欢迎参加龙架构双周会

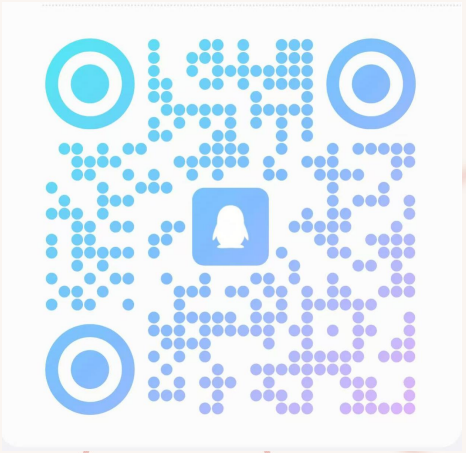
- **编辑权限申请**
  - 计划好主讲的议题和大致用时
  - 在本文档申请编辑权限且附上简短的申请理由
  - 在龙架构双周会交流群中 **@群主** 或 **管理员** 获取权限
  - 向 loongarch@whlug.cn 发送主题为龙架构双周会报告的邮件
    - 邮件内请简要说明您将要报告的内容，我们将在收到邮件后同您取得联系，为您提供文档的编辑权限
- **内容编辑**
  - 请在对应的议题版块下添加您想要分享的内容
  - 若无对应议题，请直接在幻灯片其他议题最前方添加
  - 快速报告一页控制在 3 分钟以内，报告期间请勿讨论发言
  - 专题报告 15~30 分钟，分享结束后可讨论交流

龙架构 LoongArch Biweekly 双周会

本页预定讲者

龙友会 (Loong Meetup)
龙架构双周会・第 15 期

2025 年 7 月 6 日



龙架构 LoongArch Biweekly 双周会

会前注意事项

本页预定讲者

龙架构 LoongArch Biweekly 双周会

会前注意事项

- 本次会议仅涉及软件技术课题
  - 关于龙芯相关的硬件产品，除官方层面已解禁的消息及本文档内可公开的消息外，其他均不作任何回应
- 本次会议与股市无关，不构成任何投资建议

本页预定讲者

龙架构 LoongArch Biweekly 双周会

快速报告

龙架构上游动向

龙架构 LoongArch Biweekly 双周会

本页预定讲者

GCC

- PR120807
  - Box64 开发者反映，CRC 优化导致 ICE（编译器内部错误）
  - 修复：r16-1998
  - 如无反对将回合到 15 分支
- PR101882
  - 编译器组反映，近期一项龙架构测试开始触发 ICE，xry111 初步排查认为是此古老 bug 导致的
  - 正在试图呼叫上游维护者修复
- 另外 xry111 正在扩展 target pragma/attribute 使其支持开关 amcas、div32 等“龙架构版本演进”特性
  - libatomic ifunc 优化可能会用到

Binutils

- xen0n 增加了一个新的 relax pass，在处理 R_LARCH_ALIGN 的 pass 之后进行
  - 是 6 月 25 日发布会前聚餐闲聊提到的点子之一
  - 此时不能删除指令（否则会破坏对齐，并且由于实现细节，此时不方便补救了），只能把多余的指令换成 nop，但总比没有强
    - 召回了 0.5% 本应被 relax 的操作
  - 另外此时估算被重定位的指令和相应目标符号的距离上限时，可以不用考虑对齐操作可能引起的距离增加了
  - 目前上游维护者正在催下一版（只需要改一些格式问题），~~作者之前说周五晚上会改好，但截止周日凌晨尚未发出~~已经发出了

LLVM

- zhaoqi5 修复了生成 xvshuf 时先前写错的比较（<= 应为 <）
  - 修复了 #137000，将合入 20.x 分支
- zhaoqi5 优化了插入元素、提取元素、插入刚提取的元素的操作
- heiher 优化了向/从 256 位向量的插入/提取 128 位向量元素的操作
- tangaac 减少了偏移量较小的 [x]vstelm 系列指令所涉及的栈帧需求
- heiher 支持了先前新定义的 LA32R/LA32S 重定位操作

WebKitGTK 和 Skia

- 上周提到的构建修复 (WebKit/WebKit#46949) 已合并
  - 已回合到 2.48 分支
  - 在 2.48.4 发版后就应该可以在龙架构正常构建了
- Skia 方面的沟通问题无进展

Firefox

- xen0n 启用了 libyuv SIMD 优化: D245854
  - Firefox 142 主线外补丁数量归零!
- zhaojiazhong 修复了编译错误 bug 1970081: D252624
  - 将被合回 (backport) Firefox 141 分支

Buildroot

- FlyGoat 为 buildroot 添加了 LoongArch 支持，已主线化
  - 初期支持 LoongArch64 UEFI 目标 (target)，在 QEMU 与 3A5000 桌面系统测试通过
  - 利好嵌入式开发与（个别）容器化应用场景

