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

龙架构双周会

2025 年 1 月 5 日 · 第 3 次

龙架构 LoongArch Biweekly 双周会

快速报告

龙架构社区八卦

龙架构 LoongArch Biweekly 双周会

龙芯爱好者社区

- 龙芯爱好者社区由社区共同参与管理，致力于搭建一个龙架构开发者之间互相沟通的桥梁，尽可能避免出现因为缺少沟通和及时的信息同步导致的重复开发以及其他负面影响。为龙架构开发者创造一个良好的开发环境。
  - 主要面向用户：龙架构开发者以及想要了解龙架构的开发者
  - 主要形式：论坛，群聊，双周会, 提供基础设施
  - 计划功能：
    - 基础设施集群（搭建PVE集群, 基于PVE集群搭建CI/CD设施）
    - 优化论坛使用体验(flarum)
    - 龙架构观测矩阵
    - 统一登陆平台OpenID

龙架构 LoongArch Biweekly 双周会

社区提案

龙架构社区联络组织

联合现有社区，开放接纳新社区，所有龙架构用户提供一个良好的社区氛围

- 成立目的: 社区之间明确定位, 互相宣传, 解决当前龙架构社区较为零散, 各个社区之间功能重复, 定位缺失导致对想要了解龙架构的人员不友好

- 预计分类: 龙架构开发者, 龙架构用户, 龙架构新闻记录(实时新闻分享和里程碑记录), 以及各发行版社区和开源软件的龙架构社区等

- 事务: 联络组织成员发布各自社区介绍, 组织成员直接互相宣传

龙架构 LoongArch Biweekly 双周会

快速报告

龙架构项目进展

龙架构 LoongArch Biweekly 双周会

Loongros2

ROS2 Jazzy 软件发行版 on loong64

- 项目位于 https://github.com/loongros2
- 目标 ros jazzy desktop 组件
- 目前状况：
  - 使用上游脚本修改适配 loong64，完成搭建编译农场
  - ros core 组件编译完成
  - 目前大约400个包可用

龙架构 LoongArch Biweekly 双周会

快速报告

龙架构发行版变动

龙架构 LoongArch Biweekly 双周会

安同 OS (AOSC OS)

- Linux 6.13 内核已开测，包含所有无争议的列表补丁
  - 在旧世界代码中发现使用 USB 输入设备唤醒设备的支持 (USB Remote Wake)，目前已请求陈华才老师推进上游
- darkyzhou 推送了 VSCodium 1.96.2，包含龙架构支持
- 推送了 Chromium 131（硬件加速支持修复中）
  - 目前硬件加速功能缺失，调查发现上游设定了数百条显卡黑名单
  - 该黑名单对 x86 不生效且大量规则来自十年前
  - 合理怀疑谷歌早已不使用这套规则
  - 补丁由陈嘉杰维护，欢迎发行版维护者同事们参考：**https://github.com/AOSC-Dev/chromium-loongarch64**
- x86 运行时修复：32 位修缮进度 90% 左右

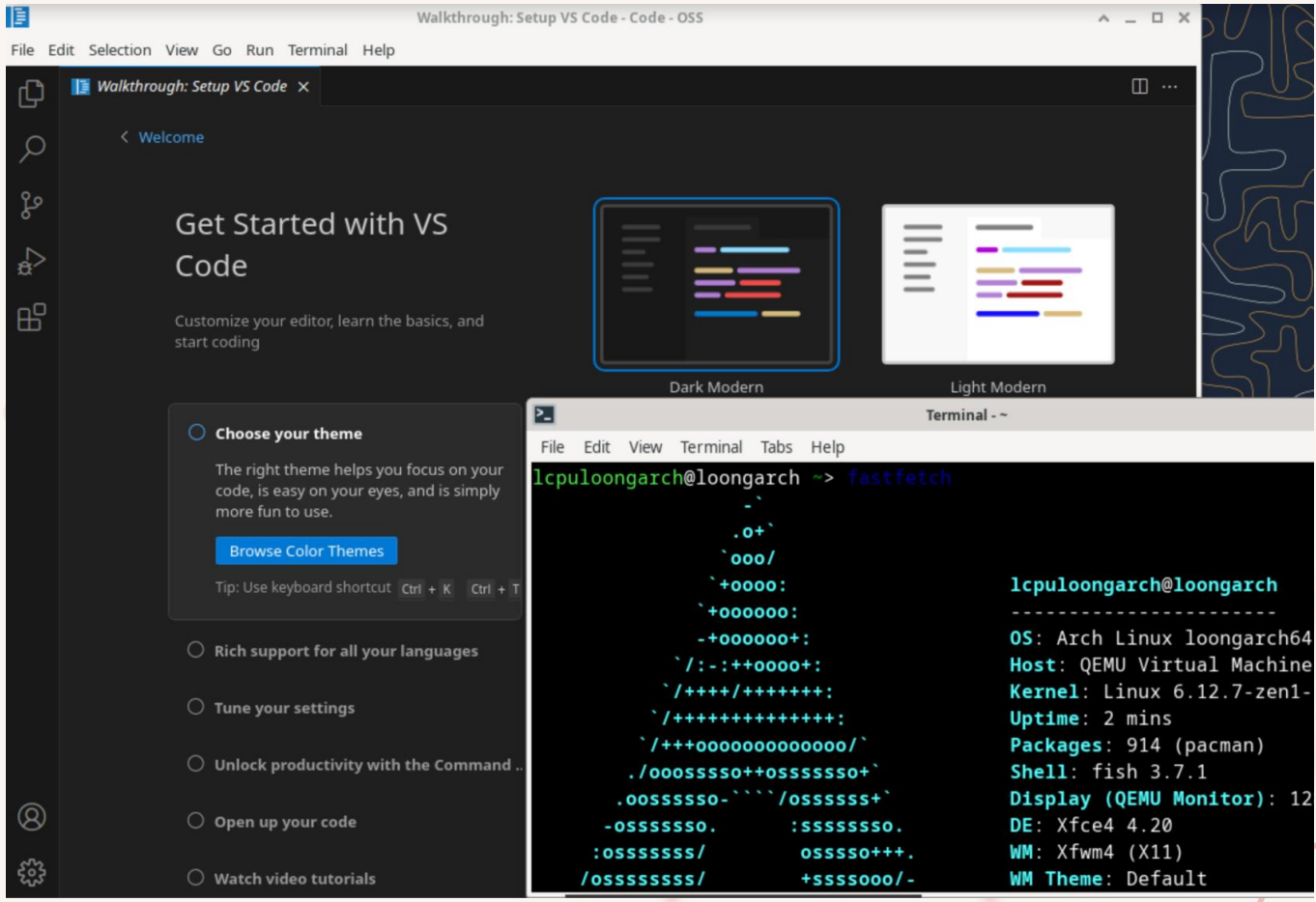
Arch Linux for Loong64

- 本项目正式接替武老师维护的原Loong Arch Linux项目
  - 与上游交流时常用名称
    - Arch Linux for Loong64
    - Arch Linux Port for Loong64
    - 避免"Loong Arch Linux"给上游造成的断句困惑
  - 出于兼容性并考虑原维护者意愿，仍认可原项目名称

