

北京理工大学

本科生毕业设计(论文)

指导手册

ReL4 中基于硬件加速的异步系统调用的设计 计与实现

**Design and Implementation of Asynchronous System Call
based on Hardware Acceleration in Rel4**

学 院： 计算机学院
专 业： 计算机科学与技术
班 级： 07112103
学生姓名： 王菁芃
学 号： 1120211759
指导教师： 陆慧梅

2025 年 北京理工大学教务部

北京理工大学

本科生毕业设计(论文)

任务书

ReL4 中基于硬件加速的异步系统调用的设计 与实现

**Design and Implementation of Asynchronous System Call
based on Hardware Acceleration in Rel4**

学院：计算机学院

专业：计算机科学与技术

班级：07112103

学生姓名：王菁芃

学号：1120211759

指导教师：陆慧梅

2025 年 北京理工大学教务部制

北京理工大学

本科生毕业设计（论文）任务书

学生姓名	王菁芃	学号	1120211759
学院	计算机学院	班级	07112103
专业	计算机科学与技术	题目类型	毕业设计
指导教师	陆慧梅	指导教师 所在学院	计算机学院
题目来源	结合实验室建设	题目性质	软件开发
题目	ReL4 中基于硬件加速的异步系统调用的设计与实现		
<div>一、题目内容</div> <p>ReL4 是 Rust 在 RISC-V 平台上编写的微内核，支持 seL4 的基本系统调用，并基于用户态中断机制，对 IPC 和系统调用进行了异步化改造。但由于引入了额外的运行时开销，异步 IPC 在低并发的场景下性能显著低于同步 IPC。本毕设旨在利用硬件取代异步调度器的部分功能，对异步运行时进行硬件加速，进而提升异步系统调用的性能。</p>			
<div>二、任务要求</div> <div><div>1. 在指导教师指导下阅读国内外文献和自学相关知识。学习相关背景知识，学习微内核操作系统的相关概念，了解微内核发展趋势和性能瓶颈，并深入学习和理解 ReL4 的系统调用机制，理解其中的优势和弊端。学习 TAIC 的基本设计和用法。</div><div>2. 在 ReL4 上适配 TAIC，进行性能加速。</div><div>3. 完成毕业设计（论文）外文翻译，锻炼跨文化交流的语言和书面表达能力，能就工程专业问题，在跨文化背景下进行基本沟通和交流。</div><div>4. 完成毕业设计论文并提交软件及相关文档。</div></div>			

毕业设计开发环境

- 裸机平台：qemu 模拟器或 FPGA。
- 硬件架构：RISC-V。
- 操作系统：ReL4。
- 编程语言：Rust。

三、进度安排

1. 学习并掌握 Rust 异步编程基础（1-2 周）。
2. 学习 ReL4 和 seL4 的相关论文、背景知识和熟悉相关代码。（3-4 周）。
3. 设计 TAIC 到 ReL4 的适配方案（5-6 周）。
4. 编码和调试（7-9 周）
5. 进行性能测试并分析数据。（11-13 周）
6. 完成本科生毕业设计（论文）外文翻译。（第 1 周-第 7 周）
7. 完成毕业论文，提交软件及相关文档。（第 13 周-第 14 周）
8. 完成本科生毕业设计（论文）答辩。（第 15 周）

四、主要参考文献

[1] Klein G, Elphinstone K, Heiser G, et al. seL4: Formal verification of an OS kernel[C]//Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles. 2009: 207-220.

[2] Heiser G, Elphinstone K. L4 microkernels: The lessons from 20 years of research and deployment[J]. ACM Transactions on Computer Systems (TOCS), 2016, 34(1): 1-29. [3] Klimiankou Y. Micro-CLK: returning to the asynchronicity with communication-less microkernel[C]//Proceedings of the 12th ACM SIGOPS Asia-Pacific Workshop on Systems. 2021: 106-114. [4] Heiser G. The seL4 Microkernel—An Introduction[J]. The seL4 Foundation, 2020, 1.

五、指导教师签字：

陆慧楠

2024 年 11 月 29 日

六、题目审核负责人意见

通过

签字：

宿峰

2024 年 12 月 1 日

北京理工大学

本科生毕业设计(论文)

开题报告

ReL4 中基于硬件加速的异步系统调用的设计 计与实现

**Design and Implementation of Asynchronous System Call
based on Hardware Acceleration in Rel4**

学 院： 计算机学院
专 业： 计算机科学与技术
班 级： 07112103
学生姓名： 王菁芃
学 号： 1120211759
指导教师： 陆慧梅

一、选题依据

（一）选题背景

当今计算领域对操作系统的性能、安全性和可靠性要求日益严苛，微内核技术因其固有的优势而备受关注。其中，seL4 微内核作为目前最先进的微内核之一，已被广泛应用于安全关键领域。然而，seL4 的同步系统调用和 IPC 会产生大量的特权级切换，且无法充分利用多核的性能。虽然微内核对异步通知有一定的支持，但仍需要内核进行转发，其中的特权级切换开销在某些平台和场景下将造成不可忽视的开销。

为了进一步提高系统性能和效率，ReL4 项目使用 rust 语言重写的 seL4 微内核，基于用户态中断技术改造 seL4 的通知机制，设计了无需陷入内核的异步系统调用和异步 IPC 框架，在提升用户态并发度的同时，减少特权级的切换次数。

