Computer Science and Application 计算机科学与应用, 2026, 16(2), 472-481
Published Online February 2026 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2026.162076

Hans汉斯

基于PPC460的硬核化安全防护技术

秦　炜，李悦坤，陆发忠，孔祥雷

上海创景信息科技股份有限公司，上海

收稿日期：2026年1月13日；录用日期：2026年2月17日；发布日期：2026年2月25日

摘　要

在高可靠、高安全的SOC及嵌入式系统的应用中，通常会由于外部复杂环境造成系统内存区出现代码段非法操作、内存非法访问、堆栈溢出、某一关键任务崩溃或者宕机等异常，造成系统运行的关键任务失效。为解决以上问题，本文以PPC460软核处理器为基础，设计了一种软硬件结合的安全防护技术，通过硬件限定内存的访问空间和权限，进而限定软件任务只能在特定的空间具有特定的操作权限，能够有效阻止系统内存和堆栈的非法访问。通过硬核化的关键任务快照与恢复技术对稳定态任务进行快照，当有关键任务受损后及时发现并恢复该任务的原始参数，来保障整个系统稳定的运行；同时辅以异常上传模块，可供软件进行异常分析。经实验验证本设计能够对内存保护区起到安全防护功能，同时通过注入故障，能够将原有稳定任务恢复并正常运行。

关键词

硬核化，堆栈溢出，安全防护，任务快照，任务恢复

Hardware-Based Security Protection Technology Based on PPC460

Wei Qin, Yuekun Li, Fazhong Lu, Xianglei Kong

Vision Information Technology (Shanghai) Co., Ltd., Shanghai

Received: January 13, 2026; accepted: February 17, 2026; published: February 25, 2026

Abstract

In the application of highly reliable and secure SOC and embedded systems, it is common to encounter anomalies such as code segment illegal operations, memory illegal access, stack overflow, critical task crashes or crashes in the system memory area due to complex external environments, resulting in the failure of critical tasks during system operation. To solve the above problems, this article designs a security protection technology that combines software and hardware based on the PPC460 soft core processor. By limiting the access space and permissions of memory through

文章引用: 秦炜, 李悦坤, 陆发忠, 孔祥雷. 基于 PPC460 的硬核化安全防护技术[J]. 计算机科学与应用, 2026, 16(2): 472-481. DOI: 10.12677/csa.2026.162076

hardware, software tasks can only have specific operation permissions in specific spaces, which can effectively prevent illegal access to system memory and stack. By using Hardware-based critical task snapshot and recovery techniques to snapshot stable tasks, when a critical task is damaged, the original parameters of the task can be detected and restored in a timely manner to ensure the stable operation of the entire system; At the same time, it is supplemented by an abnormal upload module, which can be used for software to conduct abnormal analysis. Through experimental verification, this design can provide security protection for the memory protection area, and by injecting faults, it can restore the original stable tasks and run them normally.

Keywords

Hardware-Based Hardening, Stack Overflow, Security Protection, Task Snapshot, Task Recovery

1. 引言

虽然 SOC+嵌入式实时操作系统在工业自动化、通信、船舰、航空等领域有着广泛的应用，但是在高可靠、高安全领域如卫星、核电、导弹等技术领域中，由于复杂的外部运行环境，当系统受到攻击(或外部干扰)，一但出现内存区异常的代码段访问，特殊数据区异常访问，堆栈溢出等异常情况，便会出现系统死机或复位，造成不可挽回的损失；而针对运行在 SOC+上的实时操作系统时，则可能出现关键任务运行异常或参数严重偏离，单纯依靠传统的 SOC+嵌入式实时操作系统很难保障系统长时间安全稳定的运行。

针对以上问题，本文在 PPC460 [1]软核基础上增加了硬核化安全防护模块和任务快照恢复功能，底层硬核化模块实时监控系统中各类关键技术指标的运行状态，并通过特定中断通知处理器进行相应的异常处理操作，能够有效避免代码段的非法访问、内存空间非法访问、堆栈溢出等异常，能够对关键任务进行定时快照，并在系统检测到异常的情况下，快速恢复关键任务到快照前的参数继续运行。