龙架构 LoongArch Biweekly 双周会

Arch Linux for Loong64

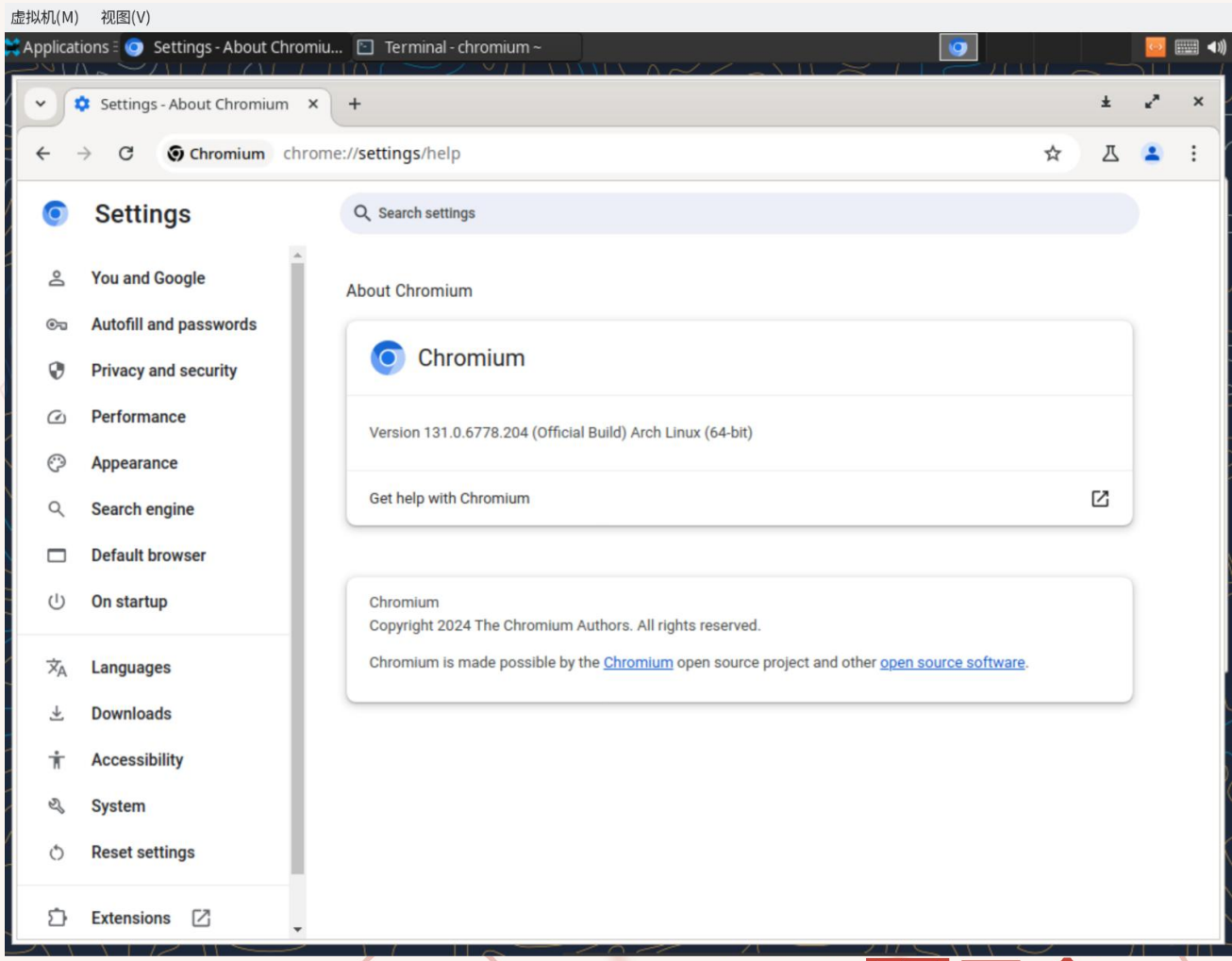
- 各种日常更新、缺失包补充
- Python 3.13升级
  - 一切顺利
- 成功从源码构建electron32
  - 基于Arch上游构建流程
  - 不依赖latx、docker等工具/环境
  - 已验证可用
- 成功从源码构建Code - OSS
  - 即开源版vscode
  - 已验证可用



龙架构 LoongArch Biweekly 双周会

Arch Linux for Loong64

- 成功从源码构建Chromium
  - 已验证可用
  - 至此三大内核浏览器在Arch Linux for Loong64上全部可用
- 发布bootstrap/container用的tarball
  - 包含上游的tar.zst
  - 额外包含.sfs
  - 预计以后会和iso同步发布



双周会

Arch Linux for Loong64

- 上游化（均已仓库中集成临时修复构建版本）
- cilium
  - 基于eBPF的网络、安全和监测方案
  - 已PR，上游未响应
- OpenRCT2
  - 开源游戏实现
  - 已PR，且群友“米饭”完成了性能测
  - 上游已合并
- root
  - 存储、处理和分析科学数据的大型项目
  - 已PR，上游审核中

Arch Linux for Loong64

新的社区团队致力于：

- 及时跟随Arch Linux官方的更新进度，持续维护Loong Arch Linux发行版
  - 最终借助Arch Linux Ports平台，推动Arch Linux官方增加对龙架构的支持，并同步发行龙架构版本
- 修复上游软件在龙架构上的构建问题，并尽可能将修复上游化
  - 最终推动各软件包上游提高龙架构的维护等级
- 培养更多能够为龙架构、Arch Linux及上游社区生态作出贡献的人才，建设更加开放健康的开源社区

龙架构 LoongArch Biweekly 双周会

Arch Linux for Loong64

- 社区团队情况
- 此前因未正式发布缺少对外宣传
- 参与者均来自LCPU
  - 实际人群有限
- 目前社区规模较小
- 欢迎大家参与！！！

负责人 / Leaders

- Pluto: 项目负责人，北京大学学生 Linux 俱乐部指导教师
- wszqkzqk: 社区负责人，北京大学学生 Linux 俱乐部 AP

支持人员 / Support Staffs

- YHStar: Loong Arch Linux Staff，ISO 维护者，北京大学学生 Linux 俱乐部 AP

贡献者 / Contributors

https://github.com/lcpu-club/loongarch-packages/wiki

龙架构 LoongArch Biweekly 双周会

Arch Linux for Loong64

- 欢迎大家参与！！！
- 文档丰富
  - 完整易用的开发者工具
  - “保姆式”维护流程教学
  - 详尽的修包问题指引
- 不定期发布Good First Issue
  - 一般经我确认过工作量
  - 单独提供相关建议/示例/指引

