开放原子开源基金会
OPENATOM FOUNDATION

openKylin

高校开发者入门指南

从零到一的学习与实践手册



学生使用版

2026

基金会简介

认识开放原子

繁荣开源事业 · 共享开源价值

开放协同的科技创新中心

开放原子开源基金会于2020年6月成立，是我国首家国家级开源基金会，也是经民政部登记的慈善组织。基金会聚焦开源领域开展科技公益活动，致力于推动全球开源事业发展，推动科技平权。

社会的、公共的、国际化的“研发中心”和“成果中心”

通过开放获取、共建共创，汇聚项目、开发者、企业与高校等多方力量，支撑开源项目成长、人才培育、企业发展与产业生态构建。

共建	共享	共治	共赢

基金会简介

六类科技公益服务

为项目、开发者、高校与企业提供支撑

01 开源法律赋能

开展项目代码合规分析、社区治理章程拟定等咨询，推进开源许可协议研究、公益课程与法律交流。

02 开源项目运营

支持版本管理与认证测试、治理体系搭建、技术社区运营、工作坊、基础设施服务及开发者激励活动。

03 开源供需对接

通过专家解读、应用案例分享与供需洽谈，连接开源技术供给和行业场景需求，促进技术与产业融合。

04 开源品牌推广

依托生态大会、开发者大会、官网与全媒体矩阵，传播项目进展、生态成果及开源文化。

05 开源教育培训

提供开源通识与技术课程，支持开源课堂和社团建设，组织实践实训、培训认证，连接学习、实践与就业。

06 开源研究

组织开源报告、数据收集与课题研究，汇聚专家力量，支持理念普及与体系化知识传播。

基金会简介

让开源持续成长

项目、人才、企业与产业生态的共同支撑

01 开源项目的“孵化器”

提供项目准入、代码托管、治理规范、社区建设与品牌传播等支持，帮助项目规范运营、持续成长。

02 人才培养的“加速器”

以校源行育人才、以大赛练人才、以大会聚人才、以评价认证选人才，支持技术成长与开发者生态建设。

03 企业成长的“倍增器”

通过中立治理、开放协作与生态连接，促进共性技术研发、资源共享与合作，实现项目与企业共同成长。

04 产业生态的“连接器”

以明星开源项目为底座，连接产业需求与技术供给，推动开源技术在各行业、各领域落地应用。

科技公益服务涵盖开源法律、项目运营、供需对接、品牌推广、教育培训与开源研究。

基金会简介

项目与社区如何成长

开放治理、社区协作与生态连接相互支撑

从项目准入到持续演进

基金会围绕项目准入、代码托管、治理规范、版本发布、社区建设与品牌传播提供支持，帮助项目建立稳定的协作方式，并通过生态连接扩大技术影响力。

开源社区是开放协作网络

社区由围绕项目的开发者、用户及其他参与者组成。持续的贡献、讨论与反馈有助于提升项目活跃度、汇聚开发者、促进标准制定，支持项目长期发展。

以开放治理促进多方共建

通过治理章程、技术协作规则、社区组织与贡献流程，让不同参与者能够围绕共同目标开展研发、交流与维护，使项目成果持续积累。

基金会不仅培育开源项目，也通过治理、社区、生态与品牌支持，为基于开源成长的企业创造合作与发展机会。

基金会简介

找到你的技术方向

连接优秀开源项目与真实协作场景

openKylin

为世界提供与人工智能技术深度融合的开源操作系统。

OpenHarmony | 开源鸿蒙

面向万物互联的开源操作系统。

openEuler | 开源欧拉

面向数字基础设施的开源操作系统。

更多开源方向

高性能数据库OpenTenBase、人形机器人OpenLoong、具身智能开源数据集OpenLEFT等，为开发者提供参与技术创新与社区协作的机会。

AtomGit | 开源人工智能基础设施

使用代码与协作平台，开展项目实践，积累可验证的贡献记录。

基金会简介

人才培养的完整路径

以校源行育人才 · 以大赛练人才 · 以大会聚人才 · 以评价认证选人才

校源行：让开源融入校园

推广开源课程，建设师资队伍，支持开源社团，举办主题活动，引导师生在真实项目中学习与实践，为社区和企业建设人才生态。

开放原子大赛：在真实问题中锻炼

通过项目赛项和实践任务，连接技术研发、应用需求与创新场景，帮助参与者在协作中解决问题、展示成果、积累工程经验。

大会与交流：连接开发者和产业

通过开发者大会、生态大会、技术论坛与工作坊，促进前沿技术交流、社区文化互动与合作，拓展开发者的技术视野和协作网络。

评价与认证：让贡献成为成长依据

探索以真实贡献衡量能力，将学习实践与可验证成果连接起来，鼓励持续参与，为人才培养、能力展示和职业发展提供参考。

基金会简介

从校园走向开源社区

学习、实践、贡献，积累可验证的成长成果

开放原子“校源行”

基金会发起的长期公益性开源教育项目，通过开源课程、师资建设、社团支持、主题活动与实践，引导开源文化、理念和技术融入校园。

01 **选择方向，开始学习**

结合兴趣与基础选择项目，使用课程、指南和社区文档开展学习。

02 **完成实践，参与协作**

从一次安装、一个示例或一个协作任务开始，逐步参与文档、测试和代码贡献。

03 **持续贡献，记录成长**

加入开发者社区，通过真实贡献建立公开、可验证的成长记录。

以贡献为导向的人才评价与认证，鼓励师生参与开源、持续贡献。

基金会简介

高校师生如何参与

让课堂学习连接真实项目，让贡献记录见证成长

学生：从小任务开始贡献

结合兴趣和技术基础选择项目，阅读官方文档与贡献指南，从运行示例、文档完善、问题反馈、测试验证或代码修改开始，在社区协作中积累公开成果。

教师：将开源融入教学与实践

可围绕课程设计、项目实践与企业协同开展探索，将真实社区问题与教学任务结合，引导学生在协作、评审和持续改进中形成工程实践能力。

高校：探索以贡献为导向的评价

基金会联合专家探索适用于高校的开源人才评价机制，涵盖场景和标准、方法和数据、工具和案例，支持高校结合课程、项目和人才培养场景开展试点。

从这本指南开始：选择一个方向，完成一次实践，保留一份成果记录，再逐步参与真实社区、持续贡献。

先用10分钟确认你的起点

先确定自己的基础、兴趣与可投入时间，再选择一条路线。第一阶段只完成一个小成果：能运行、能复现、能说明或能提交。

现在的你	建议起点	第一件事
零基础或低年级	路线A：使用与测试	安装匹配镜像，完成一组桌面测试
有编程基础、喜欢应用	路线B：应用开发	跑通一个UKUI或openSDK示例
系统、嵌入式或硬件方向	路线C：系统与硬件	核对设备、架构、镜像和环境
想贡献但还不能写代码	第三部分：第一次贡献	从文档、复现、翻译或Issue开始
已有课程或毕业设计任务	第四部分：成长路径	把任务拆成可验证的公开成果

你的身份卡

姓名 / 学校：_____

专业 / 年级：_____

准备开始的方向：_____

每周可投入时间：_____

YOUR ROUTE 先选路线

找到你的第一条路线

先选一个可完成的小目标。路线可以调整，但第一次实践尽量固定一套系统、工具与任务。

我对什么感兴趣	推荐起点	下一步
界面、产品、应用与AI	UKUI、openSDK、AI SDK	第二部分路线B，运行一个官方示例
硬件、系统与多架构	系统组件、开发板适配	第二部分路线C，先保存基线日志
质量、复现与测试	桌面功能测试、QA	第二部分路线A，完成用例并保存证据
写作、翻译与社区协作	文档、Issue、翻译、布道	第三部分，确认一个真实小任务

□	我已选定一个方向，不同时切换多套工具、版本和硬件。
□	我已收藏项目官网、目标仓与贡献指南。
□	我知道个人实践、已提交和社区已采纳是不同状态。

使用说明

面向高校学生的社区认知、开发上手、第一次贡献与持续成长指南

本资源包为学习导航和实践指引，以openKylin官方在线文档、代码仓及社区最新规则为最终准则。

适用对象：对开源操作系统、Linux和桌面应用开发、生态适配、软件测试、开源协作感兴趣的本专科生、研究生和学生社团成员。设计、外语、传播等专业的学生，可以通过交互设计、国际化、文档和社区活动参与。非在校的开发者也可以作为入门参考。

版本口径：本版以openKylin 3.0作为学习基线。系统镜像、开发工具、软件包、SIG组（Special Interest Group，特别兴趣小组）、社区流程等持续迭代和更新，实操前请跳转openKylin在线资源。

平台口径：按当前openKylin社区官网开发入口，进入Gitee平台的代码仓库，签署贡献者协议CLA、源动力实习平台、软/硬件兼容适配、社区论坛交流、社区实习任务、积分成长体系、高校活动、高校专区和人才认证等社区所有平台：

openKylin社区平台一览

认识openKylin

资源	用途	网址
openKylin官网	社区介绍、新闻、高校专区和活动入口	https://www.openkylin.top
openKylin文档平台	使用、开发、规范与参与指南	https://docs.openkylin.top
社区行为守则	社区交流和协作要求	https://www.openkylin.top/join/conduct-cn.html
社区论坛	使用交流、经验与问题讨论	https://bbs.openkylin.top
Gitee代码仓	openKylin代码托管平台	https://gitee.com/openkylin
数字看板	openKylin运营情况统计看板	https://datastat.openkylin.top

体验openKylin

资源	用途	网址
ISO镜像下载	下载各架构、版本的openKylin镜像	https://www.openkylin.top/downloads
安装指南	虚拟机安装、实体机安装、分区建议等配套说明	https://docs.openkylin.top

加入openKylin

资源	用途	网址
openKylin ID	社区身份和个人信息	https://id.openkylin.top
CLA签署	贡献者许可协议	https://cla.openkylin.top
SIG列表	社区SIG小组汇集	https://www.openkylin.top/sig/siglist-cn.html
SIG邮件列表	订阅 SIG 动态、参与技术讨论	https://mailweb.openkylin.top
社区会员	组织（企业/高校）加入社区	https://www.openkylin.top/community/partner-cn.html
高校专区	openKylin走进国内140+高校	https://www.openkylin.top/dynamic/higher_education_zone-cn.html