Linux 内核（loongarch 列表）

- Gao Han
  - 为龙架构标记 AMD KFD 支持（第 1 版）
- Bibo Mao
  - 为 KVM 新增中断控制器 IOCSR MISC 寄存器模拟支持（第 1 版）
  - 修复 KVM 中对 EIONTC 中断控制器的模拟功能（第 4 版）
  - 增强 KVM 中对 EIOINTC 中断控制器的模拟功能（第 5 版）
  - 为 CPUCFG 及 CSR 导致的 KVM 模拟退出新增跟踪点 (trace point) 支持（第 1 版）
- Ming Wang
  - 保护 EFI 内存映射区，修复 kdump 在 boot_memmap 内容可能被覆盖，导致 kdump 不可用的问题（第 1 版）
  - 新增使用 mem=SIZE 内核参数指定可用内存容量/区间的功能（第 1 版）

Linux 内核（loongarch 列表）

- Binbin Zhou
  - 龙芯 2K0500 BMC 支持（第 7 版）
  - 龙芯 2K0500/2K1000/2K2000 MMC 控制器支持（第 4 版）
- Qunqin Zhao
  - 龙芯安全引擎、TPM 设备支持（第 12 版）
- Yao Zi
  - 为 SMCv0 固件设备添加 CPU 调频支持（第 1 版）
  - 避免直接对 FDT 内容进行字符串替换，导致 FDT 校验失败（第 1 版）

快速报告

龙架构发行版变动

龙架构 LoongArch Biweekly 双周会

本页预定讲者

Arch Linux for Loong64

- Linux内核默认使用Intel Xe驱动（by wszqkzqk）
  - 所有内核变体（linux、linux-lts、linux-zen、linux-hardened）均调整了Intel Xe显卡驱动默认配置
  - 启用CONFIG_DRM_XE_FORCE_PROBE="*"
  - Intel显卡理论上现在开箱即用
  - https://github.com/lcpu-club/loongarch-packages/pull/623
- OCaml使用机器码支持补丁（by wszqkzqk）
  - OCaml不启用机器码支持会导致大量奇怪问题且上游难以处理
  - 仍然Backport了OCaml暂不愿合并的机器码支持补丁
  - https://github.com/lcpu-club/loongarch-packages/pull/627
  - 依赖ocaml-findlib的程序现在可正常构建

龙架构 LoongArch Biweekly 双周会

本页预定讲者

Arch Linux for Loong64

- 修复Firefox Dev 141构建失败（by wszqkzqk）
  - 修复Assembler-loong64.h缺失类方法定义问题
  - https://github.com/lcpu-club/loongarch-packages/pull/634
  - 暂时没来得及上游化
- 修复plasma-vault的新增依赖gocryptfs（by wszqkzqk）
  - https://github.com/lcpu-club/loongarch-packages/pull/616
- 常规跟进上游升级和补丁清理等（by Pluto Yang）

龙架构 LoongArch Biweekly 双周会

本页预定讲者

安同 OS (AOSC OS)

- LoongGPU 内核模块已完成重构
  - 已兼容 6.16（目前 RC）内核
    - 保持了对 6.12（含）以上内核版本的兼容性
      - 不再兼容更老的内核
  - 由于不明原因，性能似乎有所提高
    - 使用笔记本屏幕时，KDE 开混成拖窗口不再卡成 PPT
  - 由于不明原因，可以显示虚拟终端了
    - 但 KMSCON 会崩溃，然后回退到传统的 framebuffer console
  - 背光问题待查
    - 可能是由于内核构建时没有启用 PWM 支持导致的
  - 仍不支持 LG200
- 3A6000 笔记本（联想开天除外）触摸板修复及 CPU 调频支持已推送
- 2K0300 镜像已基本准备就绪，近期发布

龙架构 LoongArch Biweekly 双周会

本页预定讲者 xry111

安同 OS (AOSC OS)

- **本期安全更新**
  - **Firefox 140.0.2 及 Thunderbird 140.0**
    - 修复 Mozilla 基金会近期安全公告中披露的 10 个安全漏洞（含 2 个高危漏洞）
  - **Node.js 22.16.0**
    - 修复 2025 年 5 月 14 日安全更新公告中披露的 3 个安全漏洞（含 1 个高危漏洞）
  - **Node.js 20.19.2**
    - 修复 2025 年 5 月 14 日安全更新公告中披露的 2 个安全漏洞（含 1 个高危漏洞）
  - **sudo 1.9.17p1**
    - 修复两处本地提权漏洞，编号 CVE-2025-32462 和 CVE-2025-32463（严重性：高）
- **建议关注公众号“安同开源”或社区主页 (aosc.io) 新闻**