贡献指南 / Contributing

如果你是第一次贡献这个项目，可以首先阅读：

- 开始项目贡献和社区交流前必要学习的概念
- Loong Arch Linux 移植工作流程

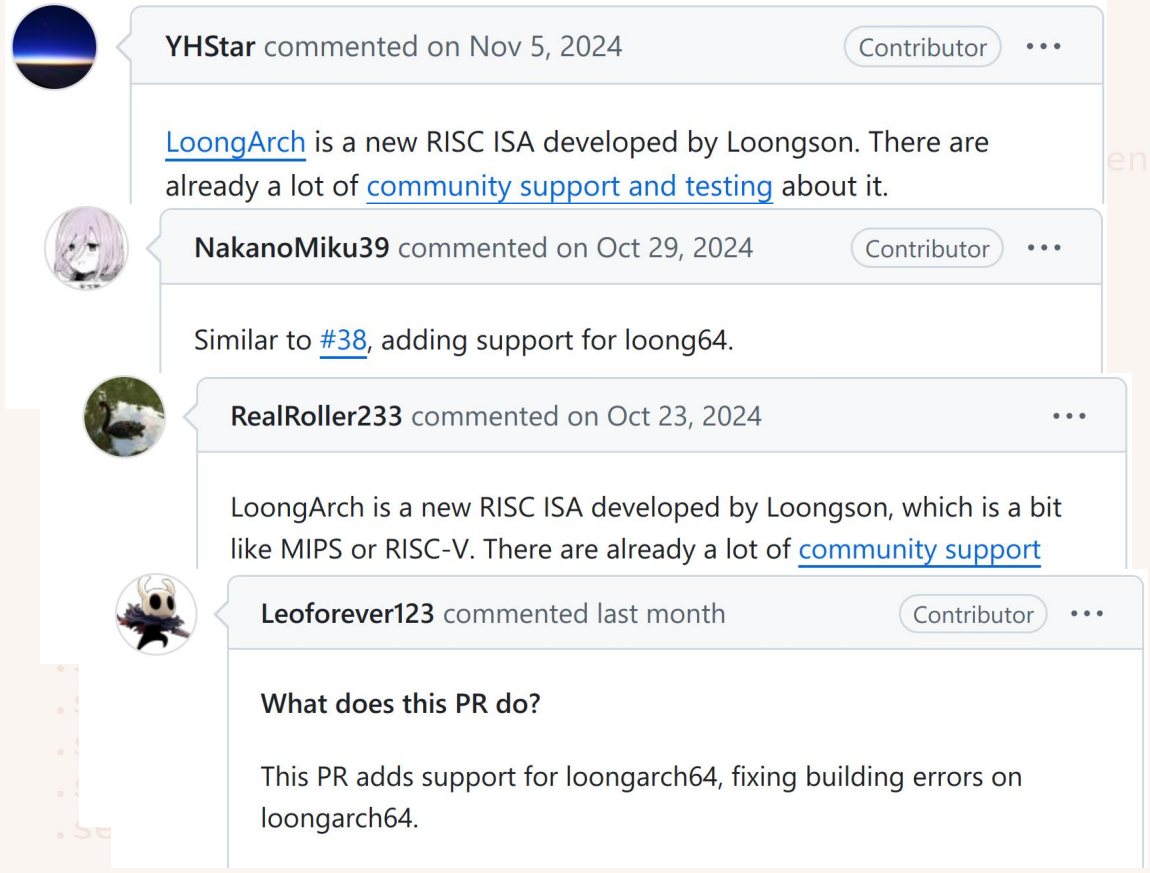
掌握以上内容后，如果在构建或者修复过程中遇到问题，可以进一步查阅：

- 参与移植工作的注意事项、工作指引及FAQ
- 利用本地仓库实现有依赖关系的软件包的顺序构建 - Loong Arch Linux下要求同步升级的软件包的构建方法
- 面向新人的Bootstrap构建基础指引

龙架构 LoongArch Biweekly 双周会

Arch Linux for Loong64

- 绝大多数贡献者已熟悉维护和上游化流程
- 已实现社区化



- 前两大维护者完成了绝大多数的修复
  - 不是所有人都是工作狂
  - 无法也不应要求贡献量
  - 其他人的贡献绝对量也并不算小
  - 会做Good Fist Issue与会自己寻找任务的鸿沟

Arch Linux for Loong64

> Whether it's FOSS or not, someone has to cover the development cost. If users are not paying, it's on the **developer**.
>
> ——rui314 (maintainer of mold)

- 分散主要开发者负担
- 吸引/培养更多开发者
- 个人认为培养对本项目不容忽视
  - 学校/社团的性质
    - 无资金/赞助，参与者无利益获取
    - 较难直接吸引人长期/有保证地贡献
    - “培养”才能产生归属感

- 尝试
- 已将镜像打包等杂务拆分给YHStar
- 以后双周会尽可能让其他参与者讲
  - 帮助全局认识项目
  - 帮助找到参与者自身的兴趣点
  - 我仍然可以审核/补充

龙架构 LoongArch Biweekly 双周会

Arch Linux for Loong64

- CI的必要性
- 机械的打包任务
  - **合理的**CI可以减轻维护者工作量
  - Pluto一人完成6419/11860
  - 有一定脚本化但不够完善
- 校外参与者使用
  - 免于外部上传二进制包
- 自动提供审核参考
  - 目前PR审核几乎均由我一个人完成

- 现状
  - 一键构建文件拉取与补丁应用
  - 已集成补丁导出工具
  - 已制定多包批量构建基本流程
  - 初步开发了上游追踪工具
  - 其他较简单、缺乏环境兼容性的由各开发者自己维护的批量构建工具
- 难点
  - 与标准流程对接
  - 集成/配合合适的仓库管理工具
  - 构建顺序问题
  - Rebuild问题
  - ……

龙架构 LoongArch Biweekly 双周会

Arch Linux for Loong64

- 经不到半年的工作正式发布
- 从头构建了更易于维护的补丁集式维护仓库
- 实现社区化
  - 开发了强大的devtools-loong64
  - 贡献文档完善/门槛大幅降低
- 提交大量贡献到上游
- 完成度媲美在龙架构上耕耘更久的社区发行版
  - KDE 6 / GNOME / Xfce / ……
  - Firefox / Chromium / Webkit
  - Code - OSS / Electron
  - LibreOffice
  - ……
- 离不开其他社区的贡献