贡献openKylin

资源	用途	网址
代码仓库	代码仓库管理、Issue和PR提交	https://gitee.com/openkylin
community治理仓	SIG、包归属和社区规则	https://gitee.com/openkylin/community
单包维护相关资料	软件包维护信息与流程	https://gitee.com/openkylin/community/tree/master/packages
开发者平台导航	编译、构建、CI、代码质量等openKylin官网入口	https://www.openkylin.top/developers/index-cn.html
软/硬件兼容适配平台	软/硬件兼容适配申请与展示	https://www.openkylin.top/support
软件包编译平台	软件包构建与日志	https://build.openkylin.top
发行版构建平台	发行版镜像构建相关工作	https://factory.openkylin.top
持续集成平台	自动化构建与检查	https://jenkins.openkylin.top
SIG会议	了解 SIG 运作节奏和参与讨论的公开渠道	https://www.openkylin.top/sig/meeting-cn.html
翻译平台	国际化、本地化协作	https://weblate.openkylin.top
源动力实习平台	社区实习任务平台	https://task.openkylin.top/dashboard
成长体系	查询成长等级与激励规则	https://www.openkylin.top/community/pgs-cn.html
人才认证	openKylin人才认证体系	https://www.openkylin.top/community/talent_certification.html

CONTENTS 目录

部分	内容	页码
PART 01	第一部分	openKylin高校开发者HandBook	9
PART 02	第二部分	openKylin开发上手手册	26
PART 03	第三部分	openKylin第一次社区贡献指引	43
PART 04	第四部分	openKylin高校开发者成长路径	52

页码对应本册正文页码；四个部分均设独立 PART 标题页。

TOOLS 工具与资源

工具与资源	用途	页码
社区平台一览	认识、体验、加入与贡献	4
附录A：7天入门计划	入门安排与每日行动	62
附录B：六张入门任务卡	任务输入、步骤与验收	64
附录C：可直接复制的模板	Issue、沟通、PR与成果记录	70
附录D：常见问题FAQ	15个新手问题	74
附录E：openKylin资源索引	官方资源与可点击网址	77
附录F：国内平台二维码	公众号、交流群等	80
附录G：国际平台二维码	Twitter、Linkedin等	81

PART 01

第一部分

openKylin高校开发者HandBook

用最短时间看懂openKylin，并找到适合自己的技术入口。

1. openKylin是什么

OpenAtom openKylin（简称“openKylin”）是由开放原子开源基金会孵化运营的开源项目。社区以“打造全球领先的智能操作系统开源根社区”为目标，通过开源、开放的方式构建合作伙伴生态体系，联合软硬件厂商、开发者等社区力量，共建覆盖架构适配、AI融合、桌面体验、应用兼容的智能操作系统。openKylin系统部署量已超 2400万套，服务全球 300万+用户，汇聚 30000+贡献者，拥有 2500+会员单位，建立 近160个SIG组 和 60个国际用户组，支持 100余种国际语言。核心桌面环境 UKUI 已被 12款国际主流Linux发行版 收录。openKylin社区连续三年获中国信通院“先进级可信开源社区”认证，获“央企十大国之重器”“CCF科技进步特等奖”等多项荣誉；“面向上海合作组织国家开展openKylin开源操作系统项目”入选外交部公布的上合成果清单，并获得 RISC-V基金会等国际行业组织认可，持续推动中国开源操作系统参与全球数字领域合作。

> 同学们可以理解为：
> 你无需“学完整个操作系统课程”就可以参与。 openKylin由Linux内核、UKUI桌面环境、AI子系统、基础组件共同构成。社区的100多个SIG，覆盖内核/硬件/驱动、桌面环境、开发框架、系统安全、人工智能、文档等多个技术方向。你可以从应用开发、文档编写、版本测试、多语言翻译、驱动适配、内核补丁、安全修复、AI能力集成等任一适合自己的入口开始——社区“源动力实习平台”每个月都会上架一批新任务，任务涵盖系统开发、版本测试、多语言翻译、创新适配、安全漏洞等多个领域。

openKylin项目：https://gitee.com/openkylin

代码仓库：https://gitee.com/organizations/openkylin/projects（项目的所有开源仓库）

源动力实习平台：https://task.openkylin.top/dashboard （社区的任务汇集，领取任务、提交成果、获得收获）

2. 五层技术架构：你可以在哪一层参与

五层技术架构（1/2）

层级	核心内容	适合的学生
应用层	Windows/Android/Linux 应用兼容运行环境（KMRE、KWRE、KARE运行环境）及 Deb/麒麟软件包格式支持。	前端、软件工程、应用开发初学者、对跨平台兼容感兴趣的学生。
桌面层	UKUI 桌面环境（基于 Qt 的现代桌面交互与系统级 UI 组件）。	计算机、软件工程、UI/UX 设计、Qt/QML 开发方向的学生。
系统服务层	openKylin SDK、WLCOM 显示服务、KART AI 子系统、原子更新服务、网络/音频服务等。	本科生、研究生、系统方向、AI 方向、后端开发方向。
内核与驱动层	Linux Kernel、Glibc、LLVM、GCC、Python、Go、Rust 等基础运行库、编译工具链与系统语言环境。	操作系统、计算机底层架构方向、对编译器或编程语言感兴趣的学生。
底座硬件层	CPU 架构支持（X86、ARM、RISC-V、LoongArch）及底层硬件驱动适配。	嵌入式、电子工程、自动化、物联网方向的学生。

五层技术架构（2/2）

层级	常见贡献	相关活动 / 大赛 / 沙龙
应用层	开发与移植、打包与适配、测试与反馈、文档与设计	专项赛事：Godot × openKylin全国开发大赛、麒麟杯全国开源应用软件开发大赛、Hack-Astron × openKylin挑战赛（软件包管理）、开放原子大赛（应用创新/贡献实践赛道） 高校通道：开放原子校源行、开源之夏、openKylin高校站沙龙、社区MeetUp
桌面层	代码开发、美化交互、测试与修复、文档与翻译	专项赛事：Hack-Astron × openKylin挑战赛（UKUI自动化配置）、开放原子大赛“基于openKylin桌面环境的多智能体交互技术赛”、麒麟杯大赛高校沙龙 高校通道：开放原子校源行、openKylin高校沙龙、UKUI SIG开发者分享会、社区MeetUp

层级	常见贡献	相关活动 / 大赛 / 沙龙
系统服务层	开发与封装、优化与运维、测试与工具、文档与示例	专项赛事：openKylin × 百度飞桨黑客松（AI+OS）、Hack-Astron（AI Agent技能开发）、第三届开放原子大赛“面向openKylin智能引擎的大模型推理优化赛”与“人工智能异构算力调度平台挑战赛”、中国研究生操作系统开源创新大赛（智能数据管理/系统服务优化） 高校通道：开源之夏、开放原子开源实习计划、openKylin开发者大会、AI/OS融合技术Meetup、社区MeetUp
内核与驱动层	内核与底层、编译与语言、测试与分析、文档	专项赛事：中国研究生操作系统开源创新大赛（内核/启动优化等赛题）、金灵光杯·麒麟软件“天马”原生生态开发赛道（内核特性与安全）、麒麟杯大赛（系统驱动管理模块） 高校通道：开源之夏、openKylin高校沙龙、Kernel SIG线上/线下技术研讨会

层级	常见贡献	相关活动 / 大赛 / 沙龙
底座硬件层	系统与硬件适配、驱动与调试、测试与认证、文档	专项赛事：开放原子开源大赛（RISC-V指令集与GPU优化适配）、中国研究生操作系统开源创新大赛（RISC-V硬件支持）、openKylin开源实践赛（嵌入式开发板体验） 高校通道：开放原子校源行、openKylin开源实践赛、RISC-V SIG / LoongArch SIG社区Meetup

延伸阅读：openKylin 社区总入口中的“系统架构”说明、各个 SIG的主页以及开源代码仓库的 README 索引。

3. openKylin的核心技术

3.1 核心组件选型构建策略

核心组件选型是操作系统根社区构建版本的基础工程，涵盖 kernel、glibc、gcc 等关键软件的版本基线。openKylin 建立六维评估模型（合规性、兼容性、重要程度、活跃度、质量、安全性），按 A/B/C/D 四级约束引入范围；版本策略以跟随上游稳定分支为主，兼顾发行版选型节奏与组件联动升级，主版本生命周期内不做大版本跃迁，仅以特性与补丁回合收敛质量风险。

SIG名称：Packaging

SIG介绍：负责维护openKylin社区的软件包打包规范，维护公共软件包，以及协调和决策社区版本发布过程中的包依赖问题。

访问地址：https://gitee.com/openkylin/community/tree/master/sig/packaging

3.2 X-Kernel 内核

X-Kernel 是面向可信执行环境构建的 Rust 语言内核，由 X-TeeOS SIG 维护，目前已实现多架构支持、任务与调度、虚拟内存与地址空间管理、VFS、网络协议栈、驱动框架及 POSIX 兼容接口。内核支持 AArch64、x86_64、RISC-V64、LoongArch64 四类架构，适配 QEMU、海光 CSV、pKVM 运行环境，并以类 Linux Xconfig 实现按平台与功能模块裁剪。

SIG名称：X-TeeOS

SIG介绍：旨在探索并落地一套面向关键业务场景的“轻量级、高可靠、可验证、永不中断”的可信执行环境。

访问地址：https://gitee.com/openkylin/community/blob/master/sig/X-TeeOS

3.3 KART AI 子系统及 KylinBot智能体

KART 是由AISubsystem SIG 主导开发的面向异构算力的 AI 子系统，向下屏蔽 x86、ARM、LoongArch 架构与 NPU、GPU、CPU 算力的差异。采用四层架构：接口层提供 AI SDK；运行时层划分基础能力服务与场景服务；统一推理框架层兼容多推理后端，负责资源协同与调度；模型层承载大语言、图像与向量化模型。KylinBot 在其上构建系统级智能体框架，将模型、工具、系统权限与桌面能力纳入统一运行时，经 MCP 协议调用桌面接口而非模拟输入事件，形成“推理—决策—操作—反馈”闭环。

SIG名称：AISubsystem

SIG介绍：负责AI子系统相关软件包的规划、维护和升级，管理本地和云端大模型，提供通用的AI能力。

访问地址：https://gitee.com/openkylin/community/tree/master/sig/AISubsystem

SIG名称：KylinBot

SIG介绍：KylinBot SIG小组致力于系统级AI智能体的规划、设计与开发工作。本SIG聚焦于将人工智能技术与操作系统深度融合，构建具备自主感知、决策与执行能力的智能体系统，为用户提供更智能、更便捷的桌面交互体验。

