文章编号:1671-4598(2024)03-0293-07 DOI:10.16526/j.cnki.11-4762/tp.2024.03.043 中图分类号:V19 文献标识码:A

基于双核锁步的多核处理器 SEU 加固方法

郭 强[1], 伍攀峰[1,2], 许振龙[1]
(1. 山东航天电子科学技术研究所, 山东 烟台 264006;
2. 哈尔滨工业大学 航天学院, 哈尔滨 150001)

摘要: 以单粒子翻转为代表的软错误是制约 COTS 器件空间应用的主要因素之一; 为了满足空间应用对高集成卫星电子系统抗辐照防护的要求, 提出了一种面向通用多核处理器的单粒子翻转加固方法, 通过软件层面双核互检, 在不额外增加硬件开销的前提下, 充分提高了 COTS 器件的可靠性, 具有良好的可移植性和较强的工程实用价值; 进行软件故障注入实验, 在程序执行的关键节点注入错误信息, 验证该双核互检方法实用性; 实验结果表明双核互锁方法可以 100%检测出系统中产生的单粒子翻转, 抗软错误能力满足应用需要。

关键词: 双核锁步; 抗辐射加固; 单粒子翻转; 多核处理器; 软件故障注入

SEU Reinforcement Method for Multicore Processors Based on Dual Core Lockstep

GUO Qiang[1], WU Panfeng[1,2], XU Zhenlong[1]
(1. Shandong Aerospace Electronic Science and Technology Research Institute, Yantai 264006, China;
2. School of Astronautics, Harbin Institute of Technology, Harbin 150001, China)

Abstract: It is one of the main factors restricting the spatial application of commercial off-the-shelf (COTS) devices for the soft error represented by single event flipping; In order to meet the requirements of radiation protection for highly integrated satellite electronic systems in space applications, a single event flip reinforcement method for general multi-core processor is proposed. Through the dual core and mutual inspection in software level, the reliability of COTS devices is fully improved without additional hardware overhead, with a good portability and strong engineering practical value; The software fault injection experiments are conducted to inject the error information at the key nodes during execution program, and verify the practicality of the dual core and mutual inspection method; The experimental results show that the dual core interlocking method can fully detect the single event flipping generated in the system, and meet the application of its ability to resist soft errors.

Keywords: dual core lock step; radiation hardening; single-event upset (SEU); multicore processor; software fault injection

0 引言

近年来, 随着国内外航天产业和新兴技术的蓬勃发展, 卫星功能密度和星上实时性要求日益增长, 基于传统单核处理器设计的星载计算机系统已无法满足航天器任务需求, 多核处理器应用于航天任务已成为了发展的必然[1]。

航天应用的电子产品要考虑空间辐射效应的影响。空间辐射环境中的高能质子、粒子等都有可能使高集成卫星电子系统中的半导体器件在敏感区域产生瞬态脉冲, 进一步触发单粒子翻转 (SEU, single event upsets) 事件, 导致系统得到错误的运行结果以及系统中的其他故障, 如挂起和崩溃等[2]。当辐射足够高时, 可能致使存储器单元、寄存器、锁存器和触发器的数据位发生翻转, 从而引起系统错误[3]。单粒子效应引发的错误类型包括硬错误和软错误两种不同类型的错误。随着集成电路工艺尺寸的持续优化, 软错误出现的几率大幅度提高, 已成为航天器设计中的重点研究对象[4]。

就目前的技术而言, 传统的星载计算机一般采用由专门定制的宇航级器件设计而成, 宇航级的器件考虑了真空、辐射、原子氧等空间环境适应性和长期工作可靠性, 但在性能上普遍落后于同期商用货架产品 (COTS, commercial off-the-shelf), 而且价格成本远远高于后者。相比于高等级抗辐射加固宇航器件, COTS 器件因性能优异、成熟度高、采购成本低廉的特性在国际上已得到充分认可[5]。但 COTS 器件在芯片设计时没有进行抗辐射加固设计, 应用于空间任务易发生单粒子翻转, 进而对星上电子系统产生影响, 需要进行相应的抗 SEU 容错[6]。

星上电子系统常用的抗单粒子加固措施有双机切换、多模冗余、错误检测与纠正 (EDAC, error detection and

收稿日期:2023-10-06; **修回日期**:2023-11-10。
作者简介:郭 强(1999-),男,硕士,助理工程师。
通讯作者:伍攀峰(1982-),男,硕士,研究员。
引用格式:郭 强,伍攀峰,许振龙.基于双核锁步的多核处理器 SEU 加固方法[J].计算机测量与控制,2024,32(3):293-299.