龙架构 LoongArch Biweekly 双周会

本页预定讲者 xry111

Debian

- 包含 Intel Iris 驱动的 Mesa 已经合并到 experimental 仓库，以便龙架构用户使用 Intel 显卡

快速报告

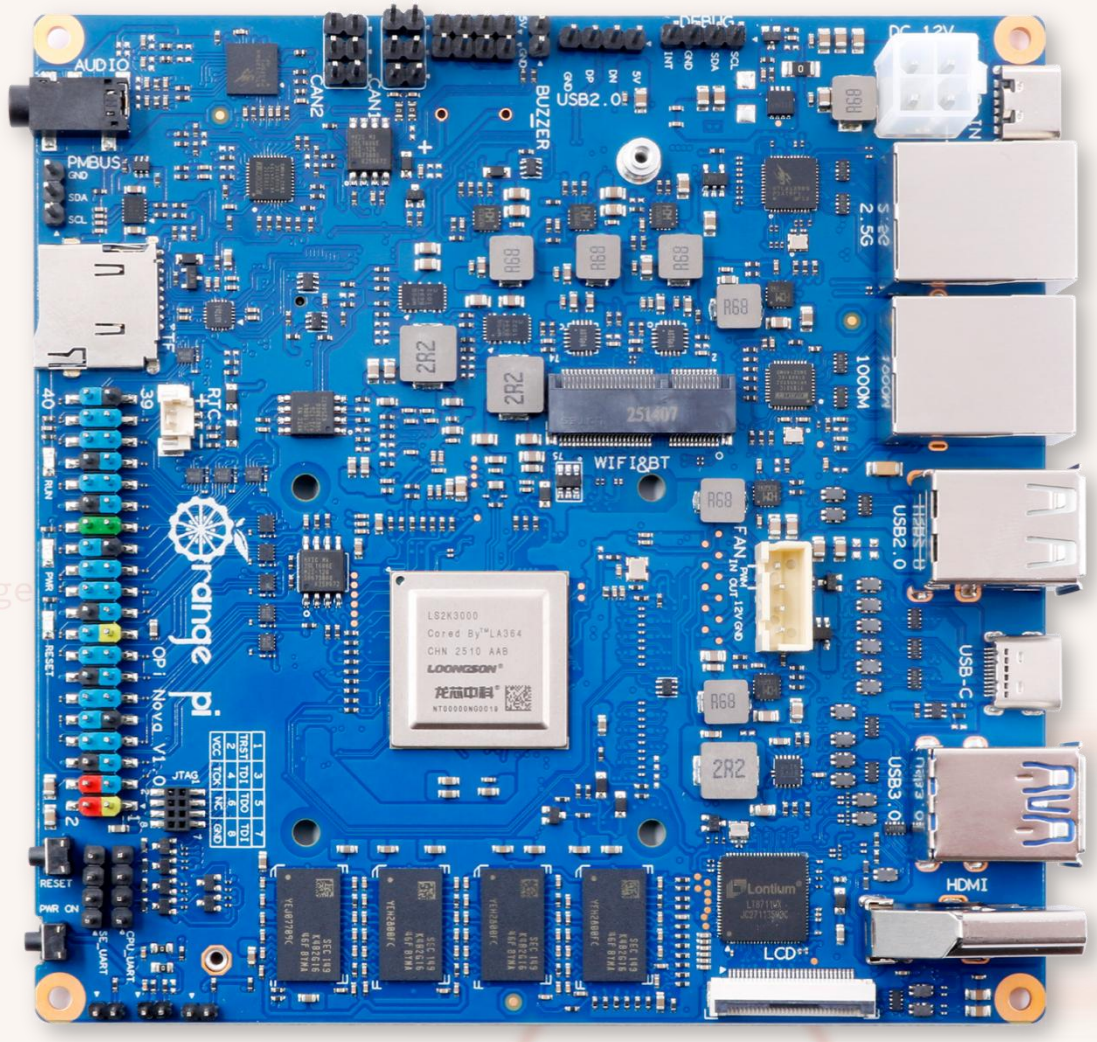
社区事务

龙架构 LoongArch Biweekly 双周会

本页预定讲者

悬赏任务

- 龙架构及 OrangePi Nova U-Boot 上游化
  - 可通过社区开发板漂流计划借用开发设备
- 奖品：32GiB 内存版本的顶配 OrangePi Nova 一张