- 严重消耗Leaders精力
  - 修包/打包
  - devtools-loong64
  - 文档、教程、指引……
  - 审核、具体（甚至精确到行的）协助

- 期待得到更广泛的帮助
  - CI
  - 仓库管理工具
  - 硬件支持（3C6000？）
  - 软件包修复的援助

龙架构 LoongArch Biweekly 双周会

龙芯俱乐部 多系统社区迷你PC

- 优利龙 迷你PC 首台适配Openharmony 5.0 的龙芯迷你PC
  - 硬件配置：龙芯3A6000 主频2.5G 16G内存 512G SSD硬盘 2XUSB3.0 4XUSB2.0 2XHDMI 1X千兆网口 尺寸：15X15x4.2 厘米 全铝外壳 可定制CNC外壳
  - 系统：Loongnix\诚鸿OS社区版（OpenHarmony 5.0 ）\Deepin等可定制多系统启动
  - 交流社区：用户交流《龙芯俱乐部》loongsonclub.cn 技术交流《龙芯爱好者社区》loongbbs.cn
  - 2024年12月20日 武汉2024开放原子开发者大会暨首届开源技术学术大会上首次展示龙芯OpenHarmony 5.0 迷你PC
  - **参加龙芯爱好者社区的漂流板项目，需要申请者基于诚鸿OS社区版进行开发。**

龙芯俱乐部开源鸿蒙PC社区
联系方式:
QQ 200888797
微信 17205117117
网站:www.openloongson.org

龙架构 LoongArch Biweekly 双周会

诚鸿OS-OpenHarmony PC

- 诚鸿OS介绍

  项目由诚鸿和龙芯合作，基于Openharmony 5.0版本，开发的桌面终端操作系统，填补了开源鸿蒙PC生态在龙芯平台上的空白。

- 研发历程

  2024年1月份项目启动

  2024年5月份完成龙芯2K2000平台OH 4.1 Mini版本适配

  2024年10月份完成OH 5.0beta 标准版在3A6000 + AMD 580显卡适配

  2024年11月份完成OH 5.0beta 标准版在3A6000 + JM9100显卡适配

  2024年12月份完成OH 5.0beta 标准版在3A6000 + 龙芯集成显卡适配

  2024年12月份完成OH 5.0 release 标准版在3A6000 + 集成显卡/JM9100显卡适配

诚鸿OS-OpenHarmony PC

- 支持的硬件设备
  - 望龙/诚迈台式机: 3A6000+7A2000/JM9100
  - 望龙/诚迈笔记本: 3A6000+7A2000
  - 优利龙 3A6000 MINI主机/开发机
  - 智龙派 2K2000开发板

- 已完成的软件应用
  - PC桌面环境（桌面、文件管理器、系统设置、锁屏、登录...）
  - 桌面小游戏（连连看、五子棋）
  - 常用应用软件（计算器、日程、文本编辑器、画图...）

诚鸿OS-OpenHarmony PC

- **安装程序与镜像**
  - 提供了ISO安装镜像
  - 全新的安装程序
  - 象PC Linux一样安装
  - 支持U盘、光盘、网络等多种安装介质
  - 通过Grub启动菜单引导，可以支持多系统启动

- **开发调试工具**
  - 通过HDC提供灵活的调试方式
  - 支持网络连接、USB连接、蓝牙连接、串口连接等
  - 支持设备连接、文件推送、应用安装、查看日志、查看设备信息等功能
  - 可以进行命令行交互