correction）检验等方式。文献［7］针对 PowerPC460 处理器核原有存储机制以及纠检错加固算法展开研究，实现了对 PowerPC460 两种不同纠错能力的 EDAC 加固设计；文献［8］中采用软硬件结合的方式，设计了一种低成本容错方法。针对星载计算机硬件采用双机冷备方案，同时通过现场可编程门阵列（FPGA，field programmable gate array）仲裁处理器状态，在检测到异常时进行切机；文献［9］提出了一种三模冗余设计，采用部组件级冗余，3 个 CPU 进行热冗余备份，通过 FPGA 完成仲裁，信息通过高速串行外设接口（SPI，serial peripheral interface）实现信息交互；文献［10］中在可重构 FPGA 内部设置独立工作的三核处理器，通过反熔丝 FPGA 可对任一处理器进行重构，构成了可修复的三模冗余系统。

尽管硬件层面的加固方式可以有效提高电子系统的可靠性，但也带来了额外的硬件开销和资源占用问题。从软件层面通过牺牲处理器部分性能来提高系统可靠性是另一种行之有效的方法。车辆、航空电子系统中常用锁步技术来实现双核处理器抗软错误软件加固[11-14]。文献［15］提出了一种扩展的双核锁步方法，将执行周期和锁步周期分离，并引入了" 写历史表" 的概念，降低了系统性能开销，但需要额外引入监测模块对两核计算结果作判断；文献［16］发明了一种基于 Zynq-7000 的双核 ARM 处理器抗软错误防护方法，结合基于复算域的双核互检方法和基于检查点的回卷恢复方法，可在双核处理器核间实现软错误的检测和恢复，但检查点设置在每段程序末尾，回卷带来的系统性能开销过大。

本文基于双核锁步机制，提出了一种多核处理器核间互检机制，在不引入额外硬件的前提下，将写命令作为 CPU 间的同步点，检查点在写同步点中选取，仅对系统关键数据的改动做记录，以较小的性能开销实现了软错误检测和系统恢复。最后采用软件故障注入方式对加固前后的处理器抗单粒子翻转能力进行评估，验证了该机制抗软错误能力满足应用的需要。

1 双核锁步机制

锁步技术是一种错误检测机制，常应用于车辆、航空电子系统中，是双核处理器抗软错误加固设计的重要技术手段。

双核锁步系统通常被称为主从锁步处理器。冗余处理器称为从属，在开机复位（POR，power on reset）时以与主处理器相同的状态开始，运行相同的指令，接收相同的输入，并在没有发生错误时在每个时钟周期中生成相同的输出。如果主核和从核输出结果不同，系统会自动丢弃错误并重新执行指令。如果尝试纠正错误的次数超过了允许的阈值，则会触发系统重置。双核锁步技术的原理如图 1 所示。

检查点（CP，check point）技术是在应用程序的执行过程中，每间隔一段时间将处理器当前正确的状态以检查点文件的形式存储到存储器中；回卷恢复（RR，rollback recovery）技术是在检测到系统发生故障后，由处理器从存储器中读取最近一次保存的检查点文件，使处理器恢复至上一个正确状态并重新执行故障程序，避免在发生故障后，处理器重新开始执行应用程序，从而减少了因故障带来的计算损失，有效地提高系统的可用性和容错能力，检查点和回卷恢复技术的原理如图 2 所示[17]。

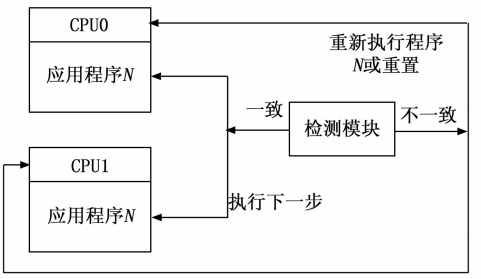

图 1 双核锁步原理

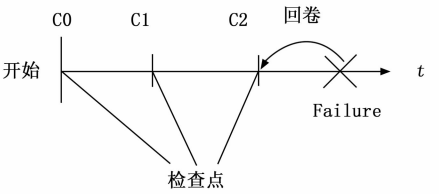

图 2 检查点和回卷恢复技术

2 双核互锁加固设计

2.1 硬件架构

本文提出的双核互锁加固设计是在 PowerPC 架构中 Qorlq 系列的一款双核处理器 P2020 上设计和实现的，该方法仅需要一些调整就可适用于其他系列处理器，是面向通用型多核处理器的抗软错误加固设计。硬件架构如图 3 所示。

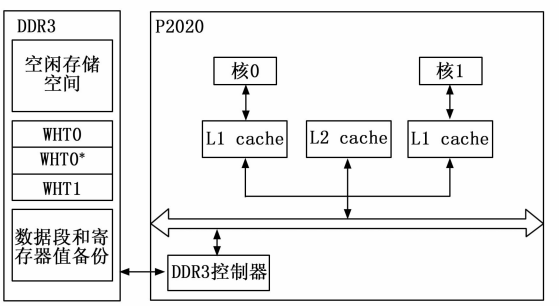

图 3 双核互锁硬件架构

两核有各自的 cache 存储互锁机制产生的过程信息，通过共享的 L2 cache 及 ddr3 存储器存储程序及运行过程中的寄存器、内存数据。本文双核互锁机制基于写操作进行同步，每次写同步都会进行两核交互，若彼此计算结果一致则继续执行程序，否则将系统状态恢复到最近的检查点，重新执行部分指令。

