,对 BCH 码的应用研究较为缺乏.

2.3 研究平台

本文以 SPARC V8 为平台,研究 BCH 编译码器对高速缓存的纠错检错. SPARC V8 是一种精简指令集(Reduced Instruction Set Computer, RISC)体系结构,与其他 RISC 处理器类似的是,同样在 CPU 和外部主存之间添加了高速缓存(Cache),能够有效解决处理器快速处理能力与主存慢速读写速度之间速度不匹配的问题. SPARC 采用流水线结构,将 Cache 结构分为指令 Cache 和数据 Cache 分别独立存储指令和数据,能有效地避免流水过程中取指和读写数据的冲突,提升指令并行执行能力,诸多优势令该体系结构被广泛应用. 而 LEON2 是一款 SPARC V8 架构的开源的处理器实现,方便本设计的添加和调试.

本设计选用基于 SPARC V8 的 LEON2 处理器,其中指令 Cache 和数据 Cache 均采用直接映像方式,存储容量都为 2 k 字,块大小是 4 个字,字宽度 32 位,标记存储器容量为 128 字,字宽度 25 位,区号标记 21 位,字有效标记(Valid)4 位. 针对单粒子引发的指令/数据 Cache 的标记/数据存储器的多位错误,采用本文所提出的并行 BCH 算法实现高效的纠错检错.

3 高速缓存检错设计方案

本文对 Cache 的检错方案包括了整体设计、Cache 的命中设计、BCH 编译码器的设计和在 Cache 中的实现,本节具体介绍这四个方面.

3.1 整体方案设计

在处理器中,Cache 主要包含三部分:Cache 控制器、标记存储器和数据存储器,如图 1 所示,容易发生 SEU 和 MBU 的部位是标记存储器和数据存储器,因此 Cache 的检错方案针对这两部分设计.

当主存向 Cache 标记存储器和数据存储器中存数据时,执行 BCH 编码过程. 当数据从 Cache 中读取送给处理器内核时,一方面 Cache 控制器根据标记存储器和数据存储器与主存的映像关系判断要访问的数据是否在 Cache 中以及是否有效,如果在 Cache 且有效,Cache 控制器会生成一个初步命中信号,即图 1 中的 hit0 为 1;另一方面,对标记存储器和数据存储器的数据执行 BCH 译码过程进行检错,根据 BCH 译码器生成的错误信号 t-err 和 d-err

检测是否发生错误. 如果没有错误, $t\text{-err}$ 和 $d\text{-err}$ 均为 0. 只有 $\text{hit}=1$, $t\text{-err}=0, d\text{-err}=0$ 同时满足的条件下, Cache 的命中信号 hit 有效, Cache 控制器直接将数据送给处理器内核; 任何一方不满足, Cache 都不命中, Cache 控制器从主存中取数据送给处理器内核, 从而实现错误的纠正.

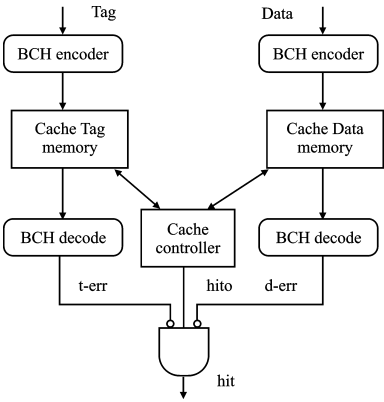


图 1 整体方案

3.2 高速缓存命中设计

Cache 中的标记存储器用主存块号来访问, 数据存储器由主存和块内地址来访问^[6]. 本设计在 Cache 的标记存储器和数据存储器中添加 BCH 校验位 (parity), 分别对区号和有效位以及数据进行 BCH 校验, 在 Cache 的命中判断过程中增加 BCH 的编码和译码过程. Cache 进行命中判断时, 有三个判断条件: 区号比较结果、字有效判别结果和 BCH 校验结果, 如图 2 所示.