诚鸿OS-OpenHarmony PC

- 应用生态
- 鸿蒙原生应用

鸿蒙原生应用是基于方舟编译器开发的鸿蒙应用，以ArkTS为主要开发语言，也支持C语言的编译。支持在手机、平板电脑、电脑PC等终端平台上运行。

截至目前，鸿蒙生态已经有1万5千多个原生应用。

- Qt应用

QT是一个非常流行的跨平台开发框架，广泛应用于Linux、Windows、macOS等操作系统上的桌面和嵌入式应用开发。

通过在鸿蒙上对QT中间件进行支持，可以在鸿蒙上编译和运行QT应用程序。

诚鸿OS-OpenHarmony PC

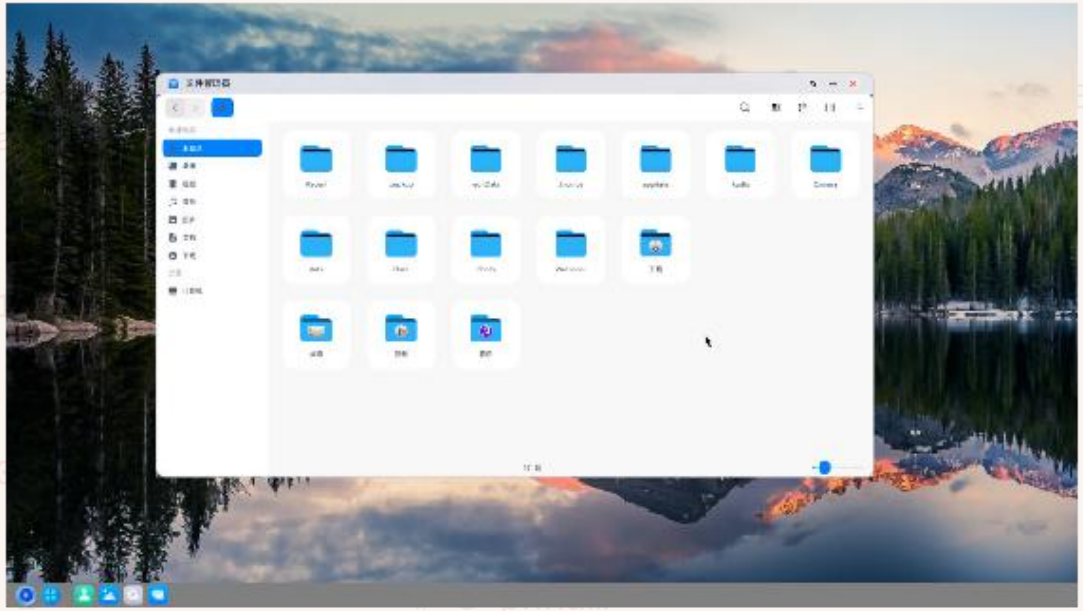
- 版本策略
  - 商业定制版

    针对定制机型发布商业定制版，目前仅支持望龙/诚迈电脑、笔记本，提供长期技术支持和定制服务。
  - 社区开发版

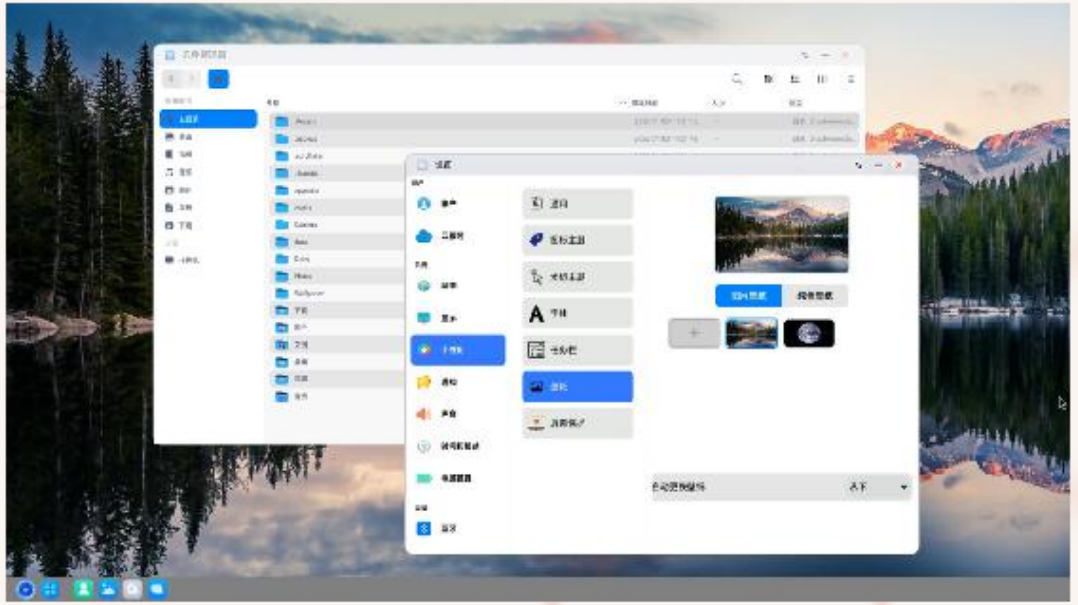
    提供系统源码、镜像和开发套件供社区开发者免费使用，目前已完成优利龙3A6000开发机和2K2000开发板的适配。
- 版本计划
  - 商业定制版-预计2025年上半年正式版本发布
  - 社区开发版-预计2025年1月底提供镜像和开发套件下载

诚鸿OS-OpenHarmony PC

- 应用截图



Qt应用-文件管理器

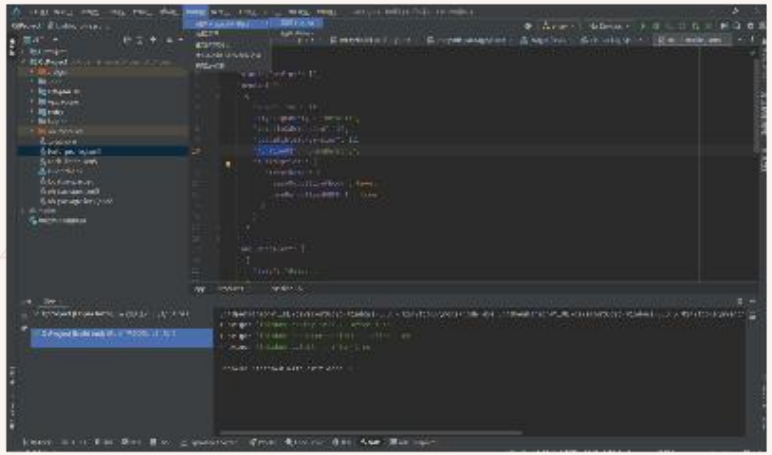


鸿蒙原生应用-设置

诚鸿OS-OpenHarmony PC

开发 OpenHarmonyOS 应用及元服务的集成开发环境（IDE）

- 支持的功能
  - 代码编辑
  - 编译构建
  - 应用推送并运行



- 支持的应用
  - ArtTS应用（原生应用）
  - Qt应用

    参考资料：openharmony-sig/qt/wikis/基于DevEco的Qt工程配置
  - DTK应用

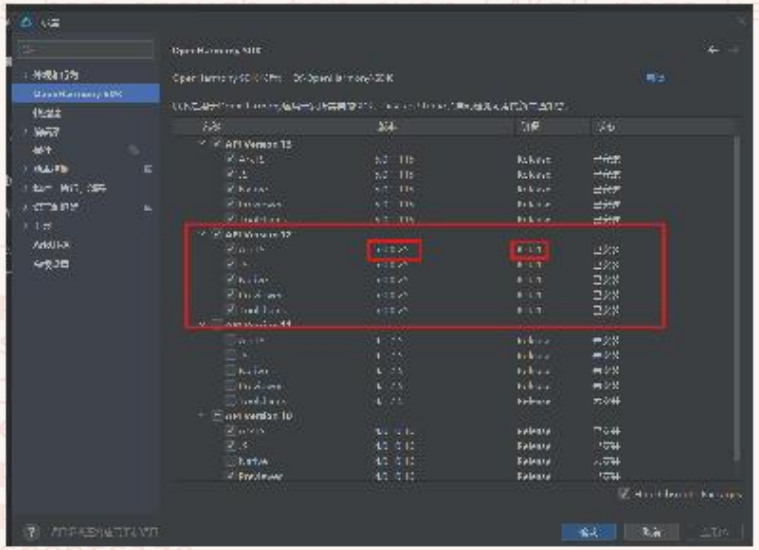
诚鸿OS-OpenHarmony PC

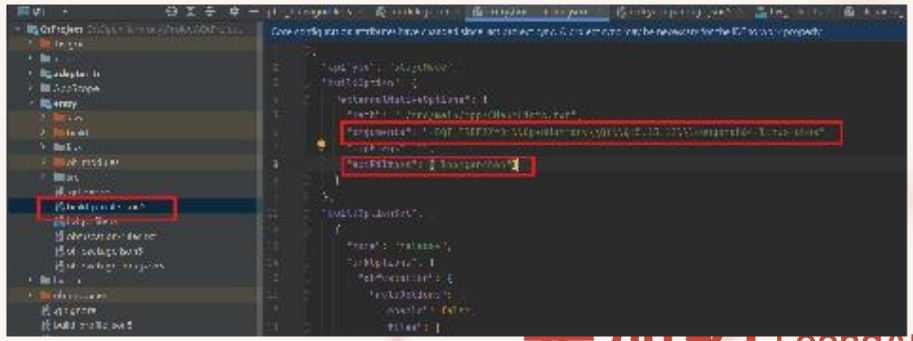
开发工具及SDK

- DevEco Studio（5.0.3.814-loongarch64）

  Windows下开发和使用，支持loongarch64架构

- OpenHarmony SDK API 12 （5.0.25/5.0.71）
- Qt (5.15.12)
- DTK(5.5)





问答环节

社区问答及意见反馈

龙架构 LoongArch Biweekly 双周会

商业软件问题汇总

- 目前常用的办公协作/聊天平台飞书依然只提供 MIPS 龙芯版本，请问目前是否有与其合作推出龙架构版本的计划?