访问地址：https://gitee.com/openkylin/community/tree/master/sig/KylinBot

3.4 UKUI 桌面环境

UKUI是采用 Qt/C++ 实现的轻量化桌面环境，由 UKUI SIG 维护，覆盖开始菜单、文件管理、登录锁屏、任务栏、控制中心、会话管理等桌面组件集。技术架构以 UKUI Framework 为核心，将业务层与功能层解耦，功能以插件形式装配，由框架层统一收敛数据与配置；同时兼容 X11 与 Wayland 双显示协议，支持 PC 与平板模式切换。4.24 版本统一封装模块 API，引入 KACP 协议作为智能体访问桌面能力的标准通道，并新增隔空手势识别、全局语音输入、系统级屏幕朗读与任务栏三岛布局。

SIG名称：UKUI

SIG介绍：UKUI桌面环境相关软件包的规划、维护和升级工作

访问地址：https://gitee.com/openkylin/community/tree/master/sig/UKUI

3.5 全球化多语言支持体系

多语言能力由 I18N SIG 统一组织，以 openKylin 翻译平台作为中枢，构建“词条提取—自动化翻译—人工审核—双向同步”的高效翻译流程，保证各组件译文口径一致、随版本节奏收敛。在通用国际化之外，体系重点覆盖中文、英文及民族语言：联合高校完成藏语、蒙古语、维吾尔语、哈萨克语、柯尔克孜语、京语的系统级原生支持，并针对竖排书写、多编码并存适配输入法与字形渲染。3.0 新增朝鲜语、泰语、阿拉伯语、俄语、德语、日语、法语等 18 种语言，截至目前openKylin已支持100 余种国际主流语言，助力操作系统全球化落地。

SIG名称：I18N

SIG介绍：国际化和本地化相关工作，包括多语言开发框架，多语言平台开发和维护，社区和版本内词条、文档的翻译等。

访问地址：https://gitee.com/openkylin/community/blob/master/sig/i18n

3.6 WLCOM 显示框架

WLCOM 是基于 wlroots 研发的 Wayland 合成器（kylin-wayland-compositor），由社区Wayland SIG 主导开发和维护。WLCOM用于以替代架构陈旧、维护成本高的 X11 显示栈，同时弥补社区合成器在协议兼容与稳定性上的不足。实现上不依赖 Qt/GTK，降低依赖耦合；应用与合成器间协议按需定义，规避协议库版本错配；特效以插件形态加载。提供触摸板、触摸屏、数位板手势支持，并具备防截屏安全协议与 X11 应用桥接。

SIG名称：Wayland

SIG介绍：新一代图形显示服务器相关技术研究，包括wayland合成器、X兼容等。提供wayland相关软件包的技术规划、设计、开发、维护和升级服务，共同推动新一代图形服务器技术发展及落地。

访问地址：https://gitee.com/openkylin/community/tree/master/sig/Wayland#waylandsig

3.7 三大生态兼容技术

openKylin操作系统为满足用户对丰富的应用软件的使用需求，一直致力于构建跨平台的生态兼容能力，通过研发三大核心技术矩阵——KMRE、KWRE和KARE，openKylin实现了Android、Windows及Linux应用生态的无缝融合，打破了操作系统与应用间的壁垒。无论是移动端应用的深度“原生”适配、Windows软件的高效虚拟化运行，还是Linux老旧应用的版本兼容焕新，openKylin均以创新技术方案重塑了桌面应用生态的边界。

SIG名称：AOSP

SIG介绍：负责AOSP在云手机和安卓硬件生态设备上的适配开发、版本维护、源码仓库管理和开发手册编写等工作，致力于打造基于AOSP的开源智能终端操作系统。

访问地址：https://gitee.com/openkylin/community/tree/master/sig/AOSP

SIG名称：KARE

SIG介绍：KARE(Kylin Application Runtime Environment) SIG小组致力于完善openkylin系统兼容性。

访问地址：https://gitee.com/openkylin/community/tree/master/sig/KARE

SIG名称：CompatWinApp

SIG介绍：研究应用兼容技术和指令翻译技术，研制完善的windows应用兼容方案，让更多的windows应用能兼容运行于openKylin 系统。

访问地址：https://gitee.com/openkylin/community/blob/master/sig/CompatWinApp

3.8 RVtrans 二进制翻译技术

RVtrans技术以提高软件兼容性能和效率为核心，研究动静结合的二进制翻译优化技术，实现软件的快速翻译执行；基于相关性原理对动态指令翻译进行优化，研究基于时空局部性的指令Cache管理策略和基于地址关联的跳转指令翻译优化方法；研究动态库本地化映射封装和软件隔离兼容运行技术，以及面向RISC-V的兼容解决方案，实现X86架构软件在RISC-V平台上高效兼容运行。该项目由社区RISC-V SIG组维护。

SIG名称：RISC-V

SIG介绍：发布openKylin的RISC-V 版本，进行软件包构建、系统构建等工作。

访问地址：https://gitee.com/openkylin/community/blob/master/sig/RISC-V

3.9 openSDK 及 KylinCode 集成开发环境

openSDK 是面向 openKylin 桌面操作系统的开发者套件，划分为应用支撑、系统能力、基础开发三个模块：应用支撑 SDK 提供自研控件与统一 UI 框架，规范窗口管理与系统交互接口；系统能力 SDK 开放系统、硬件与运行时信息访问能力；基础开发 SDK 封装日志、字符串处理等通用组件，保持跨版本接口兼容。配套 AI SDK 以统一接口适配多家大模型。KylinCode 基于 Code-OSS 开发，补齐 LoongArch、申威等国产平台支持，提供本地与远程双模式，远程侧由 remote-dev 插件与 kylin-ide-server 实现跨架构开发。

SIG名称：openSDK

SIG介绍：负责开发者套件（base、system、applications）规划、开发、维护等工作，致力于解决应用在多操作系统中的兼容性问题。

访问地址：https://gitee.com/openkylin/community/tree/master/sig/OpenSDK

SIG名称：Framework

SIG介绍：致力于为openKylin社区提供集程序编辑、编译、调试、发布、分析等全套开发功能的编程环境，涵盖通用集成开发环境、编译工具链、运行时环境、类库等。

访问地址：https://gitee.com/openkylin/community/tree/master/sig/framework

3.10 后量子密码算法布局

面对“先存后破”的量子威胁，openKylin 将后量子迁移作为系统工程推进，由 PQCTransition SIG 牵头算法工程化与系统级替换。SIG 已按 NIST 首批标准落地三个代码实现：kyber-package 对应 ML-KEM，用于 TLS 握手等场景的密钥封装；dilithium-package 对应 ML-DSA，基于 Module-LWE 提供抗量子数字签名；sphincs-package 对应 SLH-DSA，安全性仅依赖哈希函数，作为签名安全底线。3.0 将三项标准集成入系统，由 Security SIG 引进 openHiTLS 密码套件提供系统级调用，兼容 SM2/SM3/SM4 国密算法，并与 TPM 可信启动链衔接，形成算法、套件与信任根三层防线。

SIG名称：PQCTransition

SIG介绍：致力于推动后量子密码（PQC）算法的工程化和创新应用，以应对量子计算带来的安全挑战。

访问地址：https://gitee.com/openkylin/community/tree/master/sig/PQCTransition

SIG名称：Security

SIG介绍：负责研制openKylin操作系统安全防护技术和功能，提供并维护操作系统安全防御机制等。

访问地址：https://gitee.com/openkylin/community/tree/master/sig/security

如需了解 openKylin 社区更多核心技术点，可访问 https://www.openkylin.top/public/pdf/OpenAtom_openKylin_Community_Panorama_Case_Collection_2025.pdf 获取全景技术案例集。

4. 十类高校学生参与方向

以下是选择建议；实际任务和负责仓库以SIG资料为准。

十类高校学生参与方向（1/2）

方向	建议前置知识	相关SIG
社区运营、版本发行与Issue管理	基本 Git 协作、沟通与项目管理意识	Community、Release、FAQ等
文档与国际化	Markdown、Git 基础、技术写作	Docs、I18n、InternationalPromotion等
开源布道与社区推广	良好的沟通表达能力、基础图文/视频剪辑能力、对openKylin的基础认知	Community、I18n、InternationalPromotion
软件生态适配	Linux基础安装、熟悉openKylin软件商店	Compatibility、HWTestKit
AI 与智能体	Python、机器学习基础、大模型 API 调用	AISubsystem、AgentOS、KylinBot等
系统安全	安全漏洞治理、网络安全、Linux 权限模型、密码学基础	SecurityGovernance 、Security、NetworkSecurity、HygonSecurity、PQCTransition、TrustedComputing等
内核与硬件适配	C/汇编、操作系统原理、计算机体系结构	Kernel、X-TeeOS、Compatibility、OpenLoong、Embedded、ARM、RISC-V、LoongArch、Server等

方向	建议前置知识	相关SIG
图形与显示	C、图形学基础、Wayland 协议概念	Wayland、GPU等
测试与质量	软件测试理论、Python/Shell脚本	QA、HWTestKit等
打包与工具链	Linux 命令行、Deb 打包基础、构建系统	Packaging、Compiler、openSDK、Qt等

十类高校学生参与方向（2/2）

方向	第一实践	可持续贡献方向
社区运营、版本发行与Issue管理	在 Gitee 仓库认领一个 good first issue，跟进复现并协助维护者整理复现步骤	Issue 分类与复现、PR 初审、SIG 会议记录、社区任务平台运营、新手引导
文档与国际化	修订一处 SIG README 或用户文档，或参与一个软件包的界面/文档翻译	SIG 文档维护、安装指南、FAQ、多语言翻译、新手教程与示例仓库
开源布道与社区推广	将自己的“0-1入坑过程”（如：如何在虚拟机安装openKylin、如何提交第一个Issue）写成一篇图文教程，发布在CSDN、知乎或B站，并带上社区标签	内容创作、校内宣讲、社群答疑
软件生态适配	在1台实体机或开发板上安装openKylin，将所需要适配的软件在系统上安装、功能测试，记录测试的状态和情况，提交Issue/PR	兼容层应用适配测试、兼容问题排查