DDR3 中设置两级历史表（WHT，write history table）文件，分别存储第 N 个和第 $N-1$ 个锁步周期内产生的写命令，写同步处更新历史表 0，检查点处更新历史表 1。当系统检测到错误时，调用回卷中断，从历史表中获取自上一个检查点起被更改的数据，并进行恢复。

2.2　软件平台

本文双核互锁设计基于 Vxworks 操作系统实现。Vxworks 操作系统是美国风河（WindRiver）公司于 1983 年开发的一种高性能、可裁剪的嵌入式实时操作系统，具有较强的时性和可靠性，广泛应用在航空、航天、军事等实时性要求极高的高精尖技术领域中[18-19]。

本文使用 Vxworks 操作系统搭建非对称多处理（AMP，asymmetric multiprocessing）工作环境，在两核中运行相同的程序，通过消息队列进行两核交互，使用 PowerPC 处理器自带的核间中断进行检查点和回卷恢复操作。

3　双核互锁加固实现

在双核互锁机制中，两处理器运行相同的应用程序，基于写指令进行同步，每 N 个写同步设置一个验证点（VP，verification point）。根据应用程序功能和可靠性要求等因素，合理设置 VP 的数量和间隔。图 4 详细介绍了 CPU0、CPU1 的执行概况及互锁加固流程。

1）写同步：CPU0、CPU1 执行相同的应用程序，在写命令处进行同步，交换数据；

2）一致性判断：CPU 写同步后，将自身数据与来自另一处理器的数据进行比较，比较结果有一致和不一致两种；

3）WHT0：CPU0 的一级写历史表，负责存储第 N 个检查点（最近的一次检查点）后 CPU0 对关键数据的写命令；

4）WHT0＊：CPU1 的一级写历史表，负责存储第 N 个检查点（最近的一次检查点）后 CPU1 对关键数据的写命令；

5）VP：检查点；

6）WHT1：二级写历史表，负责存储第 $N-1$ 个检查点到第 N 个检查点（最近的一次检查点）间处理器对关键数据的写命令，在检查点处更新。

数据初始化后，两核并行执行相同应用程序，在执行到写命令时进行同步。在写同步处，两核通过消息队列进行写指令交互，然后比较二者的差异，如果对比结果一致，则认为系统中没有发生故障，更新 WHT0、WHT0＊；若对比结果不一致，则认为当前系统中发生故障，生成回卷恢复中断信号，CPU0、CPU1 分别响应中断信号，使系统恢复到上一个验证点状态。

执行到验证点时，若 CPU 写同步均判断一致，则代表连续通过了 N 个同步点，此时应用程序暂停执行，对比 WHT0 和 WHT0＊，若表决一致，两核生成检查点中断信号，调用中断服务例程获取 WHT0 内容，覆盖 WHT1，并清空 WHT0、WHT0＊，然后继续执行应用程序。同一时刻，DDR3 存储器中保留有恢复至系统第 N 个检查点的 WHT0 和恢复至第 $N-1$ 个检查点的 WHT1 文件。

处理器调用回卷中断程序，即写同步判断不一致时，CPU0、CPU1 从 WHT0 中获取第 N 个检查点后的所有写命令，执行回卷恢复操作，从第 N 个检查点重新执行该段程序，直到应用程序执行结束。

在本文中，写操作范围被定义为：

1）通用寄存器：GPR0-GPR31；

2）特殊寄存器：条件寄存器（CR，condition register）、链接寄存器（LR，link register）和计数寄存器（CTR，count register）等；

3）DDR3 存储器。

同时，在设计时要考虑以下问题：

1）CPU0、CPU1 连接到外部 DDR3 存储器，并将应用程序和检查点文件存储在不同地址中；

2）CPU0、CPU1 共享 DDR3 内存，对于不可同时访问的资源需要设置互斥信号量等进行保护；

3）验证点在所有写同步中选取，但二者并不等同，大部分写同步不对 WHT1 作改动。

3.1 互锁判断

在双核互锁机制中，由 CPU 本身担任监测模块，任一 CPU 判断两核写命令不一致，则认为系统中出现了软错误，需要采取相应措施。监测机制基于 Vxworks 操作系统编写，采用 C 语言进行设计，它负责验证 CPU0 和 CPU1 的一致性并通过输出的中断信号控制锁步。监测模块通过比较 CPU0、CPU1 的写命令来判断二者是否一致，比较方式有以下两种。