* 板卡渲染图，请以实物为准

本页预定讲者　白铭骢

龙架构 LoongArch Biweekly 双周会

龙友会・专题报告

安同 OS 的龙架构移植

龙架构 LoongArch Biweekly 双周会

安同 OS 的龙架构移植

一个冷门社区发行版何以成为龙架构社区的主流选择

白铭骢

龙架构 LoongArch Biweekly 双周会

本页预定讲者

安同 OS：基本概况

- 「根发行版」
  - 2014 年结束衍生版历史，开始独立维护
  - 基于 APT/dpkg 包管理
  - 重新设计依赖树，相对于 Debian 大幅度简化
  - 核心包组 (Core) 每年迭代，其余部分通过协商自由更新
- 设计目标
  - 针对个人桌面设备提供开箱即用和简便可靠的使用环境
  - 支持多个架构并尽可能确保使用逻辑、体验一致
    - 尤其是对非 x86 架构存在兴趣性关注
  - 为 CJK（主要是汉语）使用场景优化
  - 结合本地包管理及跨发行版生态，最大化应用兼容性
  - 以志愿为原则，鼓励贡献者及用户友好互助

本页预定讲者

龙架构 LoongArch Biweekly 双周会

本页预定讲者

安同 OS 与龙芯

- MIPS 方面
  - 最早针对 3A2000 进行移植（2016 年）
  - 由于资料缺失及内核、引导器支持混乱等问题消极维护（2016 - 2023）
    - 为衍生项目星霞 OS 提供龙芯 2F 移植的构建环境，持续维护（2020 至今）
  - 内核主线化、体验优化及安装盘正式发布（2024 年）
- 龙架构方面
  - 从 3A5000 小范围铺货开始，启动移植工作（2021 年初）
  - 由于新旧世界迭代辩论放弃移植（2021 年底）
    - 由于规范迭代过快，放弃第二次移植（2022 年底 - 2023 年初）
  - 新世界移植启动（2023 年中）
    - 移植完成并发布为一级架构（2024 年初）
    - 旧世界应用兼容（2024 年初），旧世界固件兼容（2024 年中）

龙架构 LoongArch Biweekly 双周会

本页预定讲者

安同 OS 与龙架构

- 安同 OS 龙架构移植成形的原因
  - 购买第一台 3A5000 台式机进行测试时发现其性能显著高于 3A4000，社区开发者们表露出了极大的兴趣，认为**其性能可能达到日用需求**
  - 在参观龙芯武汉子公司的过程中发现绝大多数职工都在用龙芯办公，**远远超出我们对非主流架构开发应用的预期**
  - 社区维护者间存在文化及政治上对龙芯的**认同和期许**
  - 社区用户间对好用、易用的新世界龙芯发行版呼声较大，也**对旧世界应用的兼容性存在顾虑**
  - 3A6000 的低廉定价**大幅度扩充了社区用户的群体**
- 龙架构在这些条件的支持下，迅速成为了安同 OS 移植维护的重点
  - 换言之：**一级架构**

本页预定讲者

安同 OS：既得利益者与「叛徒」

- 新世界的受益者
  - 两次移植失败过程中，安同 OS 成功**躲开了新旧世界的争执**
  - 新世界生态规范基本稳定的 2023 年，安同 OS 推动移植的条件十分成熟
    - 软件上，关键桌面应用已基本「到位」（能用）
    - 硬件上，社区已有多台 3C5000 工作站，3A6000 也已开始铺货
    - **容易安装、使用便利、支持更新硬件**的新世界系统呼之欲出
- 新世界的怀疑者
  - 新世界生态在 2023 年依然十分贫瘠，商业软件全员缺席
  - 旧世界兼容性成为工作目标
    - **矛盾：兼容旧世界应用让日用新世界系统完成生产工作成为现实，但背叛了当时社区对新世界彻底胜利的期许**

本页预定讲者

安同 OS：何以成为社区主流？

- 安同 OS 在社区中的受欢迎程度远超预期
  - 2024 年二月中发布之后，社区群组人数翻了 2-3 番
  - 更是彻底结束了社区「开发者比用户多」的历史
  - 更有人认为龙架构已替代 x86 成为安同 OS 用户群体的基本盘
- 根据用户的反馈，安同 OS 的优势较为显著
  - 安装简易：全程图形化安装
  - 兼容性强：可以兼容许多旧世界应用程序，尤其是常用的 QQ、微信及 WPS
  - 紧跟前沿：对新龙芯硬件的完善支持往往在 Day 0 达成（如 3C6000）
  - 灵活规避：对龙芯硬件存在的问题有较为到位的规避（如 AMD 显卡稳定性问题）
  - 社区友好：贡献者直接参与用户支持，提供专业意见，用户亦能互助

本页预定讲者

主流的背后

龙架构移植的《安同秘籍》两卷

龙架构 LoongArch Biweekly 双周会

本页预定讲者

秘籍之一：适配优化

- 安同 OS 做了什么？
  - 确保基础桌面设施的可用性
    - Firefox 视频硬件解码、WebRTC 支持（2023 年 10-11 月）
    - Intel 显卡支持、LATX 集成（2024 年 2 月）
    - Spiral：Debian/Ubuntu 应用兼容性设施（2024 年 3 月）
    - LoongGPU 驱动集成 （2025 年 1 月）
    - 3A6000 笔记本触摸板支持、龙架构笔记本动态调频支持（2025 年 7 月）
  - 确保用户常用软件的兼容性：libLoL 新旧世界兼容层（2023 年 12 月）
  - 抹平世界线隔阂：「旧世界」固件兼容性及新旧世界双启动支持（2024 年 8 月）
  - 发现、研究并规避了…
    - AMD 显卡驱动复位问题（2024 年 4 月）
    - 3C6000 PCIe 设备协商速率问题（2025 年 6 月）
    - …