2. 安全防护总体架构

本文设计了一种 PPC460 硬核化安全防护系统技术如图 1 所示，由软核(FPGA 现场可编程门阵列代码)和硬件(混合集成系统原理样机)两部分组成[2]。软核部分提供 PPC460 处理器芯片逻辑功能和操作系统硬核化逻辑功能；硬件部分提供基于 PPC460 处理器芯片的接口及测试电路，包括各类控制接口、RS232 接口、IIC 接口、SPI 接口、CAN 接口、中断及 RTINSIGHT 测试接口。

硬核化部分包括 3 个硬核化逻辑模块，包括：内存区安全防护模块、任务快照恢复模块及异常发送模块。硬核化逻辑模块协调软件和硬件一起对嵌入式系统运行中最重要的 SRAM 区及关键任务进行协同防护。

内存区安全防护模块通过 PLB 总线[3] [4]直接串联在 PLB4 总线和 SRAM 模块之间。能够实现主处理器 C8000 的 PLB 总线监控并转发，同时对代码端、内存数据段、堆栈区进行异常防护。

任务快照及恢复同样挂载在 PLB4 总线上，主要针对运行在 SOC 上的实时操作系统的关键任务，进行信息提取、定期快照、异常监控、快速恢复等功能。

异常发送模块为安全防护模块检测到异常操作后将异常信息进行记录，并同时产生异常中断，通知 CPU 进行相应的操作。同时异常发送模块支持特定指令识别、并将异常信息以及特定指令操作后发送给