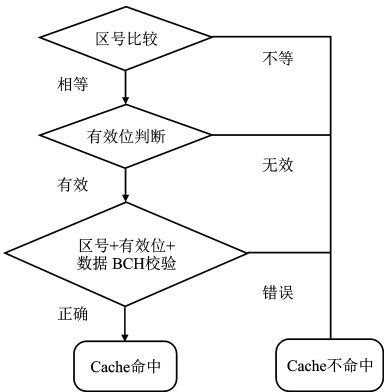


图 2 Cache 的命中条件

在判断 Cache 是否命中时, 首先从标记存储器中取得区号标记, 和主存地址区号进行比较, 如果两者相等, 则表示 Cache 中已经存在要访问的主存数据块, 然后用块内地址从标记存储器中取得字有效标记 Valid, 如果对应的 Valid 有效, 再看标记存储器中区和有效位的 BCH 校验结果, 以及数据存

储器中数据的 BCH 校验结果, 如果正确, 则此时 Cache 命中, 否则 Cache 不命中.

指令 Cache 和数据 Cache 的标记存储器和数据存储器都没有发生错误的前提下, Cache 才能命中, 处理器内核直接接收 Cache 数据存储器传来的数据; 任何一方出现 1-4 位错误, BCH 编译码器都能检测出来, 强制 Cache 不命中, 并且 Cache 控制器发出信号令处理器的流水线停滞, 申请访问主存储器, 从主存中重新取得数据. 这样就避免了 Cache 中出错的指令或数据流向处理器内核, 令系统异常甚至造成严重后果. 针对上述设计要求, 选用适合的 BCH 码, 对 Cache 实现 1-4 位错误检测.

3.3 BCH 编译码器的设计

BCH 码对数据的纠错检错过程包括编码和译码, 本小节主要介绍编码和译码的设计.

3.3.1 并行编码设计

在串行编码过程中, 一组码字从信息位输入到编好码字输出需要经过 n 个周期才能完成, 本文对 Cache 中的 32 位宽数据存储器 和 25 位宽标记存储器进行 BCH 编码, 如果采用串行结构会有很大延迟, 因此按照文献^[7]的方法采用并行编码.

确定 BCH 码的相关参数, 对 32 位宽 Data 和 25 位宽 Tag 实现 4 位检错, 可知本原多项式为

$$p(x) = x^6 + x + 1 \tag{1}$$

式中, 码长 $n = 63$, 纠错个数 $t = 4$, 校验位个数 $n - k = 24$, 信息位长度 $k = 39$, 最小距离 $d_{\min} = 9$. 进而确定生成多项式

$$g(x) = 1 + x + x^2 + x^4 + x^5 + x^6 + x^8 + x^9 + x^{10} + x^{13} + x^{16} + x^{17} + x^{19} + x^{20} + x^{22} + x^{23} + x^{24} \tag{2}$$

生成矩阵

$$\mathbf{G} =$$

$$\begin{bmatrix} g_{n-k} & g_{n-k-1} & \cdots & g_1 & g_0 & 0 & \cdots & 0 \\ 0 & g_{n-k} & \cdots & g_2 & g_1 & g_0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & g_{n-k} & g_{n-k-1} & \cdots & g_1 & g_0 \end{bmatrix} \tag{3}$$

得到 $[63, 39]$ BCH 码的生成矩阵 $\mathbf{G}_{39 \times 63}$. 而最终要编码的信息位是 32 位和 25 位, 因此对 $[63, 39]$ BCH 码进行缩短, 进而求得缩短后的生成矩阵 $\mathbf{G}_{32 \times 56}$ 和 $\mathbf{G}_{25 \times 49}$.

设 m 为信息位序列, 码字

$$C = m \cdot \mathbf{G}^{[7]} \tag{4}$$

即可得到 $[56, 32]$ BCH 码和 $[49, 25]$ BCH 码, 并行编码完成, 校验位宽为 24 位, 并且用组合逻辑在一

个周期内即可实现。

3.3.2 并行译码设计

完整的 BCH 译码算法包括计算伴随式,计算错误位置多项式,钱搜索确定错误位置。根据 Cache 的检错设计要求,只检测出 Cache 的 1-4 位错误,不需要纠正。根据伴随式就可以判断是否出错^[8]。伴随式为

$$S = R \cdot H^T$$
 (5)