1）第一种方法：将 CPU0、CPU1 同步时的写地址和写数据分别发送给对方，然后二者对各自数据的每一位进行比较；

2）第二种方法：在 CPU0、CPU1 写同步的数据中应用签名，两 CPU 仅对比签名来判断是否一致，签名方式有数据总和、异或掩码等。

在第一种方法中，不需要对应用程序做额外修改，优点是监测覆盖率较大，但会增大系统开销。在第二种方法中，需要在同步前计算签名，即在应用程序上附加代码，但降低了监测模块对比数据量，有效降低了系统消耗时间。

综合考虑上述两种方法的优缺点，采用第二种方法在 CPU0、CPU1 的写同步数据中应用签名，而后再由 CPU0、CPU1 进行对比，签名的实现方式是：将写地址和写数据相与，然后计算总和中二进制数据“1”的个数，如表 1 所示。

表 1　CPU0 写同步应用签名计算过程

数据来源	写地址	写数据	相与	签名
CPU0	0x2409b8	0x12d687	0x86	0x03
CPU1	0x2409b8	0x12d687	0x86	0x03

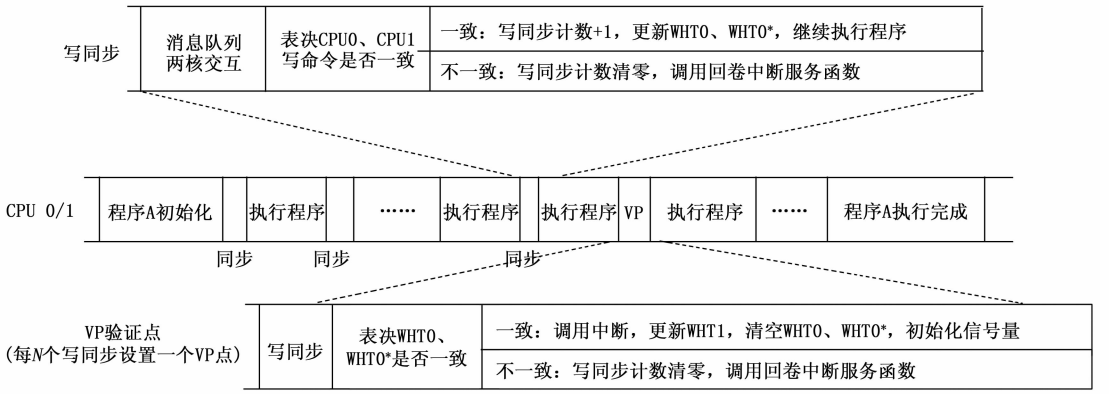

图 4 双核互锁加固流程

以 CPU0 为例，写同步流程如图 5 所示。首先 CPU0 计算自身写地址和写数据的签名，然后通过消息队列将签名发送至 CPU1，等待来自 CPU1 的签名，超时则触发回卷。

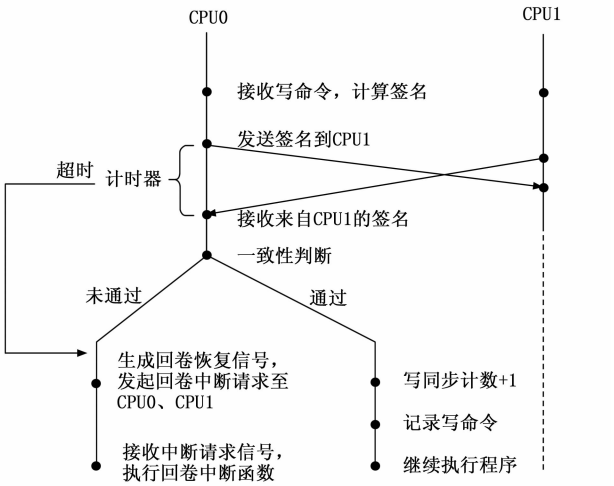

图 5 双核互锁 CPU0 写同步过程

接收到来自 CPU1 的签名后，CPU0 进行二者对比，不一致触发回卷。若一致性判断通过，则记录写命令，继续执行程序。

3.2 检查点与回卷恢复技术

3.2.1 检查点技术

检查点的设置基于 CPU0、CPU1 的写同步，设置检查点间隔为 N，每连续 N 个写同步检验通过，就向 CPU0、CPU1 发送一个检查点中断请求，CPU0、CPU1 响应中断信号后执行检查点中断操作。处理器响应中断流程如图 6 所示。