外设，以便软件进行安全分析。

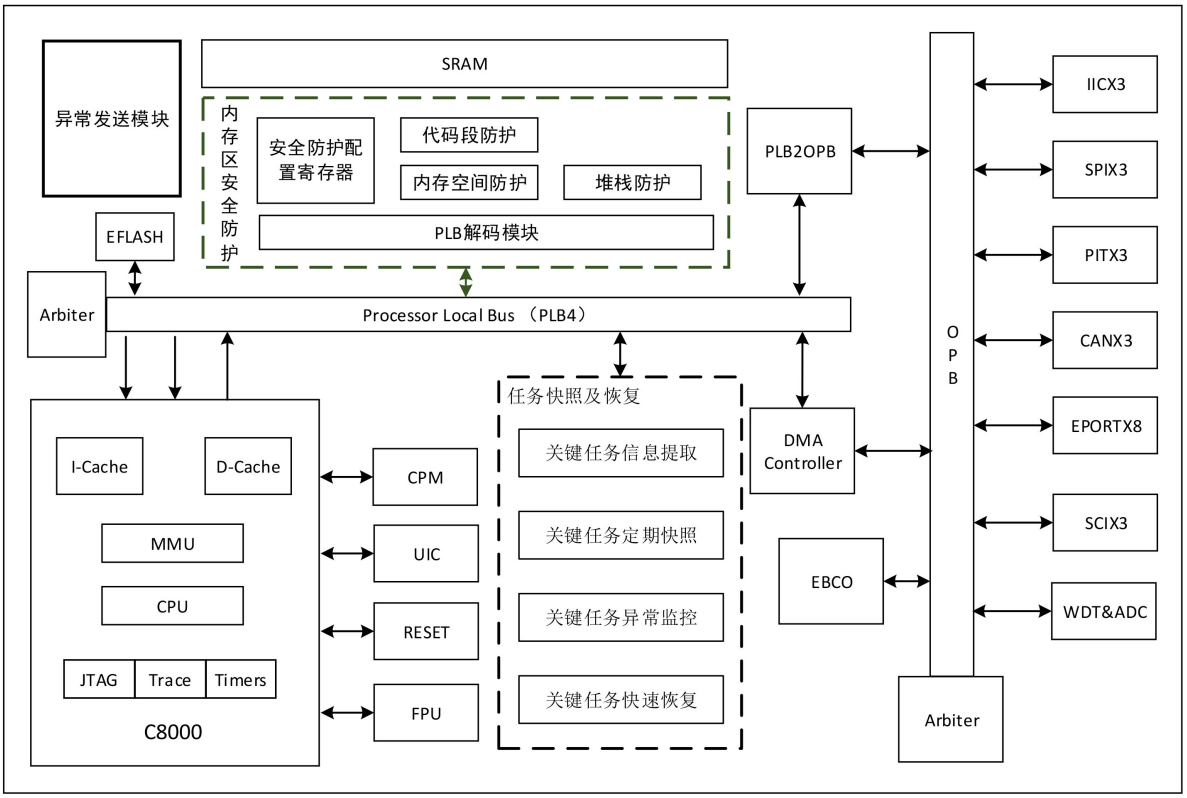

Figure 1. Security protection overall architecture
图 1. 安全防护总体架构

3. 硬核化安全防护设计

硬核化安全防护技术，采用硬件逻辑实现对安全空间的空间范围，访问权限进行有限限制，主要包括：内存区安全防护模块、任务快照及恢复模块、异常发送模块。

内存区安全防护模块主要由以下部分组成：安全防护配置寄存器、PLB 解码模块、代码段防护、内存空间访问防护、堆栈溢出防护、任务安全防护。安全防护配置寄存器是配置的代码段地址空间、内存访问地址空间、堆栈地址空间以及任务运行时间。PLB 解码模块将 PLB4 总线转化为 Local Bus 总线。

任务快照及恢复模块：通过守护任务对关键任务进行监控、快照、异常诊断、快速恢复。关键任务为系统最为重要的应用任务，在复杂环境中，一定要确保其稳定正确的运行；守护任务为控制硬件底层的逻辑任务，系统需要通过守护任务进行关键任务的定时备份、监控及重启恢复。该模块主要由以下部分组成：关键任务信息提取、关键任务定期快照、关键任务异常监控、关键任务快速恢复。

3.1. 代码段防护设计

本文设计的代码段防护功能，主要由安全防护配置寄存器和代码段安全防护模块实现。系统通过初始化安全防护配置寄存器，对代码段的空间范围以及异常模式进行设置。设定一个代码空间为只读监控，当该代码段空间存在写操作时，硬件会禁止该写操作，同时硬件将产生故障异常中断通知 CPU 进行相应的处理，同时将异常信息加入时间信息通过异常发送模块上传，以供异常故障分析使用，执行流程如图 2 所示。

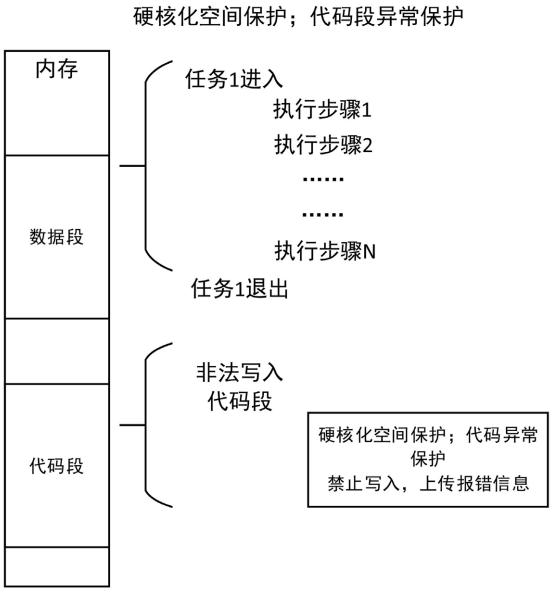

Figure 2. Code segment exception protection execution process
图 2. 代码段异常防护执行流程

3.2. 堆栈防护设计

本文设计的堆栈安全防护功能，主要由安全防护配置寄存器和堆栈防护模块[5]实现。系统通过初始化安全防护配置寄存器，对堆栈的空间范围以及异常模式进行设置。例如一个内存空间为堆栈溢出监控，系统设定需要监控的堆栈空间地址范围，当系统操作堆栈空间时出现溢出时产生异常，通过异常中断通知 CPU 进行相应处理，同时通过异常发送模块上传异常信息给软件进行异常诊断分析，如图 3 所示。