而 ReL4 项目中，异步 IPC 的引入造成了额外的运行时开销，导致异步 IPC 在低并发的场景下性能显著低于同步 IPC。

（二）研究目的

本研究旨在针对 ReL4 项目中异步进程间通信的性能瓶颈问题，提出并实施了一种创新性的解决方案。该方案通过利用硬件资源，部分替代传统的异步调度器功能，对异步运行时进行硬件层面的加速，以期显著提升异步系统调用的性能表现。

（三）研究意义

本研究旨在提高了 ReL4 项目中异步进程间通信的性能，此研究成果可直接迁移至实际操作系统开发，以增强系统运行效率，降低资源消耗，进而提升用户交互体验。

在安全性及可靠性增强方面，本研究通过降低特权级切换频率，不仅优化了操作系统性能，亦提升了系统的安全性和可靠性。此优化对于安全关键领域，如航空航天、军事、医疗等，具有尤为重要的意义，有助于确保这些领域信息系统的安全稳定。

在技术应用与产业推动层面，本研究为微内核操作系统开发者提供了切实可行的技术路线，推动了微内核技术在多核处理器环境下的广泛应用。同时，本研究对于硬件加速技术的应用具有典范作用，有助于促进相关产业的技术进步和创新。

此外，本研究促进了操作系统与硬件设计、并发编程等领域的深度融合，为跨学科研究提供了新的研究路径和方法论。这对于促进计算机科学与其他工程学科的技术融合，具有重要的学术价值和实践意义。

（四）国内外研究现状及发展动态

目前，国内在硬件调度器领域已经有了一些进展。如关沫张晓宇采用 VHDL 语言设计了适用于硬件化的实时操作系统调度器，基于 FPGA 使用组合电路和时序电路完成了系统内核调度器的搭建。

国外学者 Y.Klimiankou 提出了一种提出了 Micro-CLK，一种基于微内核的多服务器操作系统设计，其核心思想是将进程间通信从内核中移除，从而提高效率并简化内核设计。进程间通信的优化成为微内核性能优化的重点。

目前的研究工作虽然在不同领域取得了进展，但没有将硬件调度器与异步的进程间通信与异步的系统调用结合的实例。

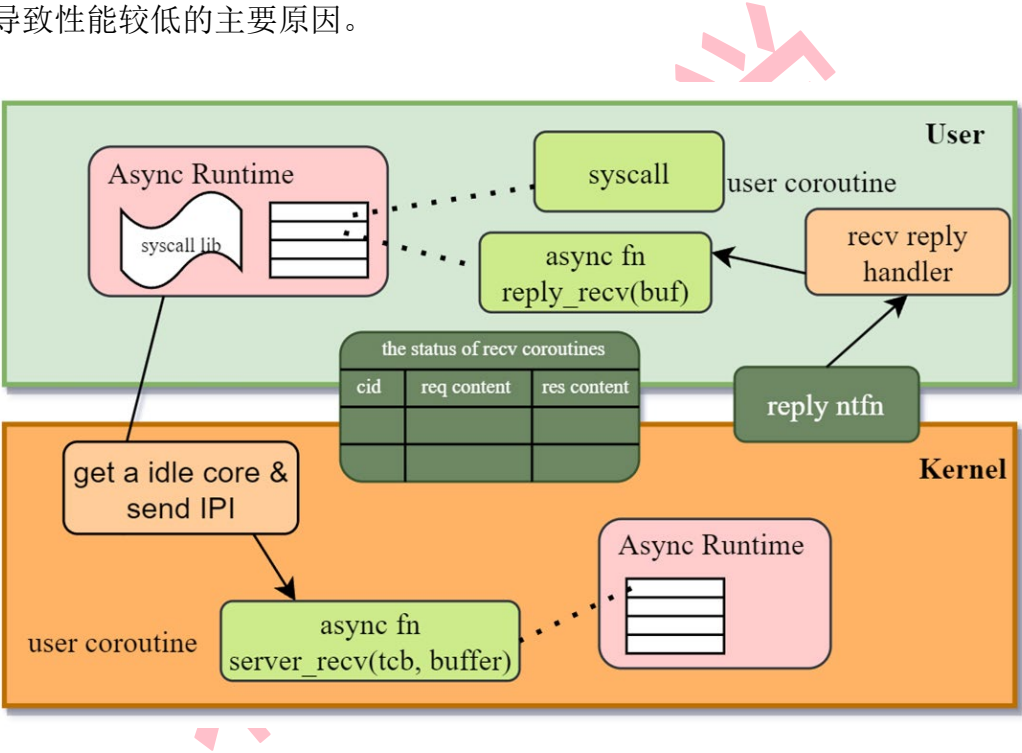
二、研究目标和内容

（研究目标、主要内容及关键问题等）

（一）研究思路

本研究利用 taic 硬件取代 Re14 系统中异步运行时的部分功能，减少异步系统调用时的调度器开销。

原有的实现中，用户态程序进行异步系统调用后，cpu 陷入内核态。系统在内核态处理时，会找到一个空闲内核，发送一个核间中断并唤醒内核态的处理协程。处理结束后，内核会向用户态程序发送一个用户态中断，在中断处理函数中唤醒回复协程和原来阻塞的发送协程。低并发场景下，频繁的系统调用和用户态中断是导致性能较低的主要原因。



本研究将异步系统调用改为 taic 实现。当用户态程序需要进行异步系统调用时，只需要操作 taic 硬件即可唤醒内核态的协程，解决了用户态中断无法发送给内核的弊端。同时，内核处理结束后，也只需要操作 taic，即可唤醒发送方的回复处理协程。