案例展示 (1/4)：libLoL

- 龙上之龙
  - 取义于 64 位 Windows NT 系统的 WoW (Windows-on-Windows) 机制
  - 在新世界龙架构系统无缝运行旧世界应用
- 设计结构
  - 内核组件：通过 la_ow_syscall 模块，给 Linux 内核新增旧世界系统调用支持
  - 运行时：提供特殊版本的 glibc，增补所需的旧世界符号版本
- 实现效果
  - 磁盘空间方面，占用磁盘空间较小
  - 性能方面，额外开销可忽略不计
  - 侵入性方面，不存在侵入式修改，特别方便安装与卸载
  - 正确性方面，足以正常运行多数旧世界典型应用

本页预定讲者

案例展示 (2/4)：AMD 稳定性规避

- 长期存在的稳定性隐患
  - 使用 7A 桥片提供 PCIe 接口的龙芯平台（不论龙架构及 MIPS）均受影响
  - 硬件设计的缺陷（实验物理学测试认为与桥片温度强相关）引发 IRQ/DMA（中断与数据）时序混乱，导致高速 I/O 不稳定
  - 这一问题在 AMD GCN 1.0 - 4.0 显卡上尤为突出
- 意味不明的上游规避，效果显著的下游规避
  - AMD 在 2015 年分别引入规避性代码，在写入内存及发出中断请求前执行了额外的 指令重复写入内存，以期解决一处（AMD 完全没有进行解释的）潜在硬件问题
  - 这一规避放大了龙芯 7A 桥片上可能造成数据不一致问题
  - 为此，社区维护者编写了规避补丁，在保持两次写入的规避代码的设定下，让两次写入的数据保持一致，规避龙架构上可能发生数据不一致的问题
  - 这一规避方案而后被 UOS 采纳，并被龙芯工程师进一步完善

本页预定讲者

案例解析 (3/4)：3A6000 笔记本支持

- 龙芯 3A6000 笔记本的新世界系统支持并不完善
  - 上游内核在移动平台上关键的节电特性、内置 HID 输入设备支持均不完善
- 节电特性：CPU 调频功能
  - 移动（甚至桌面平台）常用的节电措施，上游内核仅支持尚未实装的 SMCv1 固件规范
  - 社区开发者梓瑶实现了 SMCv0 支持，安同 OS 开设测试源，维护者收集测试数据并支持其上游化工作
- 内置 HID 输入设备：触摸板支持
  - 问题根源是龙芯 GPIO 驱动在主线化过程中，由于沟通理解问题，错误丢弃了对中断编号的定义，导致大量 3A6000 笔记本无法使用触摸板
  - xry111 编写了初版补丁，修复了大部分笔记本上的触摸板
    - 最终版本的补丁有待龙芯方面发出

本页预定讲者

案例解析 (4/4)：多媒体性能优化

- 现代硬件之美：专用加速单元分散性能负载
  - 多核心 CPU + 可编程 GPU + 负载均衡 = 「鸿蒙」一般的流畅桌面体验
- 勿以善小而不为：浏览器硬件解码加速
  - 龙架构移植过程中学到的第一个教训：**开荒者无时无刻不存在**
  - 龙架构移植过程中，部分开发者已经开始日用 3C5000 及 3A5000 工作
    - 但发现没法开小差：龙芯的处理器性能难以支撑多任务 + B 站/YouTube 负载
  - 经过社区开发者咸鱼的调查，发现 Firefox 长期在非 x86 平台禁用 VA-API 支持
    - 因为「不太可能用得上」？！
  - 一笔上游提交后，Firefox 122 开始，所有架构均已「硬解平权」
    - 一边干活一边看直播成为现实
- 和其他设备一样，龙芯设备的桌面体验受制于最弱一环
  - 当然，这不是说龙芯 CPU 性能「不好」（尽管 5000 系列芯片就是慢）
  - 而是 CPU 本来就不适合放视频

本页预定讲者

龙架构 LoongArch Biweekly 双周会

秘籍之二：社企互助与外部支持

- 安同 OS 作为互助平台
  - 安同 OS 较快的更新频率、较好的稳定性和紧跟龙芯上游动向的特性，使得安同 OS 成为了社区方面较好的参考对象
  - 安同 OS 让龙芯在上游贡献和产品集成方面的测试验证工作变得更为便利
  - 安同 OS 也是一部分龙芯职工的选择，一部分职工也利用业余时间进行贡献
- 社区驱动
  - 龙架构移植的持续健康推进离不开人民群众和社区、企业的支持
  - 龙架构绝大多数构建服务器来自社区好友捐赠（甚至用不着众筹）
  - 龙芯爱好者社区提供机房托管并协助联络了专线网络
    - 如今，大多数龙架构构建服务器都在其机房托管