方向	第一实践	可持续贡献方向
AI 与智能体	基于 KART AI 子系统调用一个本地或云端模型，跑通文字识别或简单问答示例	KART 运行时接口开发、kytensor 推理引擎、KylinBot 智能体技能、AI 应用集成
系统安全	在不可变系统中验证权限隔离机制，为软件包配置沙箱资源权限	沙箱策略优化、权限管控、安全漏洞挖掘、不可变系统安全加固
内核与硬件适配	在 RISC-V/LoongArch/ARM 开发板或虚拟机上编译内核模块或设备驱动，完成一次 dmesg 排查	内核补丁、SoC 平台适配（飞腾/龙芯/海光等）、RISC-V 驱动、开发板硬件支持
图形与显示	阅读 WLCOM 合成器代码结构，修改一个插件（如特效或窗口管理插件）并编译验证	WLCOM 合成器插件、GPU 驱动适配、视频编解码、渲染性能优化
测试与质量	使用 HWTestKit 或 openQA 在真机/虚拟机上跑一轮安装或硬件兼容测试，复现一个已知 Issue	自动化测试脚本、硬件兼容矩阵、GUI 自动化、缺陷回归与质量看板
打包与工具链	将一个简单应用打包为软件（deb/appimage 转换），在 openKylin 上安装并配置沙箱权限	软件包格式工具链、多格式转换、openSDK 接口、CI/CD 构建流程

5. 如何找到SIG和真实仓库

SIG是 openKylin 社区按技术领域划分的协作小组，负责技术规划、仓库维护、公开例会和社区成员发展。学生可以先按目标仓规则参与Issue、PR或公开讨论；正式SIG成员身份、会议参与方式与仓库权限分别遵循对应流程。参与时，可先从官网 SIG 列表进入，再查看 community 仓库中对应目录下的 README、sig.yaml，以及该 SIG 所维护软件包的信息。SIG中心

建议按下面的顺序定位任务：

（1）从技术方向选择一个SIG。

（2）阅读它的目标、维护范围、讨论方式与近期活动。

（3）从SIG资料进入目标仓库，阅读README、许可证和贡献说明。

（4）查看最近的Issue与PR，判断任务是否仍有效、是否有人处理。

（5）选一个可以在1—2周内产生结果的小任务，与维护者确认范围。

SIG中心入口：https://www.openkylin.top/sig/siglist-cn.html

创建/加入SIG：https://www.openkylin.top/sig/application-cn.html

社区治理仓库：https://gitee.com/openkylin/community

6. openKylin的资源怎么用

要做的事	优先使用的入口
社区总入口	社区官网
下载系统、查镜像和安装资料	下载中心
查使用、安装、开发教程	文档平台
找代码、Issue和PR	Gitee组织
查SIG与软件包维护关系	community仓
注册社区身份	openKylin ID
签署贡献者协议	CLA平台
交流使用问题	社区论坛
查构建、CI、翻译等平台	开发者平台导航
找实习任务与成长信息	源动力实习任务、成长体系

PART 02

第二部分

openKylin开发上手手册

先下载openKylin镜像并安装，完成一个可运行成果，再进入更复杂的源码和社区贡献。

1. 环境准备

1.1 环境准备、下载与安装

环境	适用场景	注意事项
虚拟机	Linux入门、基础开发、文档与通用测试	镜像架构须与宿主CPU一致（x86主机下载AMD64/X86镜像）；虚拟机至少分配4GB内存、50GB硬盘（流畅体验建议8GB内存、128GB硬盘）；VMware环境下不要装自带vmware-tools，改用源里的open-vm-tools，否则可能分辨率异常、无显卡加速；VirtualBox需勾选“启用EFI”并开启3D加速；安装完成后保留初始快照。

环境	适用场景	注意事项
实体PC	日常使用、显示、音频、外设和电源测试	• 安装前需确认硬件兼容性，当前openKylin支持（x86/ARM/RISC-V/LoongArch架构）,根据自己设备的架构搜索对应的ISO镜像并下载； • 实体PC机安装openKylin的推荐配置如下： CPU：2018年以后的主流产品（6核及以上）。 内存：8GB及以上（虚拟机建议分配4GB及以上）。 硬盘：128GB及以上空闲存储空间。 • 用Rufus或Ventoy制作启动盘，U盘不小于8GB；进入BIOS后关闭Secure Boot，否则可能无法从启动盘引导； • 单系统安装根分区（/，ext4）建议不低于15GB，长期使用建议50GB以上； • 双系统安装务必手动分区，不要覆盖原有EFI分区和系统分区；安装前备份数据，明确目标磁盘。

环境	适用场景	注意事项
开发板	RISC-V、ARM等硬件实验	必须使用openKylin官网上对应板卡发布的专属镜像：ARM可用树莓派、飞腾派PhytiumPi、Cool Pi、Chillie Pi等，RISC-V可用VisionFive 2、LicheePi 4A、Milk-V Pioneer、SiFive HiFive Unmatched等；部分RISC-V板卡镜像暂不包含UKUI桌面环境（如LicheePi4A早期嵌入式镜像），需确认镜像形态是Desktop还是Embedded/Server；严格按对应板卡的烧录与启动说明操作。

1.2 openKylin 安装五步法（实体PC）

第1步：下载并校验镜像

前往openKylin官网下载对应架构（如x86_64）的ISO镜像。下载后务必校验MD5或SHA-256值，确保与官网一致，避免因镜像损坏导致安装失败。

第2步：制作启动盘

准备一个容量不小于8GB的U盘，使用Rufus或Ventoy等工具将ISO镜像写入U盘，制作成可启动的安装介质。

第3步：从启动盘引导电脑

将启动盘插入电脑，重启时根据屏幕提示（通常是F2、F10、F12或Delete键）进入BIOS/UEFI设置，将U盘设置为第一启动项，保存并退出，电脑将从启动盘加载安装环境。

第4步：开始安装并配置

进入试用桌面后，点击“安装openKylin”图标，按照图形化向导依次完成语言选择、时区设置、分区方案（单系统可默认或手动规划/分区）和用户账户创建等配置。如果是双系统安装，建议手动分区，并注意不要误删原有系统的分区。

第5步：完成安装并重启

等待系统文件复制和配置完成后，安装程序会提示安装成功。拔掉U盘，点击重启，电脑将从硬盘引导进入全新安装的openKylin系统，完成初始设置即可开始使用。

虚拟机安装指南：https://docs.openkylin.top/zh/1入门与参与/1_3系统下载与安装指南/01_openKylin系统下载与安装指南/6_virt-manager&openKylin-x86虚拟机安装指南

开发板安装指南：https://docs.openkylin.top/zh/1入门与参与/1_3系统下载与安装指南/03_在riscv上安装/在LicheePi4A上安装openKylin

在ARM上安装指南：https://docs.openkylin.top/zh/1入门与参与/1_3系统下载与安装指南/02_在arm设备上安装/arm上安装openKylin

在X86上安装指南：https://docs.openkylin.top/zh/1入门与参与/1_3系统下载与安装指南/01_openKylin系统下载与安装指南/3_openKylin安装指南

2. 三条上手路线

路线	适合对象	第一阶段成果	需要什么设备
A. 使用、文档与测试	零基础、低年级、非计算机专业	一份可复现测试记录或教程改进	普通电脑与匹配架构的虚拟机即可起步
B. 应用开发与软件包	有基础编程能力的学生	一个可运行小程序，或一个可重复构建的包	普通开发电脑；按镜像要求分配资源
C. 系统组件与硬件适配	本科、研究生、系统或嵌入式方向	组件构建、缺陷回归或硬件适配报告	部分任务可用虚拟机，硬件任务需对应设备

三条路线都能进入社区贡献。第一次学习只选择一条主路线。

3. 路线A：第一次使用与测试实践

3.1 推荐实践：完成一份桌面基础功能测试单

选取文件管理器、文本编辑器或另一个已安装的普通应用，围绕一个功能进行测试。以下为教学用例建议，并非已确认存在的缺陷。

测试场景	操作	观察与记录
基本使用	启动应用，创建并保存一个文件	是否成功，保存位置与内容是否正确
中文场景	使用中文文件名，输入中文内容	文件名、输入法与内容显示是否正常

测试场景	操作	观察与记录
边界输入	输入空值、较长名称或重复名称	是否有明确提示，是否发生数据丢失
重启恢复	关闭后重新打开同一文件	内容是否保持，窗口行为是否合理
桌面适配	在支持的缩放设置下检查界面	文本截断、按钮遮挡、布局是否异常

3.2 从练习走向贡献

未发现问题：形成测试报告，按当前共测活动要求提交；也可以继续复现已有Issue。

发现真实问题：先检索已有反馈，重复问题补充环境和证据，新问题提交到对应组件仓。

不知道归属：先阅读QA参与指南，通过QA渠道或论坛帮助定位。

发现教程与实际不符：记录原文位置、系统版本、实际表现和建议改法。

入口：QA测试参与指南、文档仓。

完成标准：另一名同学能够按照你的记录重复测试。没有发现缺陷时如实写“本次范围内未发现”，不为提交Issue而制造问题。

4. 路线B：应用开发快速上手

4.1 上手目标

□	能够打开 openKylin 应用开发快速入门文档，并确认当前工具链与 openSDK 要求。
□	能够使用 Kylin-Code 或 Qt Creator 创建/导入一个 openKylin 应用工程。
□	能够看懂 openSDK 的应用支撑模块（Qt自研控件、Wayland-helper）与基础开发模块（日志、配置、d-bus）的基本结构。
□	能够编译、安装并运行一个官方 Sample 或软件包示例。
□	能够修改一个页面、数据或交互，并使用可视化构建工具重新打包验证结果。
□	能够记录环境、步骤、问题和解决办法，为后续 Issue 或 PR 做准备。

4.2 推荐的六步流程

第1步：确认版本与 SDK

从 openKylin 官方开发文档确认推荐的 openSDK 版本、AI SDK 版本及对应的系统环境（openKylin 2.0 SP1 及以上已默认集成 AI 子系统）。openSDK 聚焦应用支撑、系统能力、基础开发三大模块，需提前添加 openSDK 软件源。