改进后的处理方法，在一次异步系统调用中，省去了至少一次用户态中断，一次中断处理函数，一次系统调用的开销。

（二）研究内容

本研究将围绕以下几个核心环节展开：

首先，本研究将对 Re14 内核中的用户态异步运行时机制以及 TAIC 硬件调度器的设计理念与具体实现进行详尽剖析，将深入挖掘其工作原理、性能特点以及接口设计，为后续的优化与改进提供理论基础。

其次，将着手进行硬件适配的异步运行时的体系结构设计与编码，主要包含以下工作：

将内核中异步运行时的队列改为 taic 实现，以适应异步系统调用。

将异步系统调用的库函数实现由系统调用改为读写 taic。

将异步系统调用回复时的用户态中断改为读写 taic，唤醒用户态对应的协程。

接下来，将在 Qemu 模拟器和 FPGA 硬件环境两种不同的平台上进行仿真与测试，以确保我们的异步运行时能够在多种场景下稳定运行。主要测试内容包括：

对改进前后的异步 ipc 和异步系统调用在不同并发度下进行测试。

对改进前后的异步 ipc 和异步系统调用低并发下不同环节的时间消耗进行测试。

最后，我将对仿真与测试的结果进行全面评估与分析，针对发现的问题进行调优。通过一系列的优化措施，旨在提升异步系统调用的性能，使其更好地适应各种应用场景。

三、研究方案

（拟采用的研究方法、技术路线、实验方案及可行性分析等）

（一）研究方法

文献调研与理论分析

通过查阅国内外相关文献，深入理解微内核架构、异步进程间通信（IPC）、硬件加速技术（如 TAIC 硬件调度器）以及 Rust 异步编程模型的理论基础。

原型实现与开发

使用 Rust 语言对 ReL4 内核进行修改，将异步运行时的队列管理和协程调度功能迁移至 TAIC 硬件实现

仿真测试与评估

在 QEMU 模拟器和 FPGA 硬件平台上分别进行仿真与测试，验证硬件加速异步 IPC 和异步系统调用的性能提升效果。

根据测试结果，量化硬件加速对异步 PC 和异步系统调用的性能提升效果。针对发现的问题进行优化，进一步提升系统的性能和稳定性。

总结与论文撰写

总结研究成果，撰写毕业设计论文，详细记录研究背景、设计思路、实现过程、实验结果及分析。

（二）技术可行性

本研究的可行性主要体现在以下三个方面：技术可行性、理论可行性、资源可行性。通过分析，本方案具备实际实现理论基础与实践条件。

基于现有 TAIC 硬件的技术特性与 ReL4 系统的模块化架构，本方案具备落地的技术基础。

TAIC 硬件通过预置的协程队列管理原语，能够直接替代用户态中断机制，其硬件级操作效率已通过 asyncOS 和 FPGA 原型验证。ReL4 内核采用的用户态-内核态异步运行时分离架构，为 TAIC 驱动的软硬件协同设计提供了标准接口适配空间，核心逻辑无需重构即可实现功能替换。此外，Qemu 模拟器与 FPGA 平台的联合仿真环境已实现对 TAIC 硬件的完整行为模拟，可系统性验证技术方案的稳定性与性能边界。

理论可行性

低并发场景下用户态中断与系统调用产生的上下文切换构成主要性能瓶颈。

TAIC 硬件通过将协程队列管理下沉至硬件层，可消除单次 IPC 中至少两次用户态中断及关联的中断处理函数调用开销。同时，硬件队列的原子性操作符合高并发场景的线性扩展需求。

资源可行性

研究所需的硬件与软件资源已具备充分保障。

TAIC 原型硬件在 FPGA 开发板完成部署，其寄存器映射与驱动接口文档完备，支持用户态与内核态的直接访问。

Re14 内核的开源代码库具有清晰的模块化结构，异步运行时相关代码（如协程调度器、IPC 通信栈）可局部替换为 TAIC 驱动逻辑，核心系统功能不受影响。

开发周期方面，90%以上的代码修改集中于用户态库函数封装与内核驱动适配层，预计总代码量变动低于 2000 行，开发与调试成本可控。

四、研究计划及进度安排

2024 年 11 月 31 日前

确定毕业论文选题或方向，报学院备案。

2025 年 1 月 1 日至 1 月 7 日

学习 Rust 异步编程基础

2025 年 1 月 7 日至 1 月 22 日

学习 ReL4 和 seL4 的相关论文、背景知识和熟悉相关代码。

2025 年 1 月 22 日至 2 月 4 日

学习 TAIC 硬件设计及使用，熟悉相关代码。

2025 年 2 月 26 日至 3 月 12 日

修改 Re14 内核中的异步运行时，适配 taic

2025 年 3 月 12 日至 3 月 26 日

修改系统调用测试函数，跑通异步系统调用测试

2025 年 3 月 24 日前

完成中期报告及文献翻译

2025 年 3 月 26 日至 4 月 7 日

在 FPGA 中进行性能测试并分析数据

2025 年 4 月 7 日至 4 月 21 日

进行调优，优化

2025 年 4 月 21 日至 5 月 5 日

完成毕业论文，提交软件及相关文档

2025 年 5 月 6 日 5 月底

对毕业论文。相关文档进行修改

2025 年 6 月初

完成本科生毕业设计（论文）答辩

五、创新点及预期研究成果

（一）学术创新点

1. 结合硬件加速和异步系统调用机制的设计

在传统微内核系统中，异步系统调用的性能瓶颈通常表现为频繁的上下文切换和内核干预。本研究创新性地提出了通过硬件加速异步系统调用的方案。

2. Rust 异步编程模型的应用

本研究通过 Rust 语言的异步编程特性实现低开销的任务调度和并发处理。Rust 提供的零成本抽象和轻量级的任务调度机制，使得在进行异步任务时不会引入过

多的运行时开销。这一特性使得在微内核中实现高效的异步系统调用成为可能。

3. 跨学科技术融合

本研究的创新点还体现在操作系统、硬件加速、并发编程等多个领域的跨学科融合。通过在硬件层面引入加速技术与异步编程模型的结合，不仅推动了操作系统设计的创新，也为硬件加速技术的应用提供了新的思路 and 方向。