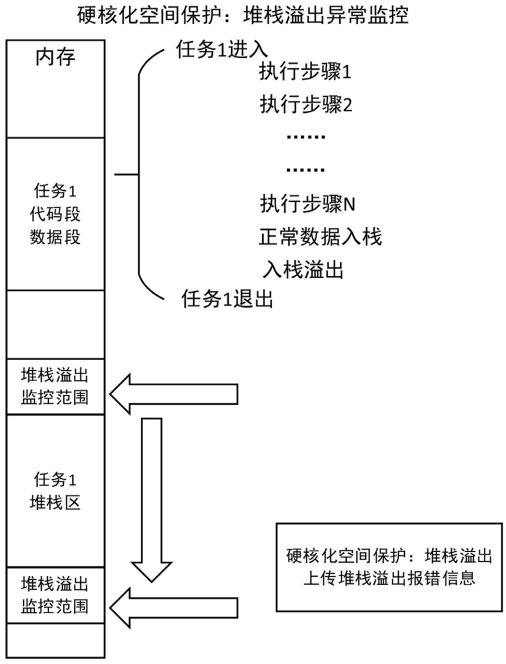

Figure 3. Stack protection execution process
图 3. 堆栈防护执行流程

3.3. 内存空间防护设计

本文设计的内存空间防护功能，主要由安全防护配置寄存器和内存空间防护模块实现。系统通过初始化安全防护配置寄存器，对内存空间只读、只写防护的地址空间以及异常模式进行设置。安全防护模块的硬件逻辑(只读防护、只写防护逻辑)根据配置的防护空间，对系统总线的操作进行监控，当总线为写某地址，该操作的地址为配置只读，则不会将数据写入该地址，并产生操作异常，将异常信息上传，如图 4 所示。读操作以及读写操作空间范围超出配置范围，同样会产生异常并上传，如图 5 所示。空间保护的颗粒度为 byte 级，可对空间范围设置为 byte 级访问。

只读段异常保护：

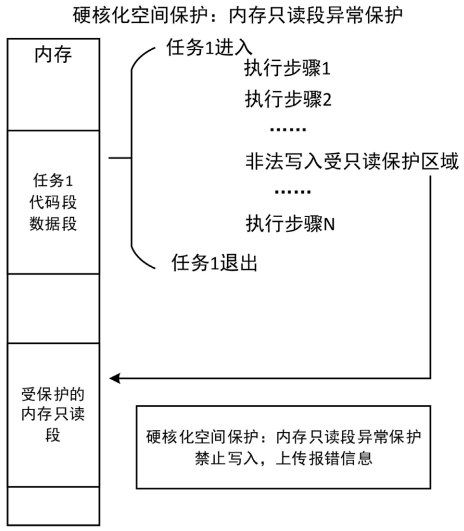

Figure 4. Read-only segment exception protection
图 4. 只读段异常保护示意图

只写段异常保护：

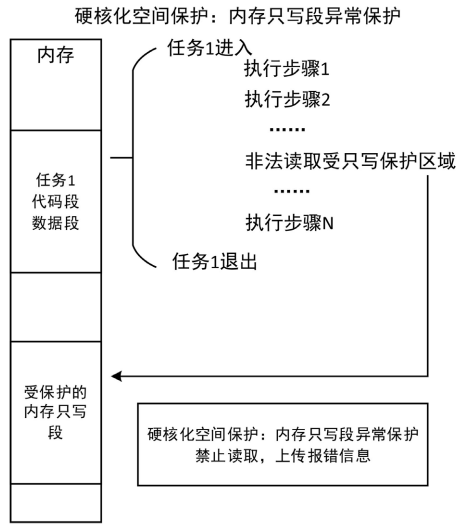

Figure 5. Write-only segment exception protection
图 5. 只写段异常保护示意图

4. 硬核化快照及恢复设计

4.1. 硬件逻辑设计

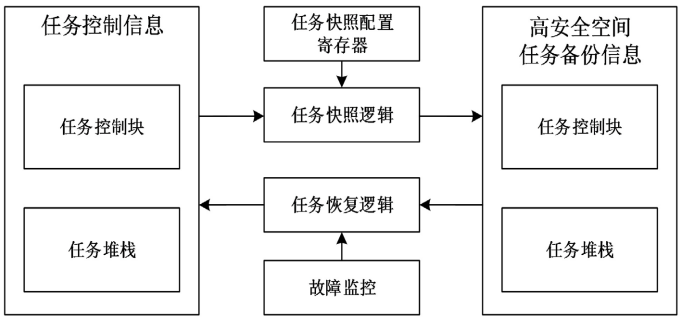

Figure 6. Task snapshot recovery logic design
图 6. 任务快照恢复逻辑设计

图 6 为任务快照恢复的硬件逻辑设计[6]，包含 6 个逻辑部分：任务快照配置寄存器、任务控制信息、高安全空间任务备份信息、任务快照逻辑、任务恢复逻辑以及故障监控逻辑。其中任务快照和恢复逻辑是整个设计的核心。