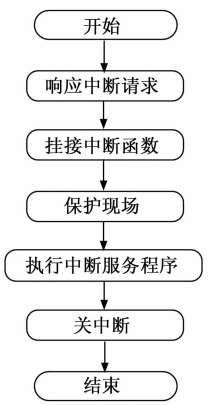

图 6 处理器响应中断流程

在检查点执行过程中，发送中断请求，调用中断服务函数，初始化相关信号量和共享变量，获取 WHT0 中数据形成检查点文件，覆盖历史表 1，并清空历史表 0。

3.2.2 回卷恢复技术

当任何一次写同步检测到两核写命令不同时，系统进入回卷恢复操作。为了尽可能降低系统回卷引入的性能资源消耗，本文设置了四级回卷，如图 7 所示。

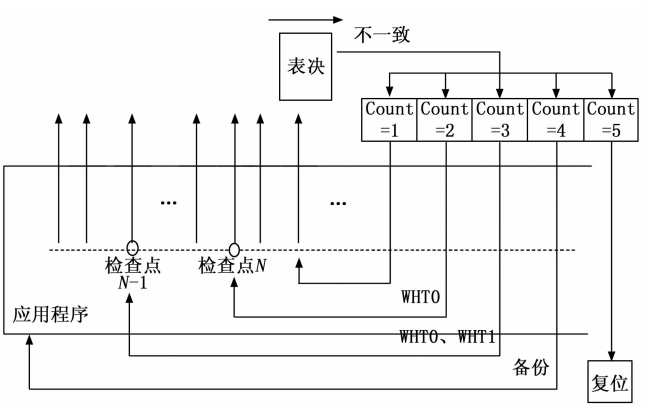

图 7 回卷恢复流程

一级回卷中断：一次表决不一致时，回卷计数值加 1，重新执行该指令；

二级回卷中断：连续两次表决不一致，按照 WHT0 内容修改 DDR3 中内存数据段与 CPU 寄存器的值，使它们恢复到第 N 个检查点状态，回卷计数值加 1，其余信号清零，退出中断；

三级回卷中断：按照 WHT0、WHT1 内容修改 DDR3

中内存数据段与堆栈数据段的值，使它们恢复到第 $N-1$ 个检查点状态，回卷计数值加 1，其余信号清零，退出中断；

四级回卷中断：中断服务程序用 DDR3 中的备份数据覆盖内存数据段、CPU 寄存器值，清空 WHT0 和 WHT1，清零所有信号，重新执行该程序。

当连续回卷计数为 5 时，代表四级回卷恢复机制仍未解决当前系统中出现的错误，控制系统复位。此外，对双核锁步过程中的数据传输进行信息冗余加固，提高系统可靠性。

3.3　核间交互

双核互锁设计以较小的性能和资源代价来换取系统可靠性的提升，两核互为检查模块，任一核认为数据异常则默认系统受到软错误影响，可见两核的交互是双核互锁设计的关键，包括两核交互、交互方式、交互频率等，两核交互设计的好坏决定了系统性能开销大小。

3.3.1　交互内容

交互内容是指系统在检查点操作时具体保存的某些状态。交互内容设置详细，那么通过回卷技术可以有效地使系统回到上一个检查点时的正确状态，重新执行程序段，得到正确的结果。相应的，也会带来系统检查点操作和回卷操作时间的增加；上下文内容缺失，会导致无法通过回卷技术得到正确的系统状态，进而不能够产生正确的计算结果，使整个锁步机制失效。

在本文中，交互内容包括写命令对内存数据段的改变以及 CPU 各寄存器值，考虑到 DDR3 存储器受错误检查和纠正（ECC，error checking and correcting）技术加固，认为内存数据可靠性较高，并在每个程序开始执行时进行一次内存数据段备份，程序运行到检查点时仅更新交互内容不对内存数据段进行额外操作。

3.3.2　交互方式

通常情况下，锁步系统通过一致性检验进行两处理器的同步。目前主要的设置方式有写操作同步和划分程序块同步两种。前者是在处理器运行到写操作时，通过等待的方式进行同步，当完成一致性检验后，两个处理器同时开始后续指令执行；划分程序块是指将应用程序划分为多个块，在两块之间设置验证点，当到达验证点时，通过等待的方式实现同步。

在本文中，考虑到划分程序块可能使错误数据污染内存，甚至对其他系统造成严重影响的情况，选择基于写操作同步的设置方式。但由于写操作频率在不同程序间的不确定性，本文选择将检查点设置与一致性检验设置进行区分，每次写操作都进行一致性检验，多次写操作后进行一次检查点操作。同时设置四级回卷，针对潜在单粒子影响有一定的抵抗能力。

3.3.3　交互频率

检查点间隔的选取对系统性能影响很大。检查点设置密集，意味着需要频繁进行检查点操作，且易产生潜在单粒子影响无法消除的现象；检查点设置稀疏，两个处理器计算结果不一致，触发回卷后需要重新执行很长一段程序。致使系统产生不必要的性能开销。

在本文中，将检查点设置与一致性检验设置进行区分，每次写同步都进行一致性检验，多次写同步后进行一次检查点操作。通过写请求先到等后到的方式进行 CPU0、CPU1 的同步。检查点数量和回卷数量设置不一致，当写同步一致性表决不通过时，根据连续回卷次数，产生相应的回卷中断信号。