- 龙芯会和JetBrains合作移植开发工具到龙芯平台上吗 jbr换一下部分能跑 但是自己移植分发可能会有版权问题 JetBrains中国代理官方工作人员的说法是等到龙芯平台上的开发者足够多才会适配 这样的速度可能太慢了

- 龙芯剪辑有计划上架UOS应用商店吗？是否支持新世界？

1系列与固件

- 针对1系列MCU，龙芯准备公开发布自己的IDE、函数库及GUI配置工具吗？或者通过某种合作形式支持第三方公司免费推出相关工具吗？

- 龙芯有计划为1系列MCU适配OpenOCD吗？

- edk2固件上游支持龙架构的难点在哪儿？（是技术问题还是非技术问题？能展开讲讲吗？）社区开源笔记本的edk2固件有望做进上游吗？

内核驱动相关

- Loongnix 25 Beta 5 在上次跳票后，是否有新的发版计划？社区许多用户对 LoongGPU 驱动的诉求强烈，希望能得到进一步的消息

- 请问龙芯是否有计划排查新旧世界内核的特性差异，以便避免旧世界用户迁移后遇到老问题？

- LATX 用户态模拟器的运行时打包是否有计划制定标准：包含哪些运行时、应用程序和脚本（如 runapp）？这对第三方适配龙芯应用合作社上的 x86 应用比较有帮助

专题报告

龙架构 LoongArch Biweekly 双周会

WebKit JavaScriptCore虚拟机解释器、JIT编译器 LoongArch64移植报告

龙架构 LoongArch Biweekly 双周会

WebKit JSC虚拟机LoongArch64移植

WebKit简介

开源的Web浏览器引擎。苹果（Safari、Mail、App Store）生态中不可或缺的基础设施。macOS、iOS、watchOS、visionOS和Linux上的许多应用程序使用的Web浏览器引擎。

LoongArch的Java生态中eclipse和openjfx的web模块依赖webkit2gtk。.NET生态中webkit2-sharp rust生态中webkit2gtk-rs依赖webkit2gtk。

通过搜索Linux软件包依赖关系，发现依赖webkit2gtk的软件包：
birdfont、capnet-assist、devhelp、evolution、geary、gnome-boxes、gnome-notes、gnome-online-accounts、libgepub、luakit、lutris、marker、nemo-preview、nyxt、shotwell、sushi、vimb、xreader、yad、yelp。

龙架构 LoongArch Biweekly 双周会

WebKit JSC虚拟机LoongArch64移植

首次移植的工作范围

模块	用途	
offlineasm汇编器	LLInt解释器	1886行Ruby代码。类似OpenJDK的adlc代码loongarch_64.ad。
LOONGARCH64Registers寄存器定义	JIT编译器	134行C++代码。类似OpenJDK的register_loongarch。
GPRInfo通用寄存器约束	JIT编译器	117行LA相关C++代码。
FPRInfo浮点寄存器约束	JIT编译器	97行LA相关C++代码。
LOONGARCH64Assembler汇编器	JIT编译器	2687行C++代码。类似OpenJDK的assembler_loongarch。
MacroAssemblerLOONGARCH64宏汇编器	JIT编译器	5540行C++代码，类似OpenJDK的macroAssembler_loongarch。
LOONGARCH64Disassembler反汇编器	JIT编译器dump调试	402行C++代码，类似OpenJDK的hsdis。

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机简介

JSC是WebKit的内置JavaScript引擎，它按照ECMA-262规范实现ECMAScript。
项目和库的名称始终是JavaScriptCore。源代码位于WebKit源代码树中，位于 Source/JavaScriptCore目录下。

JSC是一个优化虚拟机。由以下组件构成：词法分析器、解析器、专注启动性能的解释器（LLInt）
Baseline JIT、专注低延迟性能的JIT（DFG）和高性能 JIT (FTL)。

模块	类比OpenJDK	AArch64	X86	RISCV64	LOONGARCH64	优先级
CLoop解释器	zero	实现	实现	实现	实现	高
LLInt解释器	templateTable	实现	实现	实现	实现	高
Baseline JIT编译器	C1	实现	实现	实现	实现	高
Disassembler反汇编器	hsdis	实现	实现	实现	实现	低
DFG JIT编译器	C1	实现	实现	实现	实现	高
DOM JIT编译器	Intrinsic	实现	实现	实现	实现	高
RegExp JIT编译器	Intrinsic	实现	实现	实现	未实现	高
FTL JIT编译器	C2	实现	实现	未实现	未实现	一般
B3、AIR JIT编译器	C2	实现	实现	未实现	未实现	一般
BBQ、OMG JIT编译器	Intrinsic	实现	实现	未实现	未实现	一般
向量指令	LSX、LASX	实现	实现	未实现	未实现	一般
异步并行无暂停分代GC	分代ZGC	实现	实现	未实现	未实现	一般
bmalloc、libpas	Arena::Amalloc	实现	实现	未实现	未实现	低

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 修改cmake构建脚本

  修改Source/cmake/WebKitFeatures.cmake：

```
@@ -112,6 +112,12 @@ macro(WEBKIT_OPTION_BEGIN)
     set(USE_SYSTEM_MALLOC_DEFAULT OFF)
     set(ENABLE_C_LOOP_DEFAULT OFF)
     set(ENABLE_SAMPLING_PROFILER_DEFAULT OFF)
+elseif (WTF_CPU_LOONGARCH64)
+    set(ENABLE_JIT_DEFAULT ON)
+    set(ENABLE_FTL_DEFAULT OFF)
+    set(USE_SYSTEM_MALLOC_DEFAULT ON)
+    set(ENABLE_C_LOOP_DEFAULT OFF)
+    set(ENABLE_SAMPLING_PROFILER_DEFAULT OFF)
 else ()
     set(ENABLE_JIT_DEFAULT OFF)
     set(ENABLE_FTL_DEFAULT OFF)
```