任务快照配置寄存器，是用来记录需要快照的关键任务，包括关键任务的任务控制块地址以及任务的堆栈地址。该配置寄存器需要通过软件设置，寄存器的内容可由任务快照逻辑来读取。

任务控制信息包括：任务控制块[7]以及任务堆栈。其中任务控制块保存了任务的状态信息、任务优先级[8]、任务 ID、任务栈大小、任务上下文切换指针。任务堆栈为保存任务的上下文信息，在任务切换时使用[9]。

高安全空间任务备份信息，是用于备份快照任务的任务控制块及任务堆栈信息，将关键任务的控制信息包括任务控制块及堆栈内容全部备份到高安全中间中。

任务快照逻辑是根据任务快照寄存器的状态来决定是否执行任务信息备份以及备份哪个关键任务。该快照逻辑是使用多管道并行拷贝技术(将备份地址划分为多个地址空间，每个地址空间并行备份)，大大缩减了任务快照的备份时间。

任务恢复逻辑是根据故障监控的故障状态来决定是否进行关键的任务恢复，该恢复逻辑是使用多管道并行回写技术(将恢复的地址划分为多个地址空间，每个地址空间并行恢复)，大大缩减了任务恢复时间，保障了任务恢复的实时性。

故障监控(诊断服务)是用于任务的参数异常、任务堆栈异常、任务运行时常异常的监控，当产生严重的异常时产生故障中断信号给中断控制模块。进而由软件根据异常的严重性是否进行任务的恢复。

4.2. 快照及恢复实时性设计

为实现关键任务快照及恢复的实时性[10]，尽量缩短任务的恢复时间，减少系统关键任务运行时的不稳定性。硬件逻辑采用以下方式进行了快照及恢复的加速。

硬件采用内存镜像实时存储方式，对系统内存、任务内存、事件内存和信息进行快照，同时 CPU 的通用寄存器通过锁存备份的方式进行快照。

硬件内存全部采用双端口 128 位宽 DPRAM，CPU 核心通过 DPRAM PORT A 端口实现系统的运行，快照以及恢复函通过 DPRAM PORT B 端口进行任务快照及恢复；

对硬件内存进行若干拆分，即将一个大的 SRAM 块拆分成若干个连续的小内存块，恢复时采用高频

时钟同时对多个小内存块并行恢复，进而提高恢复效率。

4.3. 可恢复流程设计

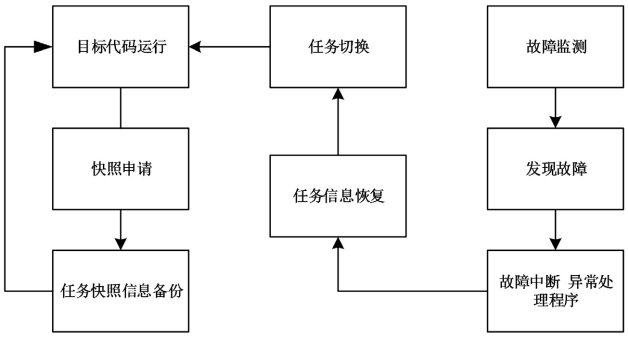

Figure 7. Software process design for task snapshot recovery
图 7. 任务快照恢复的软件流程设计

任务快照恢复的软件流程设计如图 7 所示，通过该软件流程可实现系统任务的故障监测、故障异常处理、异常任务恢复、任务的快照备份。任务的快照恢复首先在守护任务中插入快照申请函数(指定关键任务的任务 ID)；当目标代码运行到快照申请点(即需要保存的关键任务)，任务执行切换进入守护任务，该任务通知硬件完成任务信息的自动备份到高安全存储空间；当故障监测硬件监测到系统执行中的故障，进入故障中断异常处理程序，由底层硬件逻辑对备份的任务信息进行恢复，即将备份到高安全存储空间的任务信息恢复到关键任务控制块及任务堆栈[11]；最后当任务切换运行到关键任务时，该任务恢复到快照时的任务状态。