4　故障注入基本原理与方案设计

4.1　软件故障注入技术

软件故障注入技术（SWIFI，software implemented fault injection）是在软件层次来实现处理器的故障注入方法，该技术不需要任何硬件设备就可以实现故障的模拟注入，而是在处理器程序编译或者运行时动态的修改执行程序，或者人为设置寄存器值来改变处理器的运行状态，这种注入方法不需要任何额外的开销，属于一种低成本的故障注入技术，且故障注入位置灵活可控，具有相当高的自由性。另一方面，通过脚本或者程序可以实现批量的故障注入操作，完成故障注入的全自动化设置。

故障注入会因实现方式的不同或者注入对象的不同而略有差异，但是基本设计思想是一致的[20]，其原理如图 8 所示。

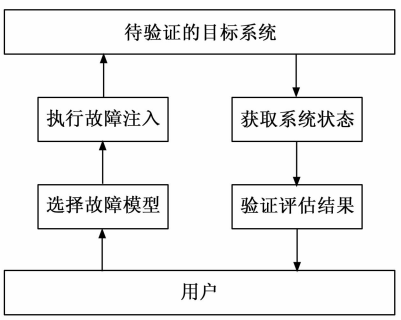

图 8　故障注入原理图

主要通过定时器超时、代码插入和异常/陷阱等技术手段[21]，对双核锁步机制及个容错技术进行故障注入实验，模拟系统中上至双核处理器、下至处理器的寄存器、存储单元等多个层次可能发生的故障，验证双核互锁机制是否能够有效工作，提高系统可靠性。

4.2　故障注入设计方案

为验证本文提出的双核互锁机制的抗软错误的有效性，在两核写同步、检查点两处注入故障来模拟系统中出现的单粒子翻转现象，观察 CPU0、CPU1 是否能正确识别软错误，并通过回卷恢复技术得出正确的结果。

4.2.1　同步点故障注入

在双核互锁机制中，CPU0 和 CPU1 执行相同的程序，在写同步处进行数据交互，从而进行一致性判断，产生相

应的检查点中断信号或回卷中断信号。因此，为了模拟处理器在执行任务过程中数据位翻转的情况，可以通过故障注入的方法直接对处理器写命令的计算签名进行修改，故障注入位置如图 9 所示。

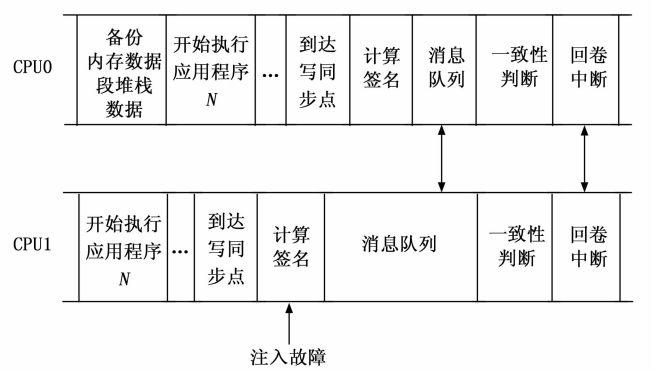

图 9 同步点故障注入

故障注入步骤如下：

1）设置信号量 sig0，核 1 每次写命令时 sig0 信号自增；

2）核 1 写同步计算签名后读取 sig0 信号，若大于预设值，则进入故障注入函数；

3）同步点故障注入函数获取核 1 计算后的签名，与 0x55555555 做与运算，将运算结果重新赋值给签名；

4）核 1 得到故障注入后的签名，通过消息队列与核 0 交换数据，两核进行一致性判断。

连续修改一次到五次写同步后形成的核 0 的计算签名，每次翻转签名中的一个 bit 位，模拟执行应用过程中数据位发生翻转的情况，致使系统产生回卷恢复中断信号，并观察系统回卷级数，后续是否能恢复到正确的系统状态，产生正确输出。

4.2.2 检查点文件故障注入

检查点文件是双核锁步机制的核心，是系统能恢复到上一个正确状态的依据。CPU 执行到检查点处时，分别从 WHT0、WHT0 * 获取数据，生成检查点文件。因此，为了模拟处理器在检查点处数据位翻转情况，可以在检查点文件生成时进行故障注入，故障注入位置如图 10 所示。