式中, R 为接收码字, H 为校验矩阵。由前面求得的生成矩阵 G ,可以得到 $Data$ 的校验矩阵 $H_{24 \times 56}$ 和 Tag 的校验矩阵 $H_{24 \times 49}$ 。上述伴随式用 VHDL 语言,仅使用组合逻辑即可实现并行计算,得到对应的 S 矩阵如果为零矩阵,则接收的码字正确,Cache 中的 Tag 或 Data 没有发生错误;如果 S 矩阵为非零矩阵,则说明 Cache 中的 Tag 或 Data 发生了 1-4 位错误。

3.4 BCH 编译码器在高速缓存中的实现

Cache 中的数据都是来自于主存,因此 Cache 的内容只是主存部分内容的拷贝^[8]。在 LEON2 处理器中,Cache 控制器从主存中取得的数据,在 Cache 存储器中存放,在 Cache 命中状态,Cache 控制器直接从 Cache 存储器中取得数据送给处理器内核-整数单元 (IU),而 Cache 不命中时,IU 通过 Cache 控制器从主存中取得所需数据。Cache 存储器包括标记存储器和数据存储器,由前文可知,Cache 中原来的标记存储器为 25 位宽,经过 BCH 编码后增加 24 位校验位;Cache 中原来的数据存储器为 32 位宽,经过 BCH 编码后也增加 24 位校验位。以指令 Cache 为例,在标记存储器未发生错误的前提下,数据存储器的 BCH 编译码器的实现方式如图 3 所示。

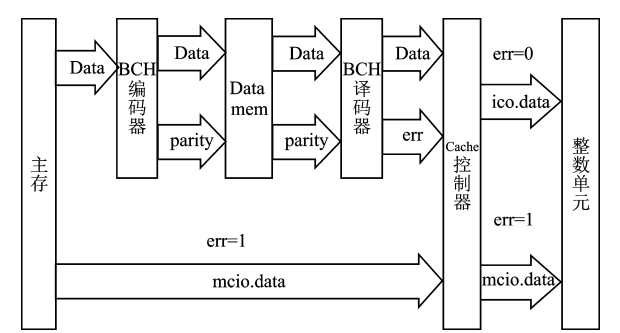


图 3 BCH 编译码器的实现

对于来自主存储器的 Data 输入 BCH 编码器,经过 BCH 编码后,Data 和 parity 作为 Data memory 的输入,这个过程即为 CPU 写 Cache。当 CPU 读 Cache 时,从 Data memory 输出的 Data 和 parity 输入 BCH 译码器,译码的结果除了 Data 外,还会生成

一个错误标志 err,BCH 译码器把译码结果报告给 Cache 控制器。如果错误标志 err 为 0,说明 BCH 校验结果正确,Data 未发生错误,Cache 控制器直接把数据(ico. data)送给整数单元 IU;如果错误标志 err 为 1,则说明 Data 发生了错误,这时 Cache 控制器申请访问主存,从主存中重新取得 Data,即图 3 中的 mcio. data 传给整数单元 IU。以上过程即为 BCH 编译码器对 Cache 的检错实现。

4 仿真结果及分析

4.1 仿真结果

本文使用 VHDL 实现所设计方案,仿真环境使用 QuestaSim 软件,工作频率为 50 MHz。分别对指令 Cache 和数据 Cache 的标记存储器和数据存储器随机注入 1-4 位错误,验证 BCH 码的检错结果,统计结果见表 2。由检错结果可以看出,设计的 BCH 编译码器能够检测出 1-4 位错误,达到了本方案对 Cache 的标记存储器和数据存储器的检错要求。

表 2 检错结果

错误位置	注入错误位数			
	1	2	3	4
标记存储器	可检错	可检错	可检错	可检错
数据存储器	可检错	可检错	可检错	可检错

4.2 性能分析

用 ISE 分别对 LEON2 处理器和指令 Cache 进行综合,比较添加 BCH 编译码器前后的性能结果,如表 3 和表 4 所示。

表 3 LEON2 处理器性能对比

项目	增加前	增加后
器件系列	Spartan-3A	Spartan-3A
4 input LUTs	8 728	9 915
Flip Flops	2 144	2 188
关键路径延迟/ns	6. 542	6. 542

表 4 指令 Cache 性能对比

项目	增加前	增加后
器件系列	Spartan-3A	Spartan-3A
4 input LUTs	289	291
Flip Flops	53	53
关键路径延迟/ns	5. 85	5. 921
最高工作频率/MHz	228. 029	212. 222

由综合结果可以看出,按本方案为 Cache 添加检错模块 BCH 编译码器后,资源占用量 4 input LUTs 数量增加了 13. 6%,Flip Flops 数量增加了

2.1%, 关键路径延迟没有变化, 对整个处理器性能影响并不明显. 而对于指令 Cache 来说, 增加检错模块 BCH 编译码器后, 4 input LUTs 数量增加了 0.7%, Flip Flops 数量不变, 关键路径延迟增加了 0.071 ns, 最高工作频率降低了 7%, 说明本设计对处理器和 Cache 的性能影响在合理范围内, 对将来片上实现和实际应用具有参考价值.