4.4. FPGA 逻辑资源

整个 PPC460 硬核化安全防护资源消耗如图 8 所示。包含了 PPC460 软核逻辑、安全防护逻辑、安全访问空间、内存快照逻辑以及快照恢复逻辑。

Resource	Utilization
LUT	415389
LUTRAM	10220
FF	264446
BRAM	382.50
DSP	6
IO	193
BUFG	22
MMCM	1
PLL	5

Figure 8. FPGA logic resource consumption
图 8. FPGA 逻辑资源消耗图

5. 安全防护验证

安全防护验证包括内存区安全防护验证以及关键任务快照及恢复验证。验证环境包括硬件平台以及软件平台。其中硬件平台是基于 VU+系列 FPGA VCU118 开发板[12]，在 FPGA 内部实现 PPC460 的软

核功能以及硬核化防护逻辑。此外该硬件平台能够实现 3 路 RS232 接口，3 路 CAN 总线接口，一个外部中断接口，一个 RTINSIGHT 测试接口，一个 JTAG 接口及复位。软件平台是基于锐华科技的开发调试环境 IDE 及 reworks 嵌入式操作系统，该操作系统支持 128 个新建任务，支持基于优先级的任务调度。

5.1. 内存区安全防护验证

首先在 VCU118 开发板平台运行 PPC460 软核硬核化安全防护逻辑，其次通过 IDE 软件环境使用 JTAG 将 reworks 嵌入式实时操作系统[13]下载到 PPC460 软核运行。通过软件分别配置好代码段空间、内存保护地址空间、堆栈地址空间，通过测试用例对分别对代码段进行写操作、内存保护空间的写空间进行读操作，对内存保护读空间进行写操作、对堆栈进行写溢出操作以及对任务进行异常卡死操作。由图 9 可以看出以上操作造成了内存监控模块 1 发生了堆栈溢出错误。通过该实验验证了该系统逻辑能够有效监控防护区的各异常操作，并对异常操作进行告警。