第2步：完成工具准备

安装 Kylin-Code（社区 Framework SIG 维护的 IDE，openKylin 1.0 已预装，也可在软件商店或 Gitee Releases 获取），或安装 Qt Creator 进行 Qt/UKUI 开发。安装插件依赖管理器，并配置 GCC、CMake、Git 等基础构建环境。

第3步：选择示例，零基础优先

优先选择单一功能的入门示例：AI SDK 的文本对话示例（libkysdk-genai-nlp-dev，涵盖 GLib 主循环、模型初始化、异步聊天）、软件 HelloWorld 脚本示例（最小软件包结构）、UKUI-quick 桌面小插件（Qt/QML 入门），或 UKUI 官方文档中的“开发第一个应用”教程。

第4步：运行原始版本，不要立即修改

先完整编译并运行示例，记录原始版本的编译结果、运行界面和主要功能。

第5步：只改一个点

修改一处文字、布局、数据、交互或一个 API 调用，确保能准确解释改动。openSDK 的应用支撑模块（Qt 自研控件、Wayland-helper）和基础开发模块（日志、配置）都是合适的切入点。改动后重新编译或重新打包，观察行为变化。

第6步：形成成果记录

保存截图、运行视频、代码差异、问题清单和 README，作为作业、作品或贡献材料。如果示例来自 openKylin 仓库，可将记录整理为 Issue 描述或 PR 说明。

参考链接

openKylin 开发文档总入口：https://docs.openkylin.top

openSDK 开发指南：https://docs.openkylin.top/zh/04_社区贡献/开发指南/openKylin+SDK开发指南

openKylin AI SDK 开发手册：https://www.openkylin.top/public/pdf/openKylin_AI_SDK.pdf

Kylin-Code 仓库：https://gitee.com/openkylin/kylin-code

openKylin AI SDK 仓库：https://gitee.com/openkylin/kylin-ai-sdk

UKUI 官方文档（含“开发第一个应用”教程）：https://www.ukui.org

4.3 初学者示例选择表

目标	建议示例类型	建议改造任务
学习 UKUI / QML 与桌面交互	ukui-quick 数字时钟插件（org.ukui.myplugin 结构，main.qml 实现界面，metadata.json 声明插件信息）或 UKUI 开始菜单插件 Demo（继承 MenuExtensionPlugin 接口）	修改 QML 布局、增加主题切换或为开始菜单插件增加一个简单的状态提示（如“提醒事项”的完成数显示）；在 metadata.json 中调整插件元数据，观察 UKUI 的加载行为
学习数据管理	SQLite / Qt SQL 本地数据示例：在 openKylin 上使用 libqt5sql5-sqlite 插件，创建一个待办列表或简单笔记应用，将数据持久化到本地 SQLite 文件	为示例增加搜索过滤、分类标签或导入/导出功能；观察 Storage/SqlitePath 等配置项对数据库文件位置的影响，理解单文件嵌入式数据库的管理方式
学习网络通信	Qt Network 示例：使用 QNetworkAccessManager 发起 HTTP GET/POST 请求，或使用 QUdpSocket / QTcpSocket 实现基础回显服务器	为网络请求增加加载状态、失败重试与超时处理；若示例涉及数据拉取，尝试将结果缓存到本地 SQLite，并与原始网络路径做对比
学习多媒体与桌面集成	Qt Multimedia 简易播放器：基于 QMediaPlayer + QVideoWidget / QAudioOutput 实现本地音视频播放，或在 UKUI 环境下调用 openKylin 自带的 kylin-video（MPV 前端）作为参考	增加播放进度控制、音量/静音切换、播放列表或截图功能；若在 Wayland 会话下测试，记录与 X11 环境下多媒体渲染行为的差异

目标	建议示例类型	建议改造任务
学习安全、沙箱与打包	软件包最小示例（top.openkylin.Hello.yaml + hello.sh，理解 runtime、command、sources 等清单字段）	
学习跨设备/分布式协作	openKylin 社区 Space SIG 或“互联协同”方向的相关仓库/SIG 文档	从 SIG 的 README、Issue 或设计文档入手，记录一项协同场景的需求或问题；若示例尚不成熟，可将重点放在“理解架构 + 复现环境 + 整理问题清单”上，作为后续贡献的起点

4.4 应用开发成果模板

内容	你需要填写
项目名称	例如：基于 openKylin 的 UKUI 桌面数字时钟插件 / 基于软件包沙箱的校园待办应用
学习基线	openKylin 版本（如 2.0 SP2 / 3.0）、openKylin SDK 版本（如 SDK V3.0 / openSDK 3.x）、示例来源链接（Gitee 仓库或社区教程链接）
完成内容	运行了什么（如 ukui-quick 数字时钟、软件 Hello 包）；修改了什么（QML 布局、metadata.json 权限清单字段）；新增了什么（如搜索过滤、播放控制、权限声明）
技术要点	使用的组件/接口（如 ukui-quick 的 platform 模块、MenuExtensionPlugin 接口、QMediaPlayer的 yaml 清单）；关键代码或配置片段
验证结果	截图 / 运行视频 / 测试步骤；已知限制（如 Wayland 下插件加载差异、RISC-V 板卡上多媒体解码器缺失、软件包沙箱对文件访问路径的限制等）
下一步	准备提交 Issue 或 PR（注明对应 SIG，如 UKUI SIG、QA SIG）；补充 README 或示例文档；修复发现的问题或开发小功能

入口提示：openKylin 的初学者示例不必从零"造一个完整应用"开始。UKUI 插件（ukui-quick-widget / 开始菜单插件）和软件包 Hello 示例都是结构清晰、边界明确的第一实践对象。完成一次"运行 → 改一点 → 重新打包/加载 → 记录"的闭环，比追求功能规模更重要。

5. 路线C：系统组件与硬件适配实践

5.1 上手目标

□	能够确认当前 openKylin 系统的内核版本、架构（X86 / ARM / RISC-V / LoongArch）与已加载硬件驱动状态（uname -a、lsmod、lspci -k、dmesg）。
□	能够从 openSDK 中找到一个系统能力接口（如硬件信息获取、电源管理、打印机管理）或一个 UKUI-quick 插件示例，理解其调用链路与编译方式。
□	能够在一款明确适配的开发板（如 LicheePi 4A、VisionFive 2、Milk-V Pioneer、SpacemiT K1）上完成镜像烧录与首次启动，或能在实体 PC / 虚拟机上复现一项硬件相关 Issue。
□	能够使用 dmesg、lsmod、modinfo 等工具定位一个驱动加载或硬件识别问题，并整理出可复现的日志片段。
□	能够修改一个系统组件的最小闭环（如 UKUI-quick 插件的显示逻辑、openSDK 硬件信息接口的调用方式），并验证改动前后的行为差异。
□	能够将环境、烧录/编译步骤、dmesg 日志、问题与解决过程整理成可提交给 Kernel / GPU / UKUI / QA 等 SIG 的 Issue 或 PR 描述。

5.2 推荐的六步流程

第1步：确认目标方向与硬件基线

确定你要做开发板（如 LicheePi 4A）、实体 PC 还是系统组件（openSDK/UKUI）。敲入 uname -a、lspci 等命令，记录当前硬件环境状态。

第2步：完成环境与工具准备

硬件方向：去官网下镜像，用 dd 或专用脚本把系统烧录到板子/硬盘。

组件方向：装好 openSDK、Qt 或 ukui-quick 的开发包。

统一装好 dmesg、lsmod 等调试工具。

第3步：选择一个最小示例，先跑通再修改

硬件方向：把板子点亮进桌面，或用 dmesg 查有没有报错。

组件方向：编译并运行一个现成的 UKUI 小插件或 API 工具。
（注意：新手绝对不要第一步就去改内核核心代码）

第4步：运行原始版本，记录基线行为

记录原始状态的完整数据。硬件方向保存 dmesg > boot.log 启动日志；组件方向保存编译截图、接口返回值。这是后续对比的基准。

第5步：只改一个点，确保能解释因果

只做一个可逆的小改动。硬件方向试试卸载/加载一个模块（modprobe -r）；组件方向改改插件的一行文字或接口调用。对比改动前后的日志变化。

第6步：形成可提交的成果记录

把烧录日志、截图、代码 diff 整理好。如果是发现了 Bug，就去对应的 SIG（如 Kernel SIG、UKUI SIG）仓库提 Issue；如果是做出了改进，就提 PR。

5.3 初学者示例选择表

目标	建议示例类型	建议改造任务
学习硬件启动与镜像烧录	选一款开发板（如 LicheePi 4A），完成官方镜像烧录与首次启动。	记录烧录工具与启动设置；对比不同存储介质（SD卡/eMMC）的启动日志差异。
学习驱动加载与硬件识别	用 lsmod、dmesg 等命令排查网卡、显卡等外设的驱动状态。	找一个报错的外设，记录其 ID 与模块名；尝试手动 modprobe 加载并对比结果。
学习 openSDK 系统能力调用	调用 openSDK 的“硬件信息”或“磁盘信息”等接口。	用命令行工具或 QML 面板展示硬件信息；增加返回空值时的异常提示。
学习 UKUI-quick 系统级插件	运行 UKUI-quick 桌面小插件（如系统信息展示）。	修改插件的显示内容或刷新频率；记录涉及的接口权限与沙箱限制。
学习内核模块基础	编写一个最小 hello 内核模块，完成编译、加载与卸载。	增加 module_param 传参来改变模块行为；记录编译头文件的来源路径。
学习硬件兼容性测试	用社区工具（hw-detect 等）对电脑或开发板采集外设清单。	整理各硬件驱动状态，输出兼容性报告；把异常项作为 Issue 提给 QA SIG。

5.4 应用开发成果模板

内容	你需要填写
项目名称	例如：LicheePi 4A openKylin 镜像烧录与启动日志分析 / 基于 openSDK 的硬件信息桌面面板 / 某型号网卡在 openKylin 下的驱动加载排查
学习基线	openKylin 版本（如 2.0 SP2 / 3.0）、内核版本（uname -r）、硬件平台（X86 / ARM 板卡型号 / RISC-V 板卡型号 / LoongArch）、镜像来源、编译/烧录工具版本
完成内容	运行了什么（镜像启动 / UKUI 插件 / openSDK 工具）；修改了什么（插件显示字段 / 模块参数 / 接口调用方式）；新增了什么（异常处理 / 日志采集脚本 / 基线对比表）
技术要点	涉及的 SIG 方向（Kernel / GPU / RISC-V / UKUI / QA）、关键命令（dmesg、lsmod、lspci -k、modprobe、dd）、关键接口（openSDK 系统能力 SDK 函数名、UKUI-quick 组件名）、内核模块名称或 PCI/USB ID
验证结果	启动截图 / 串口日志片段 / dmesg 关键行 / lsmod 前后对比 / 插件运行截图；已知限制（如某开发板上 HDMI 音频缺失、Wayland 下插件渲染差异对 modprobe 的权限限制等）
下一步	准备向对应 SIG 提交 Issue（附 dmesg 与复现步骤）或 PR（附 diff 与验证记录）；补充 README / 烧录指南 / 硬件兼容性报告；若发现的是社区已知问题，转为整理复现步骤与日志，协助维护者定位