（二）研究预期成果

本研究预期成果包括在 QEMU 模拟器中实现硬件加速的异步系统调用，并在 FPGA 硬件平台上进行实际部署与验证。随后，通过全面的性能验证与分析，量化硬件加速对异步 IPC 性能和异步系统调用的提升效果，特别是针对低并发场景下的性能改进。最终，本研究将形成一篇毕业设计论文，详细记录研究背景、设计思路、实现过程、实验结果及分析。

六、参考文献

[1] Klein G, Elphinstone K, Heiser G, et al. seL4: Formal verification of an OS kernel[C]//Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles. 2009: 207-220.

[2] Heiser G, Elphinstone K. L4 microkernels: The lessons from 20 years of research and deployment[J]. ACM Transactions on Computer Systems (TOCS), 2016, 34(1): 1-29.

[3] Klimiankou Y. Micro-CLK: returning to the asynchronicity with communication-less microkernel[C]//Proceedings of the 12th ACM SIGOPS Asia-Pacific Workshop on Systems. 2021: 106-114.

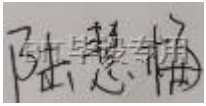
[4]关沫,张晓宇.基于 FPGA 的实时操作系统调度器硬件化设计与实现[J].信息技术与网络安全,2019,38(06):83-89.DOI:10.19358/j.issn.2096-5133.2019.06.016.

rel4 项目仓库: https://github.com/rel4team/rel4_kernel

taic 项目仓库: <https://github.com/taic-repo/taic>

七、指导教师意见


研究内容详实，方案路径合理。同意开题！

签字：
2025 年 3 月 1 日

成绩：优 (A+)，占比：0.00%

八、开题审核负责人意见

通过

签字：
2025 年 3 月 2 日

北京理工大学

本科生毕业设计(论文)

中期报告

ReL4 中基于硬件加速的异步系统调用的设计 计与实现

Design and Implementation of Asynchronous System Call based on Hardware Acceleration in Rel4

学 院： 计算机学院

专 业： 计算机科学与技术

班 级： 07112103

学生姓名： 王菁芃

学 号： 1120211759

指导教师： 陆慧梅

一、毕业设计（论文）主要研究内容、进展情况及取得成果

（一）目前进度总体完成情况

- ☒ 学习 Rust 异步编程基础
- ☒ 学习 ReL4 和 seL4 的相关论文、背景知识和熟悉相关代码。
- ☒ 学习 TAIC 硬件设计及使用，熟悉相关代码。
- ☒ 修改 ReL4 内核中的异步运行时，适配 taic
- ☒ 修改系统调用测试函数，跑通异步系统调用测试
- ☐ 在 FPGA 中分析 taic 对 rel4 异步功能的性能提升（在下阶段中进行）
- ☐ 完成毕业论文，提交软件及相关文档（在下阶段中进行）

（二）主要研究内容综述

1. Rust 异步编程的学习，rel4/taic 项目复现

在前期准备阶段，我从官方文档、社区博客入手，学习了 Rust 语言中的异步编程模型，熟悉其底层的 Future、Waker 等核心机制。重点关注了 Future 的生命周期管理、Pin 与 Unpin trait 的使用规范，以及任务调度中 Waker 机制的注册与触发方式。

同时，我分析了 ReL4 操作系统中的异步任务调度实现。其通过在用户态和内核态引入异步运行时，实现了在 seL4 系统中实现异步编程。其异步通信模型使用了用户态中断机制，通过系统调用与事件队列相结合，实现了兼容 seL4 的异步通知机制。其中 seL4_Signal/seL4_Send 会向目标发送用户态中断以唤醒目标协程。seL4_Wait 基于异步运行时和用户态中断实现。seL4_Wait 会在该协程中阻塞并等待的用户态中断，当用户态中断到来时由异步运行时调度该协程，当实现非阻塞的等待通知。

随后，我在本地对两个项目的环境进行了构建，对现有的 ReL4 项目和 Taic 的项

目成果进行了复现。通过对代码的走读，深入理解了原 seL4 项目中的能力机制与通知机制，部分内核对象以及它们在微内核权限管理与进程间通信中的角色。与此同时，我也系统性地梳理了 ReL4 中用户态异步运行时的结构设计，包括任务生成、就绪队列管理、唤醒机制以及协程状态维护等核心部分，进一步明确了本项目中需要改造的目标模块与技术要点。

2. 兼容 TAIC 硬件调度器的异步系统调用

在完成 ReL4 与 TAIC 硬件项目的基础环境搭建与理解之后，我开始着手对 ReL4 内核中的异步运行时进行实质性的改造，目标是实现 TAIC 硬件调度器对异步任务调度流程的接入。ReL4 原本的异步运行时主要依赖软件层面维护的就绪队列来管理协程的调度与唤醒，该结构虽然逻辑清晰，但在高并发场景下容易成为性能瓶颈。因此，为了充分利用 TAIC 硬件提供的异步调度能力，我将异步运行时的就绪队列（ready queue）与协程唤醒逻辑进行了重构和替换。