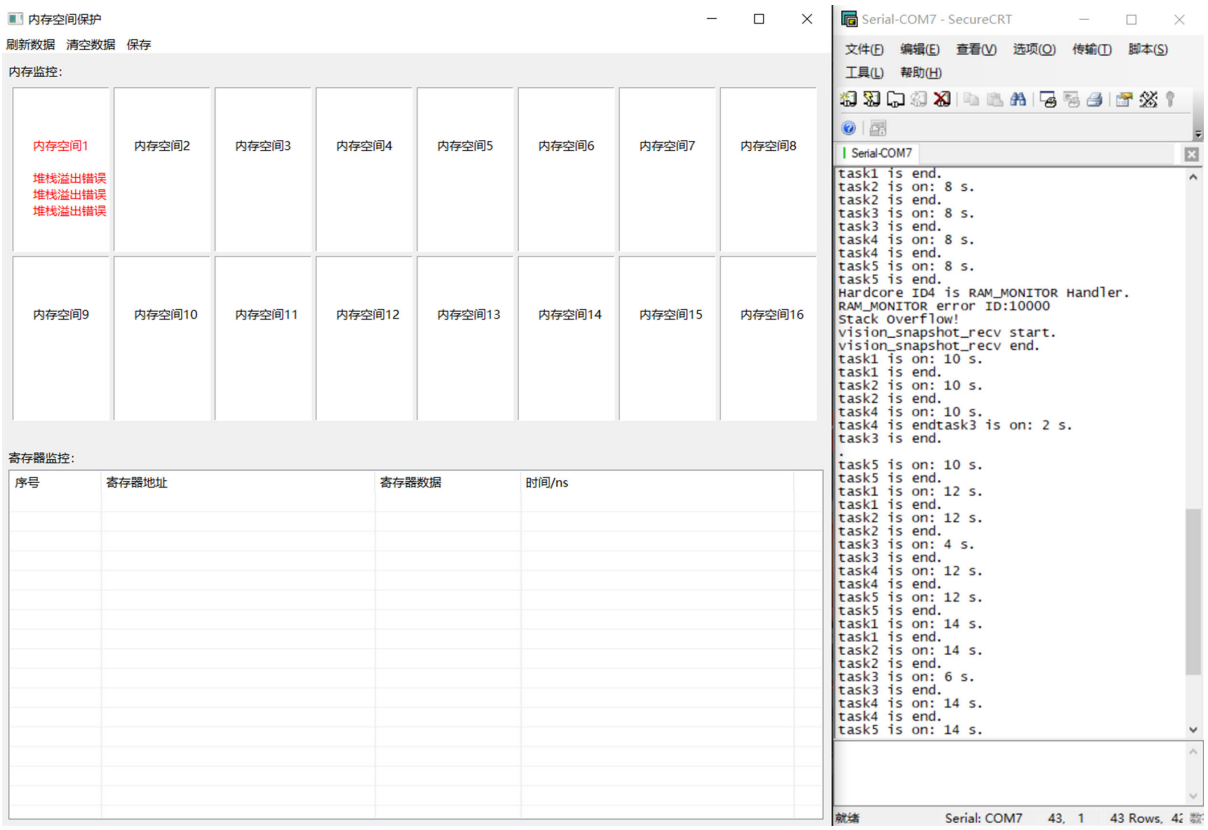

Figure 9. Memory area security protection
图 9. 内存区安全防护

5.2. 关键任务快照及恢复验证

首先在 VCU118 开发板平台运行 PPC460 软核硬核化安全防护逻辑，其次通过 IDE 软件环境使用 JTAG 将 reworks 嵌入式实时操作系统下载到 PPC460 软核运行。建立测试用例，新建 6 个测试任务，每个任务均进行打印时间信息，并假定任务 2 为关键任务，并在守护任务中设置任务快照点(假如任务 2 在运行 2 s 时进行快照)。在任务 6 中人为引入故障操作(堆栈溢出操作)，引起系统异常，并能产生异常中断[14]。图 10 为系统运行时任务进行快照的执行流程。

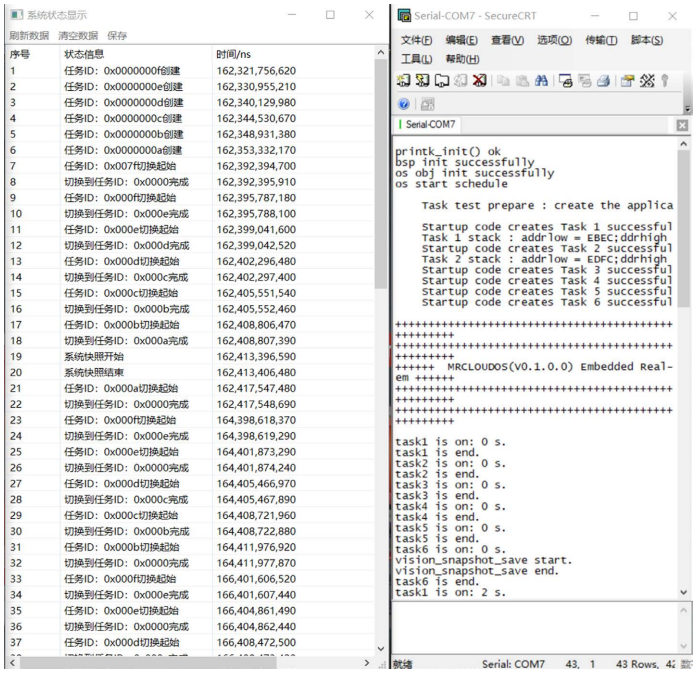

Figure 10. Key task snapshot verification
图 10. 关键任务快照验证

图 11 为系统运行后的实验结果，任务 2 能够在异常中断后进行恢复，恢复到任务快照前的状态(即运行时间为 2 s 的任务状态)，并且任务恢复时间降低到 10 μs。