龙架构 LoongArch Biweekly 双周会

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 实现offlineasm汇编器

  实现Source/JavaScriptCore/offlineasm/loongarch64.rb。将llint/LowLevelInterpreter的中间表达式（以下简写为IR）下降到LoongArch64汇编指令。类似OpenJDK将adlc源代码自动生成C++源代码，offlineasm将Ruby脚本自动生成LLIntAssembly.h等C++源代码，截取部分自动生成的LLIntAssembly.h源代码如下所示：

```
".loc 1 1735\n"   OFFLINE_ASM_GLOBAL_LABEL(vmEntryToJavaScript)
".loc 1 1119\n"    "addi.d $r3, $r3, -16 \n"         // LowLevelInterpreter.asm:1119
                   "st.d $r1, $r3, 8 \n"
                   "st.d $r22, $r3, 0 \n"
".loc 1 1124\n"    "move $r22, $r3 \n"               // LowLevelInterpreter.asm:1124
```

龙架构 LoongArch Biweekly 双周会

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 实现offlineasm汇编器

  offlineasm的Ruby脚本调试方法：在llint的IR中添加break 0x5，例如：

```
op(llint_handle_uncaught_exception, macro ()
    getVMFromCallFrame(t3, t0)
    restoreCalleeSavesFromVMEntryFrameCalleeSavesBuffer(t3, t0)
    storep 0, VM::callFrameForCatch[t3]
=> break 0x5
    loadp VM::topEntryFrame[t3], cfr
    vmEntryRecord(cfr, t2)
...
```

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 实现offlineasm汇编器

  使用gdb -ex=r --args ... 触发SIGTRAP，使用x/22i $pc-44检查汇编上下文，通过i r $r打印具体寄存器值，set $pc+=4，单步si执行，找到offlineasm/loongarch64.rb的笔误，并修复之。

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 实现LOONGARCH64Assembler汇编器

  实现Source/JavaScriptCore/assembler/LOONGARCH64Assembler.h。

  首次移植主要添加了常用的144条指令。

  通过跑testmasm回归测试，如果gdb触发SIGILL，说明操作数可能笔误。

  JSC使用枚举类定义操作数：

```
enum class Opcode7 : unsigned {
    LU12I_W_OP      = 0b0001010,
    LU32I_D_OP      = 0b0001011,
    PCADDU18I_OP    = 0b0001111,
};
```

龙架构 LoongArch Biweekly 双周会

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 实现LOONGARCH64Assembler汇编器

  使用模板定义指令类型：

  ```
  template<typename RegisterTypes>
  struct I20RTypeRegisters {
    static_assert(RegistersBase::Size<RegisterTypes>() == 1);
    using RD = RegistersBase::Type<0, RegisterTypes>;
  };
  ```

龙架构 LoongArch Biweekly 双周会

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 实现LOONGARCH64Assembler汇编器

  使用模板定义指令类型：

```
template<Opcode7 opcode, typename RegisterTypes>
struct I20RTypeBase {
    using Base = I20RTypeBase<opcode, RegisterTypes>;
    using Registers = I20RTypeRegisters<RegisterTypes>;

    template<typename RDType>
    static uint32_t construct(RDType rd, I20Immediate imm)
    {
        uint32_t instruction = 0
            | (unsigned(opcode) << 25)
            | (imm.field<0, 20>() << 5)
            | registerValue(rd);
        return instruction;
    }
```

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 实现LOONGARCH64Assembler汇编器

  使用模板定义指令类型：

```
template<Opcode7 opcode, typename RegisterTypes>
struct I20RTypeBase {
  static bool matches(InstructionValue insn)
  {
    return unsigned(opcode) == insn.field<25, 7>();
  }

  static uint8_t rd(InstructionValue insn) { return insn.field<0, 5>(); }
};

struct LU32I_D : I20RTypeBase<Opcode7::LU32I_D_OP, RegistersBase::G> {
  static constexpr const char* name = "lu32i.d";
};
```

龙架构 LoongArch Biweekly 双周会

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 实现LOONGARCH64Assembler汇编器

  为了防止笔误，LA不同于JSC其他架构，添加expected，判断汇编器生成机器码正确性：

```
void lu32i_dInsn(RegisterID rd, I20Immediate imm, uint32_t expected = 0)
{
    uint32_t mc = LOONGARCH64Instructions::LU32I_D::construct(rd, imm);
#ifdef NDEBUG
    UNUSED_PARAM(expected);
#else
    TEST_INSN(expected, mc);
#endif
    insn(mc);
}
```

龙架构 LoongArch Biweekly 双周会

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 实现LOONGARCH64Assembler汇编器

在Source/JavaScriptCore/assembler/MacroAssemblerLOONGARCH64.h添加testAssembler，

```
void testAssembler()
{
    m_assembler.lu32i_dInsn(LOONGARCH64Registers::r20, Imm::I20<int32_t>(524287),  0x16fffff4);
    m_assembler.lu32i_dInsn(LOONGARCH64Registers::r20, Imm::I20<int32_t>(-524288), 0x17000014);
    m_assembler.lu32i_dInsn(LOONGARCH64Registers::r20, Imm::I20<524287>(),  0x16fffff4);
    m_assembler.lu32i_dInsn(LOONGARCH64Registers::r20, Imm::I20<-524288>(), 0x17000014);
    m_assembler.lu32i_dInsn(LOONGARCH64Registers::r20, Imm::I20(524287),  0x16fffff4);
    m_assembler.lu32i_dInsn(LOONGARCH64Registers::r20, Imm::I20(-524288), 0x17000014);
...
}
```

龙架构 LoongArch Biweekly 双周会

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 实现LOONGARCH64Assembler汇编器

在Source/JavaScriptCore/assembler/testmasm.cpp调用jit.testAssembler

```
#if CPU(LOONGARCH64)
void testLoongArch64()
{
    auto test1 = compile([] (CCallHelpers& jit) {
        emitFunctionPrologue(jit);
        jit.testAssembler();
        emitFunctionEpilogue(jit);
        jit.ret();
    });
...
}
#endif
```

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 实现LOONGARCH64Assembler汇编器

首次移植主要调试testmasm（宏汇编器）可以通过单独执行

gdb -ex=r --args ./WebKitBuild/GTK/Debug/bin/testmasm调试LOONGARCH64Assembler汇编器和MacroAssemblerLOONGARCH64宏汇编器，包含319项测试用例。

因为Source/JavaScriptCore/assembler/testmasm.cpp使用多线程，若直接使用jit.breakpoint()代码生成break 0x5，然后gdb打印寄存器，si单步会受多线程影响，不方便观察汇编上下文，所以

gdb -ex=r --args ./WebKitBuild/GTK/Debug/bin/testmasm加上具体的测试用例名称，例如：testLoongArch64，这样si单步跟踪的时候不会受多线程影响。