向前看，齐步走

安同 OS 龙架构移植的今天与未来

龙架构 LoongArch Biweekly 双周会

本页预定讲者

安同 OS 的龙架构支持概况

- 龙架构是安同 OS 的三大一级架构之一
  - x86-64、AArch64 与**龙架构**
  - 最完整的软件源，最快的更新，最高的测试与维护关注度

本页预定讲者

一级架构

- 支持水平最高的一类架构
- 维护算力充分且维护者关注度高
- 更新最及时、特性最完整

x86-64
(amd64)

AArch64
(arm64)

龙架构 (LoongArch)
(loongarch64)

二级架构

- 支持相对完整的一类架构
- 如条件允许，可升格为一级架构
- 更新和特性完整性和时效性无保障

龙芯三号 (MIPS)
(loongson3)

IBM POWER8+
(ppc64el)

RISC-V
(riscv64)

龙架构 LoongArch Biweekly 双周会

本页预定讲者

龙架构支持概况

- 龙架构是安同 OS 的三大一级架构之一
  - x86-64、AArch64 与**龙架构**
  - 最完整的软件源，最快的更新，最高的测试与维护关注度
- 经测试验证，支持绝大多数社区关注的桌面与服务器平台
  - 5000 系列：3A5000，3B5000，3C5000-L，3C5000-LL，3C5000
  - 6000 系列：3A6000，3B6000（8/12 核心），3C6000/S，3C6000/D
  - 2K3000 与 3B6000M 除 GPU、VPU 外已有完整安装和使用支持
- 嵌入式平台
  - 来自吉林大学的 Ilikara 同学牵头的无 SIMD 移植项目已进入维护状态
  - 镜像即将发布，后续更新与主线同步
    - 将支持所有在售 2K0300 开发板
  - 过程中支持社区开发者完善 2K0300 各组件的上游化支持

未来演进计划

- 二进制翻译应用场景
  - 引入 4/16KiB 分页双内核，支持不同应用需要
  - 引入 Box64 及 LATX-sys
- 应用支持
  - 引入应用商店，进一步改含应用安装体验
  - 针对目前暴露的行业应用，推动软件兼容性改善工作，甚至推动厂商主动支持
- 平台支持
  - 围绕近期发布的 2K3000/3B6000M完善各平台核心组件及外设支持
    - **预装安同 OS 的相关设备也已在路上！**
  - 进一步探索和改进 Linux 内核，消灭与旧世界系统的支持差异
  - 完善现存设备，尤其是移动设备中的「硬茬」设备的支持
    - 联想开天 N60d 支持正在努力完善中

美丽新世界?

- 未来的「美丽新世界」还有不少疑问等着我们
  - LG200 的 GPU、VPU 驱动是否有开源甚至上游化的可能?
  - LG110 的 GPU 驱动支持是否还有进步空间，能否追平旧世界驱动性能?
  - 未来二进制翻译应用对双分页内核的需求是否有更优雅的解决方案?
  - 龙芯在上游社区是否能得到更充分的支持和信任?
  - …
- **愿龙芯不忘社区厚爱，与社区一齐向前进**

