RustSBI原型设计系统（机器态模块）的设计与实现

洛佳

华中科技大学网络空间安全学院

2024.5

本次演讲……

- RustSBI原型设计系统（机器态模块）
- 什么是动态固件（fw_dynamic）？
- 动态信息的定义
- 作为动态固件的RustSBI原型设计系统
- RustSBI原型设计系统的预期生态

RustSBI原型设计系统（机器态模块）

- RustSBI原型设计系统是RustSBI社区新增的子项目
- 本项目的机器态模块运行于RISC-V架构的机器态
  - RISC-V指令集架构具有机器（M）、内核（S）和用户（U）三个特权态
  - 机器态是RISC-V专有的特权态，它的权限大于内核态，可以完成额外的安全功能
- RISC-V机器态运行的软件分为：引导程序前级和机器态环境模块
  - 引导程序前级：与ROM代码沟通，启动环境模块后，生命周期结束
  - 机器态环境模块：启动内核，并常驻在内核后台，持续提供内核运行需要的功能
- RustSBI原型设计系统的机器态模块属于机器态环境模块
- 机器态环境模块的设计要点：安全性、高性能
  - 模块的权限大于内核，位于或接近安全系统的信任根
  - RISC-V架构下，机器态模块的性能会影响内核本身的运行效率

什么是动态固件（fw_dynamic）？

- 产业界的RISC-V机器态模块实现方法之一
  - 优点：具有可移植性，与C语言生态软件相容性好
- 在SBI标准上做增补
  - opaque参数（a1寄存器）定义为Flattened Device Tree（FDT）格式设备树的基地址
  - 定义非标准的a2寄存器，用于保存新定义Dynamic Information的基地址
- 动态信息（Dynamic Information）
  - 既包含下一引导阶段（如内核、后续引导程序等）的基地址、特权级等，也包含本阶段需要的引导信息（如最佳的启动核编号）
  - 动态信息的定义没有规范标准，不包含在RISC-V SBI国际标准中，但是社区默契地达成了事实标准
- 动态固件加载的基地址不固定，并且需要从前级引导程序中获得设备列表
  - 因此，动态固件如果具有多个不同主板的设备支持驱动，就能支持更多样的主板和平台

动态信息的定义

```rust
/// M-mode firmware dynamic information.
#[derive(Clone, Copy)]
#[repr(C)]
2 implementations
pub struct DynamicInfo {
    /// Dynamic information magic value.
    pub magic: usize,
    /// Version of dynamic information.
    pub version: usize,
    /// Address of the next boot-loading stage.
    pub next_addr: usize,
    /// RISC-V privilege mode of the next boot-loading stage.
    pub next_mode: usize,
    /// M-mode firmware options; its definition varies between SBI implementations.
    pub options: usize,
    /// Boot hart ID of current environment.
    pub boot_hart: usize,
}
```

作为动态固件的RustSBI原型设计系统

- 依赖于RustSBI 0.4.0版本的#[rustsbi(dynamic)]特性
  - RustSBI 0.4.0正式版将于近期发布，敬请期待
  - 欢迎给RustSBI点上星标！链接：https://github.com/rustsbi/rustsbi
- 适配C语言生态
  - 可直接用于U-Boot、EDK II等C语言引导程序中，使用户获得性能和安全性提升
  - 减少C语言生态厂家的适配难度
- 目前需要做的事情
  - 完善RustSBI原型设计系统的框架
  - 支持更多的设备，在更多的平台上做实际的兼容性测试
  - 使用多种技术路线启动操作系统（裸SBI、EDK II和U-Boot等软件）

我们与RROS团队展开合作

RustSBI团队已经与RROS团队正式开展合作。

期待看到RustSBI软件的宇航级引用，也期待我们能为RROS的星、地应用生态标准化注入更多的动力！

在供电方面，天仪33卫星首次采用了基于天仪自主研发的高功耗长时段供电技术，**使卫星能够支持大电流载荷常态化开机**，满足了载荷不间断在轨试验的需求。

天仪33卫星上还搭载了**北京邮电大学**以**开源**的实时双内核操作系统**RROS**为底层架构，**基于Rust语言**自主研发的科研载荷。该载荷将在星上完成以tensorflow/k8s为代表的通用任务和以实时文件系统、实时网络传输为代表的实时任务，并保障上层应用和科研任务的正常执行，如星地间时延测量、视频直播、星载web聊天服务、伪ssh实验等，**这标志着全球首款由Rust编写的双内核操作系统在卫星场景正式应用。**

RROS在兼顾通用性和实时性的同时，做到了高安全和低功耗，较好地适配卫星场景并支撑卫星载荷的复杂应用需求，为下一代标准化和通用化卫星平台提供操作系统保障。

视频与新闻链接：https://mp.weixin.qq.com/s/4CukKyJe0OUi04Y4DWiUOQ

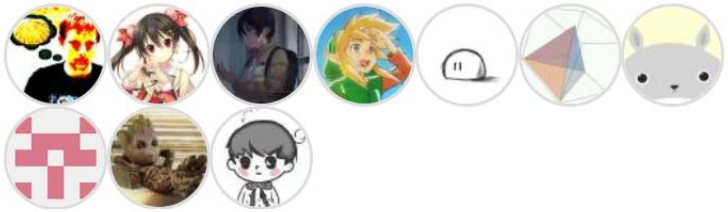
RustSBI原型设计系统的预期生态

- RROS双内核实时操作系统（https://github.com/BUPT-OS/RROS）
  - 是一个人造卫星上有实际应用的操作系统，首个应用已发射升空约半年
  - RustSBI原型设计系统将首先支持具有RROS支持的RISC-V平台
- 结合U-Boot引导程序的生态
  - 我们将上游的“OpenSBI”替换为“SBI”，强调SBI固件实现的中立性和多样性
  - U-Boot支持的老款主板将免费获得RustSBI生态的支持
- OpenHarmony、OpenEuler等操作系统发行版
  - 与对应社区沟通中，期望能和RustSBI一起发行
- 新型SBI应用和研究项目的验证平台
  - DramForever的软件模拟虚拟化、Penglai接口、CoVE-SBI接口和Raven调试器等
  - 利用原型设计系统，减少代码量，降低开发负担

感谢各位！

- 感谢华中科技大学开放原子开源社团的鼎力支持，指导老师：周威老师、慕冬亮老师和王杰老师
- 感谢开源工坊的同学们，包含：董庆同学、王振辰同学、朱俊星同学和历次参与活动的同学们
- 感谢社区成员积极的贡献和审核工作，尤其是@duskmoon314和@YdrMaster。感谢社区成员的先期支持工作，尤其是@OrangeCMS在Oreboot上的软件贡献

People



- 项目地址（欢迎star！）：https://github.com/rustsbi/prototyper
- 我的联系方式：luojia@hust.edu.cn，GitHub & Gitee：@luojia65