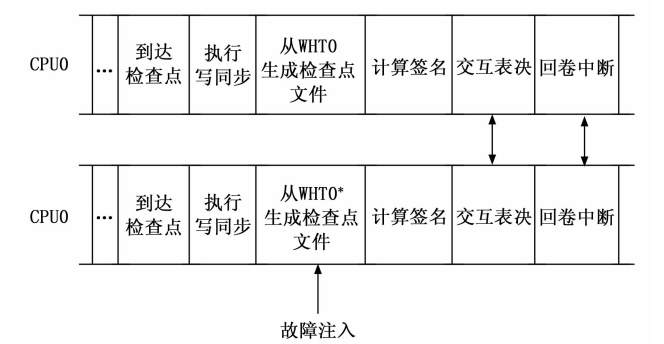

图 10 检查点文件故障注入

故障注入步骤如下：

1）设置信号量 sig1，核 1 每次进入检查点时 sig1 信号自增；

2）核 1 在计算检查点文件签名前读取 sig1 信号，若大于预设值，则进入检查点故障注入函数；

3）在检查点故障注入函数中，手动设置不同的故障注入位置，分别将检查点文件中各寄存器的值与 0x55555555 做与运算，运算结果保存到检查点文件中；

4）核 1 计算检查点文件签名，通过消息队列与核 0 交换数据，进行一致性判断。

通过修改 CPU1 生成检查点文件中各寄存器值，翻转其中的 1 bit 位，模拟从堆栈地址中读取各寄存器值时发生数据位翻转的情况，致使 CPU0、CPU1 在对比检查点文件的签名时产生回卷恢复中断信号，从而达到故障注入的目的。同样通过连续注入一次到四次错误，观察系统回卷级数，是否能恢复到正确的系统状态。

5 故障注入试验结果分析

根据 4.2 所述的软件故障注入方案，分别进行相应的故障注入设计，其实验结果如表 2 所示。

表 2 双核互锁故障注入实验结果

注入位置	注入次数	核 0 表决结果	核 1 表决结果	中断类型	回卷次数	实验次数	输出正确率
W1	0	succeed	succeed	/	0	5	100%
W1	1	fail	fail	一级回卷中断	1	5	100%
W1	2	fail	fail	二级回卷中断	2	5	100%
W1	3	fail	fail	三级回卷中断	3	5	100%
W1	4	fail	fail	四级回卷中断	4	5	100%
W1	5	fail	fail	复位	4	5	100%
W2	0	succeed	succeed	/	0	5	100%
W2	1	fail	fail	一级回卷中断	1	5	100%
W2	2	fail	fail	二级回卷中断	2	5	100%
W2	3	fail	fail	三级回卷中断	3	5	100%
W2	4	fail	fail	四级回卷中断	4	5	100%
W2	5	fail	fail	复位	4	5	100%
r1	0	succeed	succeed	/	0	5	100%
r1	1	fail	fail	一级回卷中断	1	5	100%
r1	2	fail	fail	二级回卷中断	2	5	100%
r1	3	fail	fail	三级回卷中断	3	5	100%
r1	4	fail	fail	四级回卷中断	4	5	100%
r1	5	fail	fail	复位	4	5	100%
CR	0	succeed	succeed	/	0	5	100%
CR	1	fail	fail	一级回卷中断	1	5	100%
CR	2	fail	fail	二级回卷中断	2	5	100%
CR	3	fail	fail	三级回卷中断	3	5	100%
CR	4	fail	fail	四级回卷中断	4	5	100%
CR	5	fail	fail	复位	4	5	100%
LR	0	succeed	succeed	/	0	5	100%
LR	1	fail	fail	一级回卷中断	1	5	100%
LR	2	fail	fail	二级回卷中断	2	5	100%
LR	3	fail	fail	三级回卷中断	3	5	100%
LR	4	fail	fail	四级回卷中断	4	5	100%
LR	5	fail	fail	复位	4	5	100%

其中，W1 点是该应用程序的非首个写同步点，W2 点是程序的非首个检查点，r1 是通用寄存器，CR 是条件寄存器，LR 是链接寄存器。

在写同步计算签名过程中，选取了 W1、W2 两个程序运行位置进行了故障注入，每个注入点进行了 1 次注入到连续 5 次注入实验，双核互检机制可以稳定检查到两核运行过程中出现不一致，以及连续注入错误的数量，准确率为 100%。根据连续出错次数，双核互检机制选取不同的恢复策略：1 次错误时调用一级回卷中断函数，重新执行指令；连续 2 次到 3 次错误时，系统调用二级、三级回卷中断函数，分别回卷至第 N 次、第 $N-1$ 次检查点的系统状态；连续 4 次出现错误，系统调用四级回卷中断，将应用程序初始化时备份的程序段传入内存，重新执行该程序；当连续超过 4 次错误时，处理器复位。

在检查点文件中，选取通用寄存器 r1、特殊寄存器 CR、LR 三个故障注入点，同样各个注入点分别进行 1 至 5 次手动造错。多次实验下，CPU0、CPU1 进行一致性表决时，均可得到“fail”的表决结果，并且根据连续回卷计数触发正确的回卷中断信号。连续注入故障不超过 4 次时，CPU0、CPU1 能通过回卷恢复能到正确的输出结果，避免了类似的单粒子翻转对系统可靠性的影响。当连续回卷计数值达到 5 时，认为该系统已无法通过回卷恢复技术纠正错误，进行复位操作。