本页预定讲者

感谢捧场！

请在议程后提问，关注安同谢谢喵

龙架构 LoongArch Biweekly 双周会

本页预定讲者

龙友会・专题报告

BFD 链接器的链接时松弛（linker relaxation）时间复杂度优化实践

xen0n

龙架构 LoongArch Biweekly 双周会

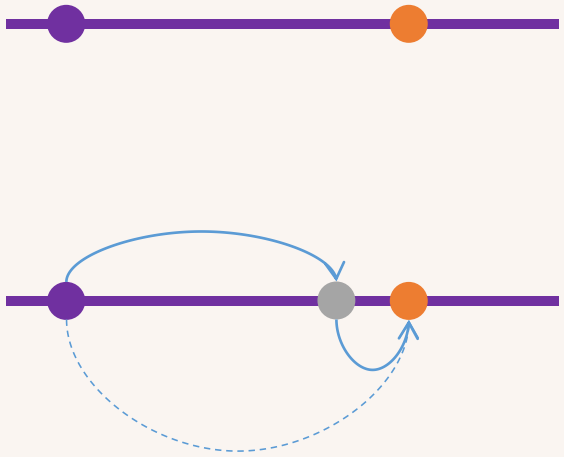
一言以蔽之

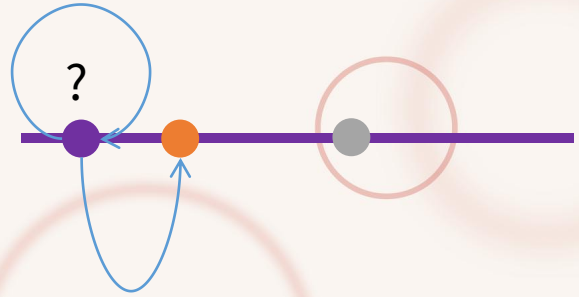
- 大部分情况下（不使用 LLD/Mold 的情况），链接大软件的速度变快了
- 极个别软件的链接时间从**数小时**降低至**几秒**
- 在 BFD 内，LoongArch 链接器松弛的先进程度暂居第一梯队
  - AArch64: 仅松弛 TLS 模型
  - RISC-V: 时间复杂度也有二次项，可用相同方式改进，但暂未这么做
  - 如果算上先前报道的 3-pass 松弛改进：目前没有其他架构像龙一样，把握了全部松弛机会

本页预定讲者

为什么需要链接器松弛?

- 在计算机程序里，经常需要从一个地方引用另一个地方（术语上叫“符号”）的内容
  - 正如现实世界中从 A 点到 B 点距离有长有短，符号间也一样
- 受限于指令集，机器语言的单条操作能访问的范围总是有限的
  - 不然无限长的距离对应的信息量也无限，而指令字长度总是有限
- 那就用多条操作呗?
  - 但对于那些仅需单条指令即可“摸到”的符号，这样做就亏了





龙架构 LoongArch Biweekly 双周会

本页预定讲者

为什么需要链接器松弛？（续）

- 直接生成最优长度的访存不就好了么
  - 但是不能在**编译时**完美预测各种符号的大小——要等编译甚至汇编结束才有机器语言呢，从而也就不知道距离。**链接时**才知道
- 怎么办？揉好的面团醒一醒
  - **编译器、汇编器**不管三七二十一，都生成多条指令的、适应（当前代码模型下）最远距离的访存操作
  - **链接器**在基本完成布局：将符号在地址空间中摆放到位之后，识别出最终距离较近的那些操作，将它们改写为更短的形式，并删除多余的字节
  - 这样操作之后，由于许多符号都缩短，进而可能暴露出更多可松弛的位点
  - 重复上述操作，直至收敛（代码段长度不再变化）
  - 该过程像极了一团内含橡皮筋或面筋的东西静置之后会发生的情况，故名 relax——放松

具体怎么做到的?

- 在 BFD 的 ELF 移植中，由各架构提供 `bfd_elfNN_bfd_relax_section` 钩子供架构无关代码调用
  - 默认什么都不做，具体工作内容由各架构自由指定
- 在 GNU ld 的各个处理阶段，调用 `lang_relax_sections`，由其负责最终调用相应架构的 relax 钩子直至收敛
  - 每个代码节（section）被单独处理
  - 有 relax pass（工序编号）与 relax trip（单一工序内部处理的遍数）等信息提供
  - 架构相关代码的自由度很高：可以在 section flags 字段记录信息，可以自由维护各种状态，自己负责自己状态的内存管理等等

一般都做些什么？以龙架构为例

- 处理较长的可松弛操作：取地址、TLS 模型、过程调用等
  - 取地址：pcalau12i + addi.d (共 32 位) -> pcaddi (R_LARCH_PCREL20_S2)
  - TLS 模型：静态链接时不需动态解析，DESC/GD/LD -> LE/IE
  - 过程调用：pcaddu18i + jirl (R_LARCH_CALL36) -> b/bl (R_LARCH_B26)
- 处理对齐：从汇编器为 R_LARCH_ALIGN 生成的 nop 序列中删除合适的数量
- 把长序列改写为短序列之后，多的指令怎么办？删掉
  - 用 memmove 把被删除段落之后的内容覆写到被删除段落起始位置
  - 对删除位置及之后的所有重定位记录：偏移 -= 删除量
  - 对删除位置及之后的所有符号：偏移 -= 删除量
    - 如果符号范围包括了被删除范围：符号大小 -= 删除量

本页预定讲者

这样有什么问题？时间复杂度！

- 设 section 大小为 m，重定位记录个数为 n，符号个数为 k
- 每次 relax trip 都对每个重定位记录进行处理，O(n)
  - 对每个重定位记录，可能删除也可能不删除字节
  - 如果删除：memmove O(m)，各种偏移、大小调整 O(n+k)
  - 综上：每次 relax trip 的时间复杂度为 O(n(m+n+k)) 有二次项！
- xry111 在 2024 年 6 月即指出了这一点，时隔一年 xen0n 被坑到了
  - mypy 是一个 Python 类型检查工具，同时也是编译器，它会将本身的 Python 代码转译（transpile）为巨大的 C 文件
  - 3347726 重定位记录，可能的松弛位点数量达百万级
  - 在龙芯 3C6000 需要链接 3 小时，AMD Threadripper 3990X 也需 1 小时
  - 以为卡住了，perf top 看一下，结果是在 relax_section……

怎么解决呢?