首先，为了更方面的在内核中使用 TAIC，我将硬件驱动 `taic_driver` 引入内核并基于原有的接口进行了简单封装。`taic_driver` 在本项目中作为内核与硬件调度器通信职责的重要中间层，任务入队、出队、通道配置、中断控制等功能接口。

其次，在就绪队列部分，我移除了原本由 Rust 容器结构或实现的就绪队列，改为 `taic_driver` 中抽象的 `LocalQueue` 对象。通过对该对象读写方法的调用，可以访问 TAIC 硬件内部维护的调度队列资源。在这一基础上，我将原有的异步运行时中执行器里生成协程的 `spawn` 方法改为使用 `taic_driver` 中的 `task_enqueue` 实现、获取就绪队列的 `fetch` 方法中改为 `task_dequeue` 实现。此外对队列的分配，协程序号分配等细节进行适配修改。以此替代原有的任务插入与分发逻辑，使得协程调度动作可以在硬件层实现，从而降低内核开销并提升调度响应速度。此外，我对异步系统调用的注册逻辑进行了修改为相应的硬件操作以适配新的内核协程生成模式。

3. 异步系统调用测试

在完成 ReL4 异步运行时向 TAIC 硬件调度器的适配改造之后，原有的异步系统调用测试框架已无法完整覆盖或验证当前的新机制。因此，我对测试模块也进行了系统性的修改，确保整个异步系统调用流程能在新架构下正常运行，并具备可观测试性与可复现性。

首先，对进程初始化时，通知注册、缓冲区注册、异步系统调用注册等初始化代码进行了改写，使之符合当前异步系统调用逻辑。其次，修改了异步库中的若干辅助函数，如 `seL4_call` 中原有的陷入内核的实现改成了在用户态发送硬件信号，实现了无需陷入内核的异步系统调用。此外，我还修改了进行异步系统调用的协程函数，实现了 TAIC 调度队列中发送方和接收方的注册。

4. 阻塞队列的缓存数量限制的解决方案

在实现过程中，遇到了硬件中阻塞队列的缓存数量存在限制的问题。协程的唤醒需要使用一个中断向量，硬件调度器的中断向量数量只有 32 个，因此实际硬件中的阻塞队列的缓存数量仅为 32。

因此，需要采用一种复用的方式解决该问题。使用 TAIC 的用户态程序需要记录中断向量的使用情况，以分配中断向量给不同的协程。中断向量表中常驻一个 0 号协程(dispatcher 协程)。客户端执行异步 IPC 的协程在发送通知前需要先申请一个中断向量。如果此时中断向量表未满时，则该协程会拿到一个中断向量，并将该中断向量写入 IPCItem 中传递给服务端。服务端收到异步 IPC 后，根据 IPCItem 中的中断向量唤醒阻塞的客户端协程。如果此时中断向量表已满，则该协程申请不到中断向量，则将 IPCItem 中的向量置 0。服务端收到异步 IPC 后，如果 IPCItem 中的中断向量为 0，则唤醒 0 号 dispatcher 协程。dispatcher 协程根据 IPCItem 中的 cid，唤醒对应的协程。

具体实现上，我更改了原有 IPC_Item 的结构声明，加入了 vec 字段用于记录发送协程所申请的中断向量。然后，修改了的原 IPC_Buffer 的部分结构，使之可以进行固定位置的读写，以满足协程绕过分发直接唤醒的需求。最后，我修改了原调用方，内核中的异步系统调用处理函数，dispatcher 协程的逻辑。

5. 多次注册模式导致的性能问题

原有的模式中，TAIC 信号接收方的注册是单次可用的，即每次接收完协程唤醒的信号后，需要重新对队列进行注册。多次注册导致高并发场景下的性能开销有所增大，因此我尝试将单次注册的逻辑改为重复注册，并尝试对比两种方案的优点与可行性。

二、存在的问题和拟解决方案

在当前阶段的研究与开发过程中，尽管已完成 ReL4 内核中异步运行时与 TAIC 硬件调度器的初步集成，但在系统运行、测试验证和性能评估方面仍然暴露出若干关键问题，需要在后续工作中予以优化与解决：

1. 部分测试用例选取不够恰当，不能很好地反映异步所带来的性能提升

问题描述：

当前所进行的性能测试排除了异步协程阻塞等待中的时间，这样的测试方式不能很好的反应异步带来的吞吐率提升。

拟解决方案:

重新选取测试指标并编写测试用例，在不同并发场景下，测试异步 IPC 和异步系统调用所用的总时间，以反应吞吐率的提升情况。

2. 当前实现仅在 QEMU 的模拟环境中得以验证，结果可靠性有待验证

问题描述:

由于开发初期以 QEMU 虚拟机为主要运行环境，便于快速调试与功能验证，目前所有修改与测试均在该虚拟化环境中进行。然而 QEMU 对硬件行为的模拟存在一定的抽象和延迟，部分行为无法真实还原 TAIC 硬件在实际运行中的行为。因此，一些基于性能优化的假设和调度逻辑仍缺乏真实环境下的验证。

拟解决方案:

计划在后续阶段将目前的系统部署至实际的 FPGA 开发板上运行测试，进一步验证系统设计的正确性与可行性。

三、下一步研究任务与进度安排

在完成前期异步运行时与 TAIC 硬件调度器的集成、异步系统调用的适配和功能测试之后，下一阶段的工作将重点转向可靠性验证、性能测试及毕业论文的撰写与整理。具体安排如下：