综上，本文双核互锁机制可以在不额外增加硬件的前提下，牺牲小部分处理器性能，使系统具备较好的软错误检测和系统恢复能力。

6　结束语

本文提出了一种面向通用处理器的双核互检抗软错误加固方法，该方法基于写命令进行 CPU0、CPU1 同步，结合检查点、回卷恢复技术实现正确状态的保存和恢复。通过简化检查点存储数据和四级回卷恢复机制，降低处理器资源开销的同时有效保障了系统可靠性。

软件层面改进的双核互检算法面向通用型多核处理器，具有良好的可移植性。根据软件故障注入仿真结果可知，双核互检方法可以 100%检测到程序运行关键数据发生的翻转，并通过多级回卷技术得到正确的结果，具备较强的工程实用价值。双核互检加固方法对处理器性能影响的评估将是未来工作的重心。

参考文献：

[1] 赵振博. 多核星载计算系统任务调度技术研究 [D]. 哈尔滨：哈尔滨工业大学，2022.

[2] TAMBARA，LUCAS ANTUNES，RECH，et al. Analyzing the impact of radiation-induced failures in programmable SoCs [J]. IEEE Transactions on Nuclear Science，2016，63 (4)：2217 - 2224.

[3] 邢泽全，郭绍陶，郑　已. 一种数字电路单粒子翻转测试方法 [J]. 微处理机，2023，44 (1)：10 - 13.

[4] 余淇睿，张战刚，李　斌，等. 大气中子及 α 粒子对芯片软错误的贡献趋势 [J]. 电子与封装，2023，23 (8)：70 - 76.

[5] 薄　鹏，汪　悦. 面向航天器型号的 COTS 元器件选用策略 [J]. 航天器环境工程，2023，40 (4)：430 - 436.

[6] 吴晓宏. 抗辐照封装加固的 COTS 器件及其制备方法 [P]. 中国：202111162341.5. 2022 - 01 - 28.

[7] 马　尤. PowerPC 处理器的加固设计与验证 [D]. 西安：西安电子科技大学，2019.

[8] 朱明俊，周宇杰. 一种低成本纳卫星星载计算机容错方法 [J]. 航天器工程，2016，25 (2)：52 - 57.

[9] 何　健，张旭光，刘凯俊，等. 基于三模冗余设计的低成本高可信微纳通用计算机 [J]. 计算机测量与控制，2015，23 (7)：2556 - 2558.

[10] 蒲卫华，谢成清，姜文志，等. COTS 星载计算机容错设计及可靠性研究 [J]. 空间控制技术与应用，2020，46 (3)：72 - 77.

[11] GOMEZ-CORNEJO，JULEN，ZULOAGA，AITZOL，KRETZSCHMAR，ULI，et al. 39th Annual conference of the IEEE industrial electronics society [C] //Vienna，Austria. Institute of Electrical and Electronics Engineers，2013：2261 - 2266.

[12] 王明羽. 一种基于双核锁步处理器的故障处理方法 [P]. 中国：202310210412. 7. 2023 - 06 - 23.

[13] ádria Barros de Oliveira，Lucas Antunes Tambara，Fernanda lima kastensmidt. 8th IEEE latin american symposium on circuits and systems [C] //Bariloche，Argentina. Institute of Electrical and Electronic Engineers，2017：1 - 4.

[14] 王　锐，段小虎. 一种具备故障自恢复能力的容错计算机架构设计 [J]. 信息通信，2017 (4)：72 - 73.

[15] ABATE F，STERPONE L，LISBOA C A，et al. New techniques for improving the performance of the lockstep architecture for SEEs mitigation in FPGA embedded processors [J]. IEEE Transactions on Nuclear Science，2009，56 (4)：1992 - 2000.

[16] 崔秀海. 基于 Zynq-7000 的双核 ARM 处理器抗单粒子翻转防护方法 [P]. 中国：201810015734. 5. 2021-11-02.

[17] 周益帆，吴　咏. 一种基于行为特征的文件检查点优化策略 [J]. 软件，2017，38 (7)：137 - 142.

[18] 张国超，周　畅. 基于 VxWorks 的 IP 层数据包监听技术研究 [J]. 舰船电子工程，2023，43 (1)：112 - 116.

[19] 鲁　超. 基于 VxWorks 的卫星载荷管理软件设计与实现 [J]. 工业控制计算机，2023，36 (6)：1 - 3.

[20] 刘宪忠，赵昶宇. 嵌入式系统故障注入技术研究 [J]. 科技与创新，2022 (7)：25 - 27，34.

[21] 幺　飞，时　光，富小薇. 基于故障注入技术的航天器系统级软件测试方法研究 [J]. 航天器工程，2019，28 (1)：130 - 136.