- 既然要进行 O(n) 次删除，删除操作的时间复杂度也有 O(n)，那么能不能全部堆在一起处理了呢
  - 说是这么说，但每次判断是否能够松弛，都要算距离，需要取到准确地址
  - 需要在不进行实际松弛操作的前提下，支持计算一个地址在松弛后的取值
- 那么用什么数据结构呢?
  - 基本观察:
    - 任何删除操作都不会影响到比它地址更小的任何状态
    - 任何删除操作的范围都局限于相应单条重定位记录的影响范围内
  - 考虑排序，或能够自主维护顺序的数据结构

最优做法：排序！

- 对 relocs、符号都排序，O(n log n + k log k)
  - 这样一来，新产生的删除记录也都自动按顺序排列。将它记录下来，O(1)
  - 求地址 A 松弛后的取值：A - sum(地址 < A 的所有删除记录的删除量)
    - 可以将该累计删除量与删除记录一同存放，这样该操作是 O(1) 的
- 松弛完成后，批量操作删除
  - 实际删除动作：扫描删除记录，将未被删除的段落挨个 memmove
    - 我们以“挪动字节”为基本操作，最多挪动 O(m) 个字节
  - 调整 relocs、符号：扫描 relocs 与符号表，同步扫描删除记录表，改写相应偏移与大小，O(n + k)
- O(n log n + k log k + n + m + n + k)
  - = O(m + n log n + k log k)

但是先别急……

- 同一地址会有多个 relocs，例如可松弛位点的表示方式就是在相同地址先放原始 reloc 再放一个 R_LARCH_RELAX 标记
- 横跨整个 LoongArch 工具链生态，不能排除有些软件会故意不按地址从低到高排列 relocs，并依赖链接器不重排它们的顺序
- binutils 代码库中缺乏高效的稳定排序算法
  - libc qsort 不承诺稳定性，先进排序算法的手工实现很麻烦
- 稳妥起见，能不能不排序？
  - binutils 里可用 libiberty，里头有个 splay tree!
  - splay tree: 一种排序二叉树，每次更新与查询都会把目标节点重排到根位置
  - 渐进复杂度佳：如果经常访问相邻元素，它们会更接近根

使用 splay tree 的做法

- 键：删除操作的原始地址，值：删除量、累计删除量
- 标记删除操作：最差 O(n)，实际 O(log n)
  - 找到上一条删除操作：O(log n)
  - 检查是否当前操作与上一条操作相邻：如是，累计到上一条操作的删除量；如否（或不存在上一条操作），新建一个节点
  - 维护累计删除量：对当前节点及其后的所有节点，累计删除量 += 删除量
    - 最差 O(n)，实际场景下，大部分操作仍然按地址顺序，“其后节点”数量 → 0
- 求地址 A 松弛后的取值：O(log n)，每个可松弛位点计算常数次
  - 找到上一条删除操作 O(log n)，A - 累计删除量
  - 边界情况：如果 A 在被删除的字节之列，那么需要以该删除操作的地址为准！

使用 splay tree 的做法（续）

- 松弛完成后，批量操作删除：
  - memmove 同先前分析，O(m)
  - 调整各种偏移：直接计算 n + k 次松弛后地址
    - O((n+k) log (n+k))
  - 调整符号大小：size -= sum(符号范围内的所有删除记录 的 删除量)
    - 找第一条删除记录 O(log n)，后续只需不断 O(1) 访问后继节点
    - 一般所有内容都有对应的符号归属，所以所有 relocs 都会被用到，也就是每个删除记录节点都会被访问 1 次
    - 整体：O(k log n + n) = O(k log n)
- 整体：O(n log n + m + (n+k) log (n+k) + k log n)
  - 对比最优：O(n log n + m + k log k)，也还行，够用了！

优化效果

- 在 AMD Threadripper 3990X 上再次链接 mypy……
  - 前 `4192.80s user 1.09s system 98% cpu 1:10:53.52 total`
  - 后 `1.76s user 0.74s system 98% cpu 2.539 total`
  - 体感速度堪比 lld!
- perf top 显示绝大部分时间都在 splay_tree_splay 中度过了
- 正确性有保障
  - 仍然能通过全套测试用例
  - 能够成功链接大型软件，如 binutils 自身、LLVM，并通过相应测试用例
- 将在 binutils 2.45 与大家见面！

问答环节

社区问答及意见反馈

本页预定讲者

龙架构 LoongArch Biweekly 双周会

双周会讨论（请先添加管理员）

爱好者交流群

龙架构 LoongArch Biweekly 双周会