PART 03

第三部分

openKylin第一次社区贡献指引

把学习中的发现变成一个可以评审和复用的成果。

1. 贡献不只有代码

贡献类型	新人可以怎样开始	成果证据
问题反馈与复现	报告真实缺陷，补充已有问题的环境和步骤	Issue、复现材料、维护者反馈
文档与FAQ	修正安装步骤、失效链接或版本说明	PR、文档修改与发布记录
测试与质量	补用例、参加版本共测、验证修复	用例、报告、回归结论
翻译与本地化	在具体界面上下文中校对字符串	翻译审核记录、界面验证
设计与体验	针对具体场景提出有依据的改进	设计源文件、方案讨论、采纳记录
软件包与代码	修复小缺陷、依赖或构建问题	PR、构建与测试、合入提交
社区协作	有依据地答疑、组织共学和分享	可复用材料、活动与参与者成果

选一种最适合当前能力的方式，先完成一次反馈与改进过程。

2. 第一次贡献十步法

第1步：注册并确立身份

登录 openKylin 社区官网完成账号注册，设定长期使用的开发者昵称与常用邮箱，并按照平台指引完成基础信息的关联与绑定。

第2步：签署贡献者许可协议（CLA）

务必基于真实身份查阅并签署对应的 CLA。如果是学生以个人名义参与，请留意并选择个人 CLA 的专门入口完成签署。

第3步：熟悉社区规范

仔细阅读社区的《行为准则》，并浏览你将要贡献的目标仓库中的 README、CONTRIBUTING 指南、Issue/PR 模板以及代码编写规范。

第4步：锁定合适的任务

根据自身情况选定 SIG 组和代码仓库，认领任务时遵循“小而精”的原则（目标清晰、易于验证，预计1-2周内能搞定）。切忌未提前沟通就开展大规模代码重构。

第5步：沟通确认再动手

在相关的 Issue 下方留言，阐述你对问题的理解、初步方案和执行计划；等待维护者确认该任务仍处于未认领状态，且你的思路方向可行。

第6步：新建独立开发分支

完成 Fork 或 Clone 操作后，切出一个专用的工作分支，避免直接在主干分支上修改；并确保每次提交的代码仅聚焦于当前任务，保持范围单一。

第7步：本地修改与自测

完成代码编写后，严格遵循仓库要求执行编译、代码静态检查、单元测试或场景验证，并妥善保存运行结果。

第8步：提交规范的合并请求（PR）

提交时务必关联对应的 Issue，并清晰写明改动背景、具体修改点、验证结果及潜在风险。提交前检查 CLA 声明、Commit 信息格式，并剔除无关文件。

第9步：积极应对代码评审

认真阅读并逐条回复评审（Review）意见；若对某些建议存在异议，请基于客观事实说明理由，保持开放心态，不要把评审意见当成对你个人的否定。

第10步：跟进合入与复盘总结

确认代码被成功合入、相关 Issue 已关闭或状态已更新；保存本次贡献的链接，记录 Review 过程中的收获，并规划自己的下一项任务。

3. CLA与Git提交信息怎么处理

openKylin使用CLA贡献协议。在线CLA签署、代码平台身份和本地Git作者信息应按平台要求核对，不能用一次git commit -s代替CLA签署。

```
在自己的项目仓库中设置作者信息：
git config user.name "你的开发者名称"
git config user.email "你的贡献邮箱"
git config --get user.name
git config --get user.email
```

4. 通用Git协作流程

```
下面是流程模板，中文占位内容必须替换，不能整段原样执行：
git clone <你的Fork仓库地址>
cd <仓库目录>
git remote add upstream <社区原仓库地址>
git fetch upstream
git switch -c fix/first-contribution upstream/<维护者确认的目标分支>
```

修改文件，并按仓库要求完成验证：

```
git diff
git status
git add <本次修改的文件>
git diff --cached
git commit -m "按仓库规范简述本次修改"
git push -u origin fix/first-contribution
```

随后在代码平台选择正确的源分支和目标分支创建PR。文档仓与软件包仓的流程可能不同，例如文档可能先合入开发分支；不要只依赖平台默认选择。

5. 第一个任务从哪里来

来源	找任务的方法	开始前确认
真实使用	记录使用中的异常、文档缺口	能否复现，是否已有反馈
目标仓Issue	筛选开放问题、近期讨论和小范围修改	是否仍有效、是否有人处理
SIG讨论	查看公开议题、会议与阶段需求	有无维护者对接与验收条件
源动力实习任务	浏览当前任务及申请规则	资格、时间、导师审核与提交要求
课程或高校活动	领取教师与社区共同确认的任务	是否对应真实仓库和社区需求
版本共测	按当期公告选择场景与设备	镜像、用例、报告提交入口

6. 高质量Issue应包含什么

【标题描述】能够简要描述问题：说明什么场景下，做了什么操作，出现什么问题（尽量使用正向表达方式）
【环境信息】
硬件信息：
1）裸机场景提供出问题的硬件信息；
2）虚机场景提供虚机XML文件或者配置信息
软件信息：
1）OS版本及分支
2）内核信息
3）发现问题的组件版本信息
如果有特殊组网，请提供网络拓扑图
【问题复现步骤】
具体操作步骤
出现概率(是否必现，概率性错误)
【预期结果】
描述预期结果，可以通过对比新老版本获取
【实际结果】
描述出问题的结果
【附件信息】
比如系统message日志/组件日志、dump信息、图片等

一个好Issue的标准

别人能够按照你的信息复现；问题边界清楚；不包含隐私和敏感信息；不是把大段日志直接丢给维护者；说明你已经做过哪些排查。

7. 与维护者确认任务的沟通模板

你好，我是来自【学校/专业/年级】的学生，目前在学习【技术方向】。

我阅读了这个Issue和相关代码/文档，我对问题的理解是：【一句话说明】。
我计划先做：【步骤1、步骤2、步骤3】。
预计在【时间】内提交初版。

想请确认：
1. 这个问题目前是否仍需要处理；
2. 我的处理方向是否合适；
3. 是否有必须遵循的测试或提交要求。

谢谢，我会在有阶段性结果后继续在本Issue更新。

8. PR提交检查表

□	PR只解决一个主要问题，未混入格式化、依赖升级等无关修改。
□	已经关联对应Issue，并在描述中写清背景、修改内容和影响范围。
□	所有Commit信息清晰，符合Commit提交规范。
□	已经按照目标仓要求完成编译、静态检查、测试或场景验证。
□	已经附上测试结果、截图、日志或其他必要证据。
□	涉及行为或接口变化时，已经同步更新文档、Sample或测试。
□	没有提交密码、Token、个人隐私、机密数据、临时文件和大体积无关文件。
□	已经自查编码规范、README规范、许可证和第三方依赖要求。
□	愿意持续响应Review，不会在提交后失联。

9. 如何回应Review

场景	推荐做法	不推荐做法
评审要求补充测试	说明已补充哪些测试并附结果	只回复“已修改”但不给证据
不理解意见	引用具体评论，复述理解并提问	忽略评论或私下争论
不同意方案	给出技术依据、数据或替代方案	以个人偏好反驳
发现任务范围扩大	在Issue确认是否拆分新任务	不断把无关修改塞入同一PR
短期无法继续	公开说明进度和预计恢复时间	长期不回应

10. 使用openKylin贡献助手工具快速入门

openKylin贡献助手（命令名 okcontrib）是社区维护的贡献辅助工具，支持查看社区SIG及其维护仓库、推荐适合新手的Issue、引导完成新手首次完成“CLA签署→Gitee Token令牌配置→选SIG→找Issue→提PR”全流程、生成分支与PR草稿、按仓库汇总个人贡献等功能。帮助大家快速了解和掌握参与openKylin社区贡献的基本知识和流程。

已安装openKylin系统（支持2.0、3.0系列版本）的用户，可以直接执行如下命令从软件源安装：

```
sudo apt update
sudo apt install openkylin-contrib-helper
```

装好后执行 okcontrib --version 输出版本号，再跑 okcontrib onboard，按提示依次完成签署CLA、配置Token、选SIG、挑Issue，进度可中断续跑。

工具详细介绍及命令清单与常见问题等内容详见项目README：openKylin贡献助手。

11. 第一次贡献如何认定成果

状态	可以准确写入学习记录的内容
已实践	跑通程序、完成测试、形成报告
已提交	提交了Issue、PR或翻译等贡献，有可访问链接
已参与评审	回应反馈、修订材料或代码，有讨论记录
已采纳或合入	问题被确认、用例被采用、文档发布或PR合入
已发布或持续维护	修改进入发布版本，或围绕同一项目持续协作

PR被关闭、问题未复现、任务尚待审核时也有学习价值，但应如实标注状态。提交后回应反馈、补充验证并记录结果，是首次协作的重要部分。

PART 04

第四部分

openKylin高校开发者成长路径

把一次实践变成持续贡献、能力履历和社区角色。

1. openKylin 成长阶段与社区角色的关系

openKylin 社区通过积分体系来记录和激励开发者的成长。随着积分的累积，开发者将经历从“OK见习”到“OK大咖”的进阶，并逐步解锁对应的社区权限与治理角色。

成长阶段与社区角色（1/2）

成长阶段	对应社区积分	阶段目标
OK见习	1 - 1000	了解社区规则，完成环境搭建与首个微小任务
OK新锐	1001 - 3000	跑通示例代码，完成首次公开协作
OK能手	3001 - 8000	围绕特定 SIG 方向持续产出，参与社区互动
OK精英	8001 - 20000	主导复杂任务，参与代码评审与新人带教
OK大咖	20000+	承担模块维护、技术方案决策与社区治理