(一) 性能评估与测试验证（4 月中旬 - 5 月初）

1. 在 FPGA 平台部署当前系统

将当前已完成的 ReL4 + TAIC 集成系统烧录至 FPGA 开发板，测试其在真实硬件平台上的运行效果，验证系统稳定性和调度的正确性。

2. 丰富测试覆盖面

覆盖不同负载强度和并发模式下的异步 IPC、系统调用，进一步对比异步

与同步机制在吞吐量、响应延迟方面的差异。

（二）功能与兼容性完善（4月中旬 - 5月初）

- 1. 针对异步系统调用处理的延迟较高问题进行改进

（三） 毕业论文撰写与系统整理（5月初 - 5月中旬）

- 1. 撰写毕业论文初稿
- 2. 整理代码与文档，将所有项目源码、测试脚本进整理并附带注释，准备提交材料

（四）预留收尾与论文修改时间（5月中旬 - 5月下旬）

- 1. 根据指导老师的反馈意见进行论文修改
- 2. 准备毕设答辩相关材料

四、指导教师意见


进展正常，工作有成效。继续努力，争取好结果！

签字：

2025年4月15日

成绩：优(A+)， 占比： 0.00%

五、中期审核负责人意见

签字:  审核专用
2025 年 4 月 16 日

北京理工大学

毕业设计（论文）指导教师评语表

指导教师对毕业设计（论文）的评语：

本文针对微内核ReL4在低并发场景中的性能问题，提出一种基于TAIC(任务感知中断控制器)的异步系统调用优化方案。该方案通过引入硬件加速异步任务调度，避免了传统实现中的特权级切换和中断触发操作，从而降低了异步系统调用的平均延迟。实验结果表明，该方案在维持高并发性能优势的同时，对低并发场景下的平均系统调用响应时间有显著降低，提升了系统整体的响应能力与资源利用效率。

王菁芃同学很好地完成了毕业设计的任务，完成毕业设计的任务过程中展现了扎实的专业素养、独立钻研的工作能力和不断创新的精神。论文结构合理，引用规范，设计方案恰当，有一定工作难度，过程规范。论文达到本科毕业设计对应的毕业要求，同意参加论文答辩。

是否有校外指导教师：☐是 ☒否；若选择“是”，校外指导教师对毕业设计（论文）的评语：

成绩：优(A+)，占比 0.00%

指导教师

陆慧楠

2025年5月28日

毕业设计（论文）匿名评阅评语表 1

学生姓名	王菁芃	学号	1120211759
学院	计算机学院	班级	07112103
专业	计算机科学与技术		
指导教师	陆慧梅	指导教师 所在学院	计算机学院
题目	ReL4 中基于硬件加速的异步系统调用的设计与实现		
评阅结果	良 (A)， 占比 0.00%		
<p>评语：</p> <p>选题针对微内核与异步通信机制问题，具有一定的研究意义，符合复杂工程问题要求。论文分析了相关领域研究现状。</p> <p>论文提出了提出一种基于 TAIC 的异步系统调用优化方案。该方案通过引入硬件加速异步任务调度，避免了传统实现中的特权级切换和中断触发操作，从根本上降低了异步系统调用的平均延迟。论文实验数据详实，结论可信。</p> <p>论文结构合理，引用规范，研究方案恰当，有一定难度，过程规范，表明学生具有较为扎实的专业基础知识和综合运用能力，已基本具备独立研究能力，论文达到本科毕业设计对应的毕业要求，同意参加论文答辩。</p> <p>评阅人：</p>			

北京理工大学本科生毕业设计（论文）评语表

2025 年 5 月 22 日

北京理工大学

毕业设计（论文）匿名评阅评语表 2

学生姓名	王菁芃	学号	1120211759
学院	计算机学院	班级	07112103
专业	计算机科学与技术		
指导教师	陆慧梅	指导教师 所在学院	计算机学院
题目	ReL4 中基于硬件加速的异步系统调用的设计与实现		
评阅结果	良 (A)， 占比 0.00%		
<p>评语：</p> <p>本文针对进程间通信问题提出一种基于 TAIC(任务感知中断控制器)的异步系统调用优化方案。该方案通过引入硬件加速异步任务调度，避免了传统实现中的特权级切换和中断触发操作，从根本上降低了异步系统调用的平均延迟。文章进行了大量的实验评估，证明了所提系统的有效性与鲁棒性。整篇论文选题新颖、难度适中，技术路线明确，论证充分，实验设计合理，数据可信，引用文献相对规范，排版严谨，体现出作者扎实的专业基础与较强的综合工程实践能力，已基本具备独立完成科研与工程项目的的能力，同意参加本科论文答辩。</p> <p>1. 在第一章中，凡首次出现的缩写均应给出英文全称及中文含义，例如“TAIC（Task-Aware Interrupt Controller，任务感知中断控制器）”。请仔细核查全文，确保同类缩写一律按此规范处理。</p>			

2. 图 3-1、图 3-2 的语言统一。“异步系统调用流程图”中的所有文字说明请从英文改为中文，符号与排版风格与正文保持一致，如字体大小等。

3. 第 4.1 节的算法描述。算法 1：请在伪代码开头明确列出输入(Input)与输出 (Output)，并与正文中对算法的叙述相呼应。算法 2 与算法 3：将算法标题及伪代码中的注释、步骤说明全部改为中文。

4. 第 5 章的 ReL4 书写与图片排版。ReL4 系统在全文应保持一致的大小写与格式（建议统一写作“ReL4”）。此外，第 5 章插图请使用居中对齐，并在图注中注明图号及简要说明，避免随段落漂移。


