




# 卢炯嘉

avimitin@gmail.com ·  <https://github.com/Avimitin> ·  <https://crates.io/users/avimitin> ·  <https://sh1mar.in>

## 工作经历

Arch Linux RISC-V 小队, 远程

2022 年 3 月 - 2023 年 5 月

中科院软件研究所, 操作系统开发工程师

- 积极贡献 Arch Linux RISC-V 软件包支持项目, 提交 PR 189 个, 在 RISC-V 平台软件生态上做出大量贡献。
- 独立负责 Arch Linux RISC-V 测试与验证, 巩固并扩展其基础设施架构, 搭建自动化构建测试环境, 提升系统稳定性。
- 积极推动 RISC-V 平台相关修改提交上游, 与社区保持密切沟通, 为 RISC-V 生态的广泛支持做出重要贡献。

Buddy MLIR, 远程

2023 年 5 月 - 2024 年 5 月

中科院软件研究所, RVV 方言和基础设施维护工程师

- 设计并创建对 RISC-V Vector 机器的底层系统基础设施, 成功实现并验证在裸机上的 PyTorch 到 RVV 的端到端验证。
- 利用 Buddy MLIR 的编译器工具链, 成功在裸机 RVV 上部署并执行 Bert ML 模型, 展现 Buddy MLIR 工具链的后端支持灵活性。
- 撰写详细的 MLIR Sparse Tensor 实现, 并提供大量简单清晰的使用范例, 帮助更多开发者理解和运用 MLIR 基础设施。
- 开发并维护了一套可复现, 可拓展的编译基础设施, 帮助用户更好的构建 Buddy MLIR 工具链。

晶圆编译器, 远程


2023 年 5 月 - 至今

中科院软件研究所, 密码学方言工程师

- 设计并实现 MD5, SM3 密码学方言, 为用户提供一层高抽象封装的密码学算子, 简化对摘要操作的调用。
- 用 MLIR 实现在 CPU/GPU 平台的密码学的同步执行, 提高在多独立数据密码学计算时的并行度和计算效率。

## 项目经历


chipsalliance/t1, RISC-V 向量 IP

 [chipsalliance/t1](https://github.com/chipsalliance/t1)

- CI/CD 基础设施管理: 主要负责 T1 项目 CI/CD 基础设施的开发与维护, 确保服务部署、服务器运行持续稳定运行。
- 仿真测试自动化: 领导 T1 项目仿真测试自动化框架的创建, 推动日常回归测试和后端验证, 以保证 RTL 可靠性。
- 性能分析框架: 为 T1 项目开发了早期基于 Probe 的 TestBench 模块, 提供的性能分析和检查框架, 用于帮助编译器团队识别瓶颈并优化代码。
- rocket-chip 开发: 开发了 rocket-chip DPI 和 verilator 仿真器, 为 T1 项目与 Rocket Chip 生态系统的集成开发了基础实现。
- 测试环境开发: 开发并引入如 newlibc 等库进入 T1 项目, 支持更多向量化测例的运行。

chipsalliance/t1, RISC-V 向量 IP

 [chipsalliance/t1](https://github.com/chipsalliance/t1)

- 体系结构: 使用 RISC-V opcodes 做代码生成以轻松用  [remns-project/sail](https://github.com/remns-project/sail) 实现 RISC-V 体系结构描述。
- 仿真模拟器: 自主设计和维护了一套 C-API 及 Rust 实现, 将 RISC-V 体系结构描述实现成可执行的模拟器。
- 对比测试: 实现了一套用于对比体系结构状态的框架, 保证体系结构描述的正确性。

## 技能

- 编程语言: 泛语言 (编程不受特定语言限制), 且尤其熟悉 Rust Scala Ruby Lua TypeScript Nix, 较为熟悉 OCaml Haskell Zig C++ (排名均不分先后)。
- 熟练的 Neovim 用户, 拥有一个约 270 stars 的配置项目, 给如 [lazygit.nvim](https://github.com/lazygit/nvim) 等千星 neovim 插件做过贡献, 能够快速为团队定制一份好用的编辑器开发环境。
- Linux 开发: 非常熟悉发行版的工作机制, 熟知在发行版上各种工作流的最佳实践, 能够独立且熟练的寻找, 测试并解决操作系统运行时遇到的问题。日常利用 NixOS Module 和 systemd 维护基于 NFS 和 netboot 的规模中等的计算集群。在日常实践中也能够熟练使用 docker 部署服务。
- Rust 开发: 熟悉 Rust 社区, 日常使用 Rust 开发程序。熟知 Rust 实现上的细节, 比如 Async Coroutine, Higher Kinded Types, 在我的个人项目 [deepl-rs](https://github.com/deepl-rs) 中都有所展现。

- Nix 开发：在使用 Nix 打包和 reproducible build 上有丰富的经历，非常熟悉 Nixpkgs 的状态，库的实现和软件包的打包方式。使用 Nix Flake 为 [chipsalliance/riscv-vector-tests](#), [chipsalliance/t1](#) 等项目 维护过大型的如 LLVM, libc 等依赖。能够快速理解项目需求并迅速搭建需要的编译和调试运行环境。
- 前端开发：使用 React 和 Vite 等技术栈，开发过两到三个传统 SSR 和现代 CSR 的前端应用。使用 Diesel 等 ORM 开发过后端项目。
- 开发工具：主要使用 Neovim 在 Linux 环境开发，能适应任何常见编辑器/操作系统，有使用 GitHub、Gitee、Slack、GitLab 等团队协作工具的经验。