成长阶段与社区角色（2/2）

成长阶段	可验证成果	对应社区状态 / 治理角色
OK见习	修复简单的文档错漏、完成多语言翻译、在社区提交规范的 Issue	用户 (User) / 潜在参与者

成长阶段	可验证成果	对应社区状态 / 治理角色
OK新锐	完成源动力初级任务、提交首个被合入的代码 PR、参与测试并复现 Bug	项目贡献者 (Contributor)
OK能手	多次有效代码合入、参与 SIG 例会讨论、在邮件列表或论坛解答他人疑问	活跃 Contributor
OK精英	独立承接中型模块任务、参与 Review 代码、撰写高质量技术文档或教程	Maintainer 候选人
OK大咖	主导版本特性开发、核心系统重构、社区技术布道	Maintainer / SIG Owner / 社区委员会成员

两套体系不要混淆：积分体系（见习-大咖）是面向贡献者的激励与能力分层，通过完成任务累积；而 社区治理角色（User-Contributor-Maintainer-Owner）是 SIG 组内的正式权限身份，需要按官方治理规则提名与审核。积分可以证明你的活跃度，是申请治理角色的重要依据。

2. 每一积分阶段做什么

每一积分阶段做什么（1/2）

成长阶段	建议周期	重点能力
OK见习	1—2 周	社区认知、账号/CLA 签署、基础 Git
OK新锐	1—2 个月	开发工具使用、Issue 规范、PR 流程

成长阶段	建议周期	重点能力
OK能手	2—6 个月	SIG 模块理解、自测试、连续任务交付
OK精英	6—12 个月	独立方案设计、Review 能力、团队协作
OK大咖	长期	架构设计、社区治理、质量把控

每一积分阶段做什么（2/2）

成长阶段	毕业成果（积分获取点）
OK见习	完成注册并绑定账号；跑通一个 UKUI 组件或 openSDK 示例；完成 1-2 个基础任务（如文档纠错、翻译），积分达到 1-1000
OK新锐	完成源动力初级任务（如简单的界面修改、外设测试）；提交并合入 1 个 PR；积分达到 1001-3000
OK能手	围绕同一 SIG（如 UKUI、Kernel）完成多次有效贡献；参加 2 次以上 SIG 例会；积分达到 3001-8000
OK精英	作为核心开发完成中型模块任务；参与社区 PR 评审；带教一名“OK见习”；积分达到 8001-20000
OK大咖	获得仓库或 SIG 信任，申请成为正式 Maintainer；主导技术特性落地；积分突破 20000+

3. openKylin 社区角色与晋升

openKylin 社区治理角色围绕 SIG 展开。积分体系与治理角色相辅相成，高积分是申请晋升的必要条件。

角色	核心职责/特征	阶段建议
User	使用系统并通过 Issue 反馈问题或建议	从真实使用、复现问题反馈开始（OK见习阶段）
Contributor	SIG 内持续活跃的参与者，响应分配任务，做出相关贡献；可以超过 2/3 票数参与 SIG 重大决策	完成首次贡献并持续积累（OK新锐至OK能手阶段）
Maintainer	审核 SIG 代码质量与正确性，评审 PR，更新维护软件包版本，跟踪安全问题	建立持续贡献记录与 Review 能力后申请（OK精英阶段）
Owner	SIG 核心成员/组长，负责技术路线规划、发布计划、内外资源协调，代表 SIG 参与社区技术委员会	属于长期核心治理角色（OK大咖阶段）

晋升路径：贡献频次较高的 Contributor 可以申请成为项目审核者 Maintainer。openKylin 的晋升以 SIG 内持续贡献和评审信任为核心依据，具体标准以各 SIG 的治理规则和社区最新文件为准。

4. 不同年级的推荐路径

对象	主目标	推荐任务	建议成果
大一 / 大二	建立兴趣和基础协作能力	源动力 初级任务、UKUI 组件示例、文档修订、翻译	可运行 Demo + 第一次公开 Issue/PR 反馈（冲刺 OK见习 / OK新锐）
大三 / 大四	进入真实工程和贡献链路	中级源动力任务、Good First Issue、安全补丁、应用适配	至少一个有效 PR；课程设计或作品（冲刺 OK能手）
研究生	结合课题形成深度贡献	子系统性能、安全、RISC-V 适配、AI 子系统、工具链与长期维护任务	模块成果、论文/毕设关联、持续贡献记录（冲刺 OK精英 / OK大咖）
社团骨干	组织共学和新手转化	源动力任务包、Office Hour、贡献冲刺、经验沉淀	稳定学习小组、贡献案例和助教能力

5. 把 openKylin 贡献转化为学业成果

高校场景	适合任务	成果建议
平时作业	文档、Issue、Sample 小改动、openSDK API 实验	链接 + 实验报告 + 复现/验证记录
期末大作业	完整应用、UKUI 插件、数据/网络/多媒体功能组合	代码仓、Demo、测试和展示答辩
课程设计	子系统实验、测试工具、驱动或应用模块	设计文档、实现、测试与社区反馈
毕业设计	真实 Issue、模块开发、性能优化、RISC-V/ARM 适配	开题对齐、社区导师、阶段 PR 与结题成果
研究生课题	内核、安全、AI 子系统、编译工具链、系统性能	研究问题 + 开源原型 + 数据 + 社区验证
竞赛 / 训练营	源动力高等级任务、校源行/高校站限定任务	合入结果、贡献排名和复盘

6. 建立个人 openKylin 贡献履历

□	稳定的开发者账号与邮箱，源动力任务平台账号与 Gitee/GitCode 账号保持一致。
□	目标方向、SIG 与仓库清单（如 UKUI SIG、Kernel SIG、RISC-V SIG、QA SIG）。
□	学习成果：UKUI 组件示例、openSDK 实验、源动力任务记录、开发板实验记录。
□	公开贡献：Issue、PR、Commit、Review 和合入链接。
□	积分记录：截图或记录自己从“OK见习”到当前阶段的积分增长曲线。
□	技术说明：背景、方案、测试、结果和个人职责。
□	社区协作：SIG 例会、邮件列表答疑、文档贡献、新人辅导。
□	阶段复盘：学到了什么、下一步做什么、需要谁帮助。

履历条目模板

内容	示例填写
时间 / 项目	2026年10月 | openKylin UKUI SIG / 源动力任务
问题与目标	复现并修复某 UKUI 组件在特定场景下的异常；完善相关测试
个人行动	定位代码、提交 Issue、实现修复、补测试、回应 Review
公开证据	Issue 链接、PR 链接、合入 Commit、测试截图、源动力任务审核记录
能力标签	C++/QML、调试、单元测试、Git 协作、SIG 沟通
阶段积分	本次任务获得 500 积分，当前总积分 2500，处于 OK新锐阶段
结果与反思	问题已合入；下一步准备承担同仓库第二个任务，向 OK能手 冲刺

7. 90 天成长计划

周期	重点任务	建议产出
第 1—2 周	选方向、跑通示例、熟悉 SIG 和任务平台	1 个运行成果 + 学习记录；完成 CLA 签署（突破 0 积分，成为 OK见习）
第 3—4 周	复现问题或完成文档 / Sample 任务	1 个高质量 Issue 或 PR；或完成 1 个源动力低等级任务（积分达到 1000+，晋升 OK新锐）
第 2 个月	围绕同一仓库/ SIG 完成第二、第三个任务	至少 1 个合入成果 + Review 经验（积分向 3000 迈进）
第 3 个月	参加 SIG 会议、帮助新人、尝试 Review 或中等任务	持续贡献记录 + 下一季度计划（目标稳固在 OK能手 阶段）

附录A：7天入门计划

天数	任务	完成标准
Day 1	认识项目与技术方向	写出自己想做什么、现有基础、每周可投入时间
Day 2	选择SIG和仓库，准备账号	收藏社区官网入口，读完一个仓库README
Day 3	安装或登录实验系统	能使用终端与网络，形成环境记录
Day 4	跑通最小程序或完成一组测试	保存代码、步骤和实际结果
Day 5	做一个小修改或复现已有问题	能解释改动，提供前后对照
Day 6	阅读规则并完成贡献准备	核对CLA、目标分支、模板和验证要求
Day 7	提交真实反馈，或确认首个任务	获得公开链接，明确下一步行动

每天的实际用时按个人情况调整。若没有真实缺陷，第7天可以选择任务沟通，不必硬凑Issue。

每天的实际任务示例

天数	可直观看到的行动	当天留下什么
Day 1	选择“文档与测试”，写每周投入4小时	一段目标与范围
Day 2	从社区官网入口进入docs或QA资料，读一个README	目标仓与规则链接
Day 3	安装匹配镜像，确认更新状态、内核和会话	环境记录与干净快照
Day 4	保存并重开“实验记录.txt”，再测空文件和中文	3条用例及真实结果
Day 5	按原教程复验步骤，记录一处确实存在的缺口	原文位置、证据和建议
Day 6	确认贡献流程、CLA状态和目标分支，准备说明	本次修改与验证清单
Day 7	存在真实缺口则提交；没有则确认一个后续任务	真实链接或明确下一步

附录B：六张入门任务卡

以下均为教学设计，正式认领前需确认对应仓库、任务和维护者。

任务卡1 | 修订一处文档

字段	内容
适合对象	零基础、技术写作、刚完成教程的学生
建议周期	2小时—3天
输入	一篇亲自执行过的安装、使用或开发教程
步骤	复现原步骤→记录缺口→检索重复反馈→修改或提Issue→回应审核
交付	原文位置、适用版本、问题证据、改文与验证记录
验收	新同学可照步骤完成；链接、图片和目标分支符合仓库规则
进阶	编写一个FAQ或完整实验教程

任务卡2 | 完成一次缺陷复现与回归

字段	内容
适合对象	测试方向、初学者、系统使用者
建议周期	2天—2周
输入	一个可观察问题或已有Issue
步骤	固定环境→最小复现→记录频率→提交证据→按修复版本回归
交付	用例、日志、Issue链接和前后对照
验收	他人可复现；区分失败、通过与未测；无法复现时如实记录
进阶	自动化该用例，或帮助定位模块