5. 目前图 5-6 实验仅给出了并发度为 4 的性能结果。建议补充并发度更高（如 8、16、32 等）的测试数据，或在正文中讨论并发度提升可能带来的瓶颈。

6. 现有参考文献数量偏少，建议结合以下方向扩展。如任务感知中断控制器（TAIC）及相关中断管理策略的最新研究。

评阅人：

2025 年 5 月 22 日

毕业设计（论文）答辩评语表

答辩委员会（小组）组长			
姓名		职称	签字
余皓然		预聘副教授	
答辩委员会（小组）组员	姓名	职称	签字
	陆慧梅	副教授	
	余皓然	预聘副教授	
	段春晖	预聘副教授	
	黎有琦	助理教授/特别副研究员	
	韩峰	工程师	

答辩中提出的主要问题及回答的简要情况：

问 1：“中断向量的动态分配策略”中，“动态”主要体现在哪个方面？

答 1：固定分配策略中，协程号和中断向量号是持续绑定的，例如 2 号协程始终使用 2 号向量。但是这种方式，在协程长时间等待时会浪费系统资源。动态的分配机制下，协程号在使用时申请，用完释放，不存在持续的绑定关系。

问 2:文中有些地方的图片位置与描述不符，参考文献数量较少。

答 2：调整文档图片格式，补充参考文献引用。

北京理工大学本科生毕业设计（论文）评语表

问 3: 一般性能的优化会带来一些副作用, 导致另一方面存在劣势, 这一方面是否有考虑。

答 3: 这个目前还没有考虑。索引分配和中断向量的开销均较小, 即使在高并发场景下也可忽略。本研究系统的主要开销主要在于硬件的引入, 硬件可以代替内核中部分的工作, 以实现性能的加速, 但需要而外增加硬件接口。

答辩委员会（小组）代表（签字）：

余路然

2025 年 5 月 30 日

答辩委员会（小组）的评语：

论文选题符合复杂工程问题标准和 OBE 中毕业设计的毕业要求，论文针对 ReL4 操作系统在低并发场景下异步系统调用性能不足的问题，提出了提出了一种基于任务感知中断控制器（TAIC）的优化方案，通过对系统调用路径中的调度机制与上下文切换开销进行分析，设计并实现了无需陷入内核的异步系统调用框架，重点包括中断向量的动态映射机制、缓冲区结构的调整及协程调度方式的重构。实验数据详实，结论可信。论文阐述清楚，撰写符合规范，结构合理，达到本科水平。答辩中问题回答正确。软件验收运行较好。表明该生已较好掌握计算机的基础知识和理论，达到了毕业设计对应的毕业要求的各项指标点要求，完成了毕业设计任务，答辩委员会一致同意通过论文答辩。

答辩委员会（小组）代表（签字）：

余路翊

2025 年 5 月 30 日

答辩委员会（小组）给定的成绩：良(A)，占比 100.00%

答辩委员会（小组）主任（签字）：

余路翊

2025 年 5 月 30 日

北京理工大学本科生毕业设计（论文）评语表

最终成绩：良(A)

成绩构成及占比：

开题成绩：优(A+)，占比：0.00%

中期成绩：优(A+)，占比：0.00%

外文翻译成绩：无权查看，占比：无权查看

指导教师评阅成绩：优(A+)，占比：0.00%

匿名评阅成绩：良(A)，占比：0.00%

答辩成绩：良(A)，占比：100.00%

指导教师签字：



2025 年 5 月 30 日

责任教授签字：



2025 年 5 月 30 日

毕业设计（论文）开始日期 2024 年 11 月 29 日

截止日期 2025 年 6 月 4 日

毕业设计（论文）答辩日期 2025 年 5 月 30 日

附件：

北京理工大学本科生毕业设计（论文）

书写规范及打印装订要求

一、毕业设计（论文）书写规范

1. 文档格式要求

（1）页面设置

A4 纸纵向打印；

页边距：上 3.5cm、下 2.6cm、左 3cm、右 2.6cm；

页眉：2.4cm、页脚：2cm、装订线：0cm；

页眉内容为：“北京理工大学本科生毕业设计（论文）” 页眉，宋体、四号，居中排列，字间距加宽 0.5 磅；

页脚内容为页码，宋体、五号，居中排列。

（2）文档格式

一级标题：黑体，三号，加粗；间距：段前 0.5 行，段后 1 行；

二级标题：黑体，四号，加粗；间距：段前 0.5 行，段后 0 行；

三级标题：黑体、小四、加粗；间距：段前 0.5 行，段后 0 行；

标题行距：1.5 倍行距；

正文部分：宋体、小四；正文行距：22 磅；间距段前段后均为 0 行。

2. 标题序号标号

一级标题：第 1 章第 2 章第 3 章

二级标题：1.1 1.2 1.3

三级标题：1.1.1 1.1.2 1.1.3

正文中一级标题居中，二、三级标题居左对齐；目录中一级标题居左对齐，下一级标题依次向右缩进一格。

3. 图、表标号

采用阿拉伯数字按章依序编码。

图 1-1 图 1-2 图 2-1 图 2-2

表 1-1 表 1-2 表 2-1 表 2-2

图、表居中，图注标在图下方，表头标在表上方，宋体、五号、居中，图表与上下文之间各空一行。

4. 公式标号

采用阿拉伯数字按章依序编码。

(1-1) (1-2) (2-2) (2-2)

公式标注应于该公式所在行的最右侧。对于较长的公式只可在符号处（+、-、*、/、 \leq 、 \geq 等）转行。在文中引用公式时，在标号前加“式”，如式（1-2）。

5. 参考文献书写规范

参考国标【GB/T 7714—2015】，参考文献书写规范如下：

(1) 文献类型和标识代码

普通图书：M	会议录：C	汇编：G	报纸：N
期刊：J	学位论文：D	报告：R	标准：S
专利：P	数据库：DB	计算机程序：CP	电子公告：EB
档案：A	舆图：CM	数据集：DS	其他：Z

(2) 不同类别文献书写规范要求

期刊

[序号] 主要责任者. 文献题名 [J]. 刊名, 出版年份, 卷号(期号): 起止页码.