WebKit JSC虚拟机LoongArch64移植

WebKit JSC虚拟机LoongArch64移植方法

- 实现MacroAssemblerLOONGARCH64宏汇编器

  实现Source/JavaScriptCore/assembler/MacroAssemblerLOONGARCH64.h。

  首次移植主要实现了常用的513个节点。

  印象深刻的

```
template<unsigned bitSize>
  JumpList branchAtomicWeakCASImpl(StatusCondition cond, RegisterID expectedAndClobbered, RegisterID newValue, BaseIndex address)
  {
      static_assert(bitSize == 8 || bitSize == 16);
      // There's no 8-bit or 16-bit load-reserved and store-conditional instructions AMCAS in 3A5000,
      // so we have to implement the operations through the 32-bit versions, with a limited amount
      // of usable registers.
```

龙架构 LoongArch Biweekly 双周会

WebKit JSC虚拟机LoongArch64移植

NumberTag

NumberTag标记，类似OpenJDK的ZGC的colored pointers，WebKit用NumberTag标记区分不同数据类型，参考Source/JavaScriptCore/runtime/JSCJSValue.h的注释：

```
 * The top 15-bits denote the type of the encoded JSValue:
 *
 *    Pointer {  0000:PPPP:PPPP:PPPP
 *              / 0002:****:****:****
 *    Double  {         ...
 *              \ FFFC:****:****:****
 *    Integer {  FFFE:0000:IIII:IIII
```

所以调试WebKit寄存器值发现高16位是0xfffe不用自我怀疑。

WebKit JSC虚拟机LoongArch64移植

NumberTag

static constexpr int64_t NumberTag = 0xfffe000000000000ll;

inline JSValue::JSValue(int i)

{

u.asInt64 = JSValue::NumberTag | static_cast<uint32_t>(i); // 编码方法

}

测试用例stress/to-int32-sensible.js常量池中的const INT32_MIN = -2147483648

被WebKit“编码”变成了0xfffe000080000000，对比没有被“编码”之前的值是0xffffffff80000000，按照0xfffe000000000000 | 0x80000000（无符号低32位）“编码”方法就变成了0xfffe000080000000。

WebKit JSC虚拟机LoongArch64移植

回归测试

回归测试覆盖JSC虚拟机的核心功能。覆盖：数组、布尔类型、日期类型、函数、全局对象、数学库、数据类型、对象、正则表达式、字符串。还覆盖解析器、词法约定、表达式、语句、类型转换和异常处理。

首次移植跑过testmasm功能测试，319项测试用例全部通过。跑过testapi测试，1453项测试用例全部通过。jsc-stress压力测试设置16分钟超时，限制最大堆大小，还有2项fail，其中1项因为WebAssembly没有实现FTL、B3、AIR、BBQ、OMG相关JIT编译器，45698 Running通过率99.97%（Running-FAIL）/ Running。

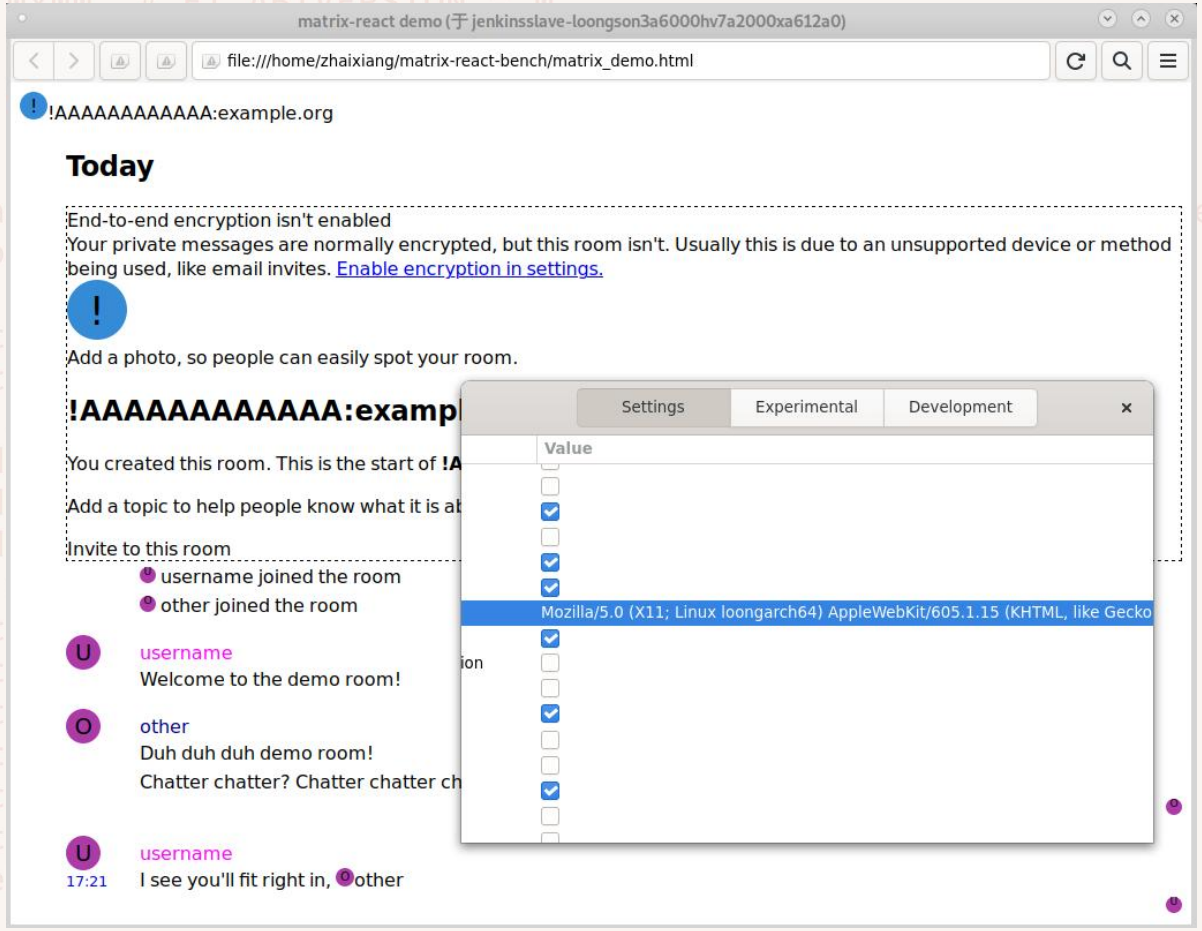
openjfx集成的WebKit测试用例全部通过 ok 8 - gradlew test web PASSED, tests: 472, failures: 0, ignored: 114, pass rate: 100%, duration: 1m25.43s。

Loongnix Desktop上直接替换jdk17里的lib/libjfxwebkit.so，跑WebViewSample。

WebKit JSC虚拟机LoongArch64移植

性能测试

在3A6000上离线跑了matrix-react-bench。



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