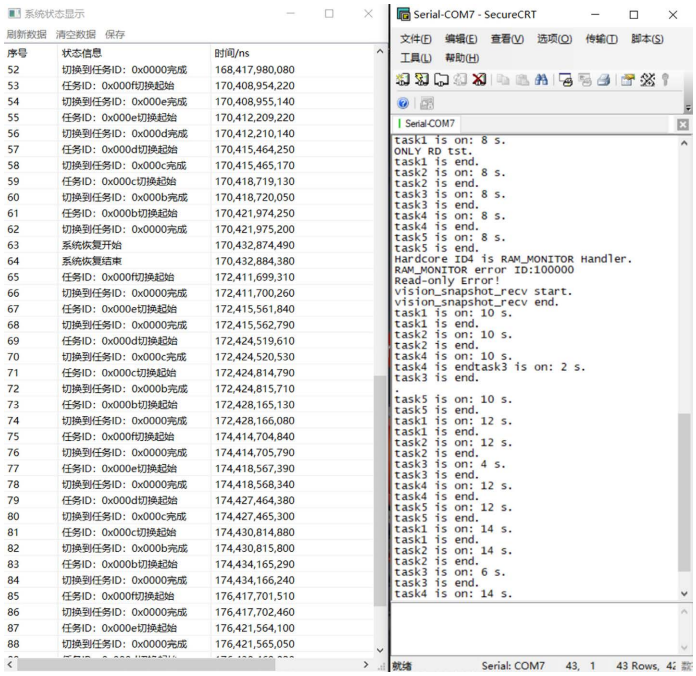

Figure 11. Critical task recovery verification
图 11. 关键任务恢复验证

通过实验验证了通过软硬件结合的任务快照及恢复，能够对指定关键任务进行快照，且快照时间为

10 μs。同时任务异常后能够恢复原任务的运行状态，并且恢复时间为 10 μs，证明了本设计的安全防护有效性。

6. 结论

本文提出一种在 PPC460 软核上增加了硬核化安全防护模块，通过该模块分别对代码空间进行防护、内存访问空间进行权限限制、堆栈空间进行安全防护。在系统出现异常访问时，能够有效避免不当的操作造成的系统运行异常，保障了系统稳定可靠的运行。

通过软硬件结合实现了任务的快照及恢复功能。在硬件逻辑上实现了硬件快照及恢复的加速，同时具有异常诊断功能。在软件上定义了守护任务，能够根据系统需要进行关键任务的快照。该技术能够在任务尤其关键任务出现异常时，进行及时恢复任务原状态，保障了系统稳定可靠的运行。本文的任务快照及恢复方法能够满足高安全、高可靠性等特定场景下的应用。

参考文献

[1] IBM (2010) PowerPC 460-S Embedded Processor Core User's Manual, 31-39. https://picture.iczhiku.com/resource/eetop/SHkequZYrYSUjcxn.pdf

[2] 张修瑞. 实时任务调度器硬件化的研究与实现[D]: [硕士学位论文]. 沈阳: 沈阳工业大学, 2023.

[3] IBM (2012) Processor Local Bus Architecture Specifications, 35-51. https://picture.iczhiku.com/resource/eetop/wyIeQjzjkwqYEVXm.pdf

[4] C*Core R&D Center (2013) PPC4601 Advance Information Revision 1.0, 5-16.

[5] 原义盈. 嵌入式软件堆栈溢出的静态测试方法研究[D]: [硕士学位论文]. 北京: 北京交通大学, 2011.

[6] 关沫, 张晓宇. 基于 FPGA 的 μC/OS-II 任务管理硬件设计[J]. 电子技术应用, 2010(2): 25-29.

[7] Jean J. Labrosse. 嵌入式实时操作系统 μC/OS-II 原理及应用[M]. 邵贝贝, 等, 译. 北京: 北京航空航天大学出版社, 2009.

[8] 崔建华, 孙红胜, 王保进. 基于 FPGA 的实时操作系统调度器硬件化设计与实现[J]. 信息技术与网络安全, 2019(38): 83-89.

[9] 王简. 实时操作系统任务调度算法的硬件化研究[D]: [硕士学位论文]. 哈尔滨: 哈尔滨理工大学, 2016.

[10] 李震, 崔骁松, 孙晨旭, 等. 基于时间和任务重要度的系统弹性恢复研究[J]. 计算机与数字工程, 2021, 49(11): 2213-2217.

[11] 徐向权. 硬件实时操作系统中任务切换技术的研究[D]: [硕士学位论文]. 青岛: 青岛大学, 2018.

[12] Xilinx (2017) HW-U1-VCU118_REV2_0_SCHEMATIC_7-14-2017, 2-75. https://docs.amd.com/v/u/en-US/VCU118-Schematics-XTP450

[13] 董巍, 马云. 基于 ReWorks 操作系统的实时多任务程序设计[J]. 中国新技术新产品, 2013(9): 53.

[14] 刘锐, 张江水, 方成. 基于 ReWorks 平台的多任务型嵌入式自主导航系统设计与实现[J]. 测绘与空间地理信息, 2015, 38(11): 53-55+65.