4.3 开销分析

本文方案与国内现有的 ECC 和分组校验等方案相比, 时间开销相同, Cache 的容错能力却显著提升, 由纠一检二提高至纠检 4 位错误. 本方案在 Cache 中对 32 位数据信息位和 25 位标记信息位增加的校验位均为 24 位, 和现有方案相比码率较低, 硬件开销增大, 但是本方案仍具备显著的优势, 即能纠检 Cache 中可能出现的 1-4 位随机错误, 这是国内现有方案不能实现的, 综合而言本设计具有一定的创新性和理论意义.

5 结束语

本文针对空间处理器的高速缓存由于 SEU 和 MBU 引发的故障, 分别对指令/数据 Cache 中的标记存储器 and 数据存储器设计了并行的 BCH 编译码器, 实现 1-4 位错误检测, 并对故障实现了有效规避. 仿真结果表明, BCH 编译码器可以有效检测 Cache 中可能出现的 1-4 位故障, 并且编译码结构都较为简单, 容易实现, 同时资源和性能开销不大, 为将来应用于空间处理器的抗辐射加固设计提供了理论支撑, 对其可靠性的提高具有一定的理论意义. 但是相对于 BCH 译码纠错的方法^[9], 这种方案的时间开销比较大. 因此, 在 RTL 级实现完整的 BCH 译码算法, 对 Cache 中检测到的 1-4 位错直接纠正, 降低纠检错的时间开销, 是下一步的研究方向.

参考文献:

[1] Baumann R C. Soft Errors in advanced semiconductor

devices - Part 1: The three radiation source [J]. IEEE Transactions on Device and Materials Reliability, 2001, 1(1):17-22.

- [2] Radaelli D, Puchner H, Skip Wong, et al. Investigation of multi-bit upsets in a 150 nm technology SRAM device[J]. IEEE Transactions on Nuclear Science, 2005, 52(6): 2433-2437.
- [3] 王新梅, 肖国镇. 纠错码-原理与方法[M]. 西安: 西安电子科技大学出版社, 2001.
- [4] 郭鹏, 房亮, 于沛玲. BCH 编译码器在 NAND Flash 控制器中的应用研究[J]. 计算机技术与发展, 2014, 24(1):179-183.
- [5] 蔺晓龙, 何文涛, 徐建华, 等. 多模 GNSS 接收机中 BCH(15, 11, 1) 并行解码算法[J]. 微电子学与计算机, 2014, 31(2):50-53.
- [6] 辛明瑞. 面向空间应用的容错 RISC 处理器体系结构研究[D]. 西安: 西北工业大学, 2006: 60-95.
- [7] 赵华, 殷奎喜. (15, 7) BCH 码编译码器的 VHDL 设计[J]. 现代电子技术, 2004, 27(20):100-101.
- [8] 孔挺, 宫芳, 方扬扬. 基于查找表的 BCH 码快速译码算法[J]. 哈尔滨商业大学学报, 2011, 27(6):819-823.
- [9] Alaa R Alameldeen, Ilya Wagner, Zeshan Chishti, 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, NY, USA: 2011:461-472.

作者简介:

王珍珍 女, (1988-), 硕士研究生. 研究方向为处理器容错技术. E-mail: wangzz062087@163.com.

杨志涛 男, (1986-), 学士. 研究方向为智能控制与无线通信技术.

王晶 女, (1982-), 博士, 讲师. 研究方向为计算机网络、操作系统和系统软件.

张伟功 男, (1967-), 博士, 研究员, 博士研究生导师. 研究方向为嵌入式计算机系统结构与容错技术、高可靠高性能处理器与 SoC 设计方法、高性能计算与数据处理等.