任务卡3 | 小应用与真实项目改进

字段	内容
适合对象	有基础编程能力的学生
建议周期	1—2周
输入	教学小应用，或目标仓维护的最小示例
步骤	跑通原版→增加一个小功能→测试边界→完善README
交付	代码、环境、运行截图、测试与说明
验收	可运行、改动可解释；个人作品与社区采纳状态分开标注
进阶	与维护者确认真实缺陷，提交修复PR

任务卡4 | 软件包构建与验证

字段	内容
适合对象	熟悉Linux和依赖管理的学生
建议周期	1—3周
输入	一个维护者推荐的、依赖较少的软件包
步骤	查分支与打包规则→构建原版→检查产物→安装运行→卸载→记录
交付	源码版本、构建日志、包信息、安装运行与卸载报告
验收	在约定环境可复现；依赖和许可信息完整；不混用软件源
进阶	依赖修复、版本更新

任务卡5 | 一次开源布道与分享

字段	内容
适合对象	善于表达、喜欢分享、社团骨干，或有自媒体经验的学生
建议周期	1周
输入	你在阅读本手册前三个部分（安装、跑通示例、提Issue）时踩过的坑和最终解决办法
步骤	梳理学习笔记 → 提炼出3个新手最容易卡壳的点 → 配上截图或录屏 → 整理成一篇“新手避坑指南” → 发布在个人博客或校园论坛，并转发到openKylin交流群
交付	一篇公开发布的文章链接、帖子截图，或在社团内做一次15分钟的线上/线下分享PPT
验收	至少有一名其他同学表示看了你的文章后成功解决了问题
进阶	参与到“源动力实习平台”的任务推广中

任务卡6 | 多架构或系统组件实践

字段	内容
适合对象	高年级、研究生，有导师或对应设备支持
建议周期	2—8周
输入	明确版本的组件、板卡或维护者确认的任务
步骤	基线验证→构建/复现→单点修改→回归→公开交流
交付	设备与环境、代码提交、完整日志、测试范围和结果
验收	能区分环境与代码问题；性能结论有测量条件；影响范围清楚
进阶	持续适配、组件维护或关联毕业设计

附录C：可直接复制的模板

C1 Issue模板

标题：[模块/功能] 在什么条件下出现什么问题

系统与镜像：
架构与设备/虚拟机：
内核、桌面会话及相关包版本：
源码分支与提交号（如涉及）：

前置条件：
复现步骤：
1.
2.
3.

预期结果及依据：
实际结果：
复现频率：例如5次中3次
关键日志、截图或最小样例：
已经尝试的排查：
相关Issue及是否重复：
是否愿意协助修复或回归：

填写样例（教学示例，方括号必须换成实测信息）：

标题：[终端/粘贴] 在[实际包版本]下右键粘贴无内容
系统：openKylin 3.0 Desktop；镜像：[完整名称]；更新日期：[实际日期]
设备与架构：[实际型号 / 虚拟机配置]；内核与会话：[命令输出]
组件包版本：[dpkg-query实际输出]
前置：剪贴板中为普通文本campus-test，终端处于空白输入位置
步骤：复制文本→右键粘贴→观察→清空输入→用菜单粘贴对照
预期：输入位置出现相同文本（依据组件支持的粘贴功能）
实际与频率：[本人填写，不能把预期复制为实测]
附件：[截图、关键日志；仅提供脱敏内容]
已有问题：先查IK7R3Y及目标仓状态，再决定补充或新建
回归：愿意按维护者指定版本重测

C2 任务沟通模板

你好，我正在学习【方向】，希望参与这个任务。
我已经阅读【Issue/文档/代码】，目前理解是【问题与影响】。
计划先完成【步骤1、步骤2】，预计【日期】提交初版。
我具备【相关基础】，目前使用【系统、架构与分支】。

想请确认任务是否仍有效、是否已有人处理，以及建议的目标分支、
测试要求和验收范围。我会在本讨论中更新进度。

C3 PR说明模板

关联Issue：
背景与问题：
本次修改：
影响范围：

验证环境：系统/架构/分支/提交号/工具版本
验证方法与实际结果：
修改前后对照：
未测试部分及原因：
兼容性或已知限制：
文档、测试、许可证同步情况：
AI辅助说明（如适用）：

C4 学生成果记录模板

姓名或开发者名称：
学校/课程/学习小组（按需）：
时间与投入：
技术方向、SIG、目标仓：
问题与目标：
我独立完成的工作：
他人提供的帮助：
环境与复现步骤：
代码、报告及公开链接：
当前状态：已实践/已提交/评审中/已采纳/已合入/其他
测试结果与限制：
下一步计划：

附录D：常见问题FAQ

Q1：没有Linux基础可以参与吗？

可以。先完成虚拟机安装和基础操作，再从教程复验、测试或文档开始。应用和系统开发根据个人基础逐步进入。

Q2：没有国产电脑或开发板可以参与吗？

可以。匹配相应架构的PC或学校实验机即可开展多类学习和贡献。特定硬件相关任务才需要对应设备。

Q3：一定要先加入SIG才能提Issue或PR吗？

按目标仓贡献规则即可开始公开参与；正式加入SIG和申请维护权限是进一步的流程，取得维护权限未作为新人的前置条件。

Q4：优先学openKylin 3.0还是2.0 SP2？

新环境可以优先使用3.0；课程、设备和目标仓已有统一版本时跟随其要求。版本、软件源、依赖和分支必须配套。

Q5：3.0系统、UKUI 4.24和SDK 3.0为什么版本不同？

它们属于不同项目，各自独立发布；以安装包和目标仓确认，不要求版本号一致。

Q6：更新后原来的缺陷无法复现，还能参与吗？

可以提交维护者需要的回归证据，写清当前版本和测试范围；未复现不等于证明所有场景都已修复。

Q7：写文档、做翻译和设计算贡献吗？

这些都可以是贡献方式，但要符合对应项目的规范和审核要求。保留验证与采纳记录，不以“写过”代替“已被采用”。

Q8：第一次PR没有合入是不是失败？

不代表没有收获。复盘需求、方案、质量或时间原因，形成下一步改进。对外展示如实写明已提交、待评审或未采纳。

Q9：怎么找官方实习、证书和奖励？

从官网进入社区实习、成长体系或当期活动页面，按现行规则申请和查询。完成教学任务并不自动获得官方证书、报酬或实习证明。

Q10：可以把贡献作为课程作业或毕业设计吗？

可以与教师及维护者提前讨论。明确个人职责、公开范围、时间和评价标准，保留代码、实验、评审和阶段交付证据。

Q11：可以用AI学习和写代码吗？

按社区、课程和仓库规则使用。对实质性辅助作说明，人工理解与验证，不编造测试、不提交来源或行为无法解释的内容。

Q12：本校还没有高校站，该如何开始？

联系社区官方推进本校加入openKylin高校站，享受更多权益。在此之前，个人可直接使用官方资源和公开仓库；几位同学也可先组成学习小组。

Q13：KARE与deb怎么选？

原生依赖与系统包练习从deb入门，独立应用交付按文档，历史Linux应用兼容按KARE资料；先看项目需求，不把三个方案当成同义词。

Q14：为什么有的教程不能直接用apt安装软件？

可能涉及系统模式、源配置、架构或包名差异。先看完整报错和对应版本文档，不要直接混源或照搬其他发行版的处理方式。

Q15：为什么还需要打包？程序能运行不就够了吗？

打包处理文件安装、依赖、版本和交付。个人目录能运行，只证明一个局部环境可用；软件包还要验证安装、升级或卸载等行为。

附录E：openKylin资源索引

下列入口用于持续查阅。页面迁移时优先从官网导航和对应官方仓库重新定位，不使用搜索结果中的同名个人Fork替代官方入口。

认识openKylin

资源	用途	网址
openKylin官网	社区介绍、新闻、高校专区和活动入口	https://www.openkylin.top
openKylin文档平台	使用、开发、规范与参与指南	https://docs.openkylin.top
社区行为守则	社区交流和协作要求	https://www.openkylin.top/join/conduct-cn.html
社区论坛	使用交流、经验与问题讨论	https://bbs.openkylin.top
Gitee代码仓	openKylin代码托管平台	https://gitee.com/openkylin
数字看板	openKylin运营情况统计看板	https://datastat.openkylin.top

体验openKylin

资源	用途	网址
ISO镜像下载	下载各架构、版本的openKylin镜像	https://www.openkylin.top/downloads
安装指南	虚拟机安装、实体机安装、分区建议等配套说明	https://docs.openkylin.top

加入openKylin

资源	用途	网址
openKylin ID	社区身份和个人信息	https://id.openkylin.top
CLA签署	贡献者许可协议	https://cla.openkylin.top
SIG列表	社区SIG小组汇集	https://www.openkylin.top/sig/siglist-cn.html
SIG邮件列表	订阅 SIG 动态、参与技术讨论	https://mailweb.openkylin.top
社区会员	组织（企业/高校）加入社区	https://www.openkylin.top/community/partner-cn.html
高校专区	openKylin走进国内140+高校	https://www.openkylin.top/dynamic/higher_education_zone-cn.html

贡献openKylin

资源	用途	网址
代码仓库	代码仓库管理、Issue和PR提交	https://gitee.com/openkylin
community治理仓	SIG、包归属和社区规则	https://gitee.com/openkylin/community
单包维护相关资料	软件包维护信息与流程	https://gitee.com/openkylin/community/tree/master/packages
开发者平台导航	编译、构建、CI、代码质量等openKylin官网入口	https://www.openkylin.top/developers/index-cn.html
软/硬件兼容适配平台	软/硬件兼容适配申请与展示	https://www.openkylin.top/support
软件包编译平台	软件包构建与日志	https://build.openkylin.top
发行版构建平台	发行版镜像构建相关工作	https://factory.openkylin.top
持续集成平台	自动化构建与检查	https://jenkins.openkylin.top
SIG会议	了解 SIG 运作节奏和参与讨论的公开渠道	https://www.openkylin.top/sig/meeting-cn.html
翻译平台	国际化、本地化协作	https://weblate.openkylin.top
源动力实习平台	社区实习任务平台	https://task.openkylin.top/dashboard
成长体系	查询成长等级与激励规则	https://www.openkylin.top/community/pgs-cn.html
人才认证	openKylin人才认证体系	https://www.openkylin.top/community/talent_certification.html

附录F：openKylin 社区国内平台二维码



微信公众号



交流群



成为开发者



微信小程序



B站

附录G：openKylin 社区国际平台二维码

Twitter

Linkedin

Telegram

YouTube