[1] 袁庆龙, 候文义. Ni-P 合金镀层组织形貌及显微硬度研究 [J]. XX 理工大学学报, 2001, 32(1): 51-53.

普通图书

[序号] 主要责任者. 文献题名 [M]. 出版地: 出版者, 出版年. 起止页码.

[2] 刘国钧, 郑如斯. 中国书的故事 [MM]. 北京: 中国青年出版社, 1979. 80-115.

会议论文集

[序号] 析出责任者. 析出题名[A]. 见(英文用 In): 主编. 论文集名[C]. (供选择项: 会议名, 会址, 开会年)出版地: 出版者, 出版年. 起止页码.

[3] 孙品一. 高校学报编辑工作现代化特征 [A]. 见: 张为民编. 中国高等学校自然科学学报研究会. 科技编辑学论文集(2)[C]. 北京: 北京师范大学出版社, 1998. 10-22.

专著中析出的文献

[序号] 析出责任者. 析出题名[A]. 见(英文用 In): 专著责任者. 书名[M]. 出版地: 出版者, 出版年. 起止页码. [4] 罗云. 安全科学理论体系的发展及趋势探讨[A]. 见: 白春华, 何学秋, 吴宗之. 21 世纪安全科学与技术的发展趋势[M]. 北京: 科学出版社, 2000. 1-5.

学位论文

[序号] 主要责任者. 文献题名 [D]. 保存地: 保存单位, 年份. [5] 张和生. 嵌入式单片机系统设计 [D]. 北京: 北京理工大学, 1998.

报告

[序号] 主要责任者. 文献题名 [R]. 报告地: 报告会主办单位, 年份. [6] 冯西桥. 核反应堆压力容器的 LBB 分析 [R]. 北京: 清华大学核能技术设计研究院, 1997.

专利文献

[序号] 专利所有者. 专利题名 [P]. 专利国别: 专利号, 发布日期. [5] 姜锡洲. 一种温热外敷药制备方案 [P]. 中国专利: 881056078, 1983-08-12.

国际、国家标准

[序号] 标准代号. 标准名称 [S]. 出版地: 出版者, 出版年. [1] GB/T 16159—1996. 汉语拼音正词法基本规则 [S]. 北京: 中国标准出版社, 1996.

报纸文章

[序号] 主要责任者. 文献题名 [N]. 报纸名, 出版年, 月(日): 版次. [5] 谢希德. 创造学习的思路[N]. 人民日报, 1998, 12(25): 10.

电子文献

[序号] 主要责任者. 电子文献题名 [文献类型/载体类型]. 电子文献的出版或可获得地址(电子文献地址用文字表述), 发表或更新日期/引用日期(任选). [21] 姚伯元. 毕业设计(论文)规范化管理与培养学生综合素质 [EB/OL]. 中国高等教育网

教学研究，2005-2-2.

关于参考文献的未尽事项可参考国家标准《信息与文献参考文献著录规则》（GB/T 7714—2015）

6. 附录

附录是论文主体的补充项目，为了体现整篇论文的完整性，写入正文又可能有损于论文的条理性、逻辑性和精炼性，这些材料可以写入附录段，但对于每一篇论文并不是必须的。主要包括以下几类：

（1）比正文更为详尽的理论根据、研究方法和技术要点，建议可以阅读的参考文献的题录，对了解正文内容有用的补充信息等；

（2）由于篇幅过长或取材于复制品而不宜写入正文的材料；

（3）一般读者并非必要阅读，但对本专业同行很有参考价值的资料；

（4）某些重要的原始数据、数学推导、计算程序、框图、结构图、统计表、计算机打印输出件等。

附录依次用大写正体英文字母 A、B、C……编序号，如附录 A，附录 B。

附录中的图、表、公式、参考文献等另行用阿拉伯数字编序号，与正文分开，也一律用阿拉伯数字编码，但在数码前冠以附录序码，如：图 A-1；表 B-2；式(B-3)；文献[A-4]等。

二、毕业设计（论文）资料的装订

毕业设计（论文）按以下标准统一装订：毕业设计（论文）封面→原创性声明、关于使用授权的声明→中外文摘要→目录→毕业设计（论文）正文→结论→参考文献→附录→致谢。

毕业设计（论文）指导手册按以下标准统一装订：毕业设计（论文）指导手册封面→毕业设计（论文）任务书→毕业设计（论文）开题报告→毕业设计（论文）中期报告→毕业设计（论文）评语表。指导手册包括但不限于以上内容，可按照学院细则增加材料。

三、外文翻译

外文翻译 5000 字左右，内容与毕业设计（论文）题目有关。外文原文用 A4 纸

北京理工大学本科生毕业设计（论文）书写规范及打印装订要求

打印或复印，中文翻译用 A4 纸打印，文档格式要求参考“毕业设计（论文）文档格式要求”。装订顺序为：外文翻译封面→外文原文→中文翻译。

北京理工大学