hypercraft, a VMM crate written in Rust.

Rust OS 训练营

齐呈祥

2023 年 11 月 6 日

Table of contents

Introduction

Introduction

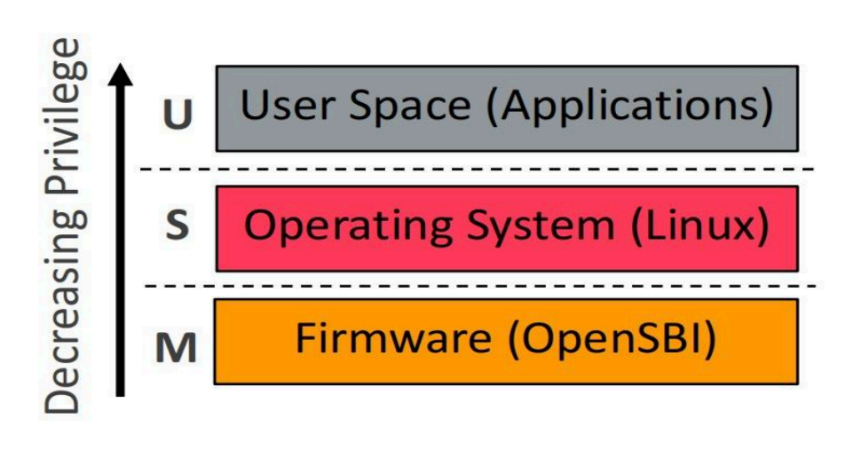
hypercraft 是一个使用 Rust 语言编写的 VMM(Virtual Machine Monitor) crate。目前 hypercraft 已经可以在 arceos 上面作为 type-2 hypervisor 运行并可以启动主线 linux。在 hypercraft 开发前我已经开发了两版 RISC-V hypervisor，分别是 hypocasut 和 hypocaust-2，hypercraft 基于之前的经验进行开发，并参考了 RVM-Tutorial 和 salus。

目前已开源：https://github.com/KuangjuX/hypercraft

RISC-V: the ISA

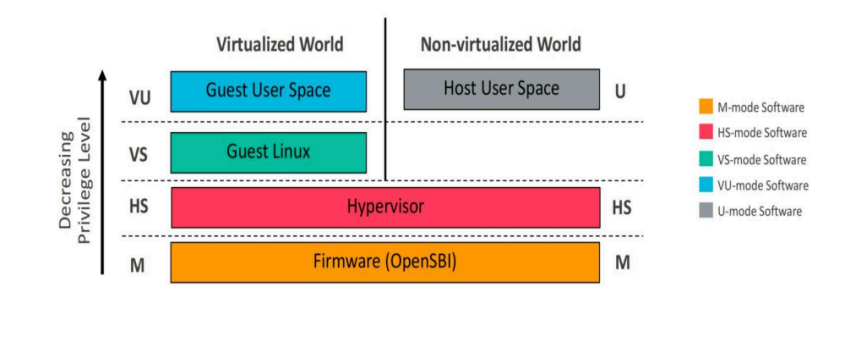
RISC-V: the ISA

RISC-V 传统特权级



- M(Machine), S(Supervisor), U(User) 三个特权级。
- 每个特权级下都有 32 个控制状态寄存器（CSRs）。
- 有 32 个通用寄存器。

RISC-V H 扩展



- HS: S mode with hypervisor capabilities and new CSRs。
- Virtualized S-mode，用于代替传统 S-mode。
- VU-mode: Virtualized U-mode，用于代替原有的 U-mode。

RISC-V: the ISA

HS-mode CSRs for hypervisor capabilities	
hstatus	Hypervisor Status
hideleg	Hypervisor Interrupt Delegate
hedeleg	Hypervisor Trap/Exception Delegate
htimedelta	Hypervisor Guest Time Delta
hgatp	Hypervisor Guest Address Translation

HS-mode CSRs

HS-mode CSRs for accessing Guest/VM state	
vsstatus	Guest/VM Status
vsie	Guest/VM Interrupt Enable
vsip	Guest/VM Interrupt Pending
vstvec	Guest/VM Trap Handler Base
vsepc	Guest/VM Trap Progam Counter
vscause	Guest/VM Trap Cause
vstval	Guest/VM Trap Value
vsatp	Guest/VM Address Translation
vsscratch	Guest/VM Scratch

VS-mode CSRs

RISC-V G Stage

- vsatp: Virtual Supervisor Address Translation and Protection Register，**用于第一阶段页表翻译**。
- hgatp: Hypervisor Guest Address Translation and Protection Register，**用于第二阶段页表翻译**。
- GVA –> GPA(vsatp) –> HPA(hgatp)

RISC-V Interrupt Virtualization Technology

- PLIC
  - **不支持** MSI
  - **不支持中断投递**
  - **需要在** hypervisor **模拟** PLIC **并进行中断注入**
- AIA(Advanced Interrupt Architecture)
  - IMSIC(Incoming Message Signaled Comtroller): **支持** MSI（**消息信号中断**），**支持** IPI Virtualization
  - VS-mode **下运行的客户操作系统对设备中断（作为** MSI）**的直接控制，减少了虚拟机监视器（**hypervisor）**的干预**
  - APLIC：**可以更高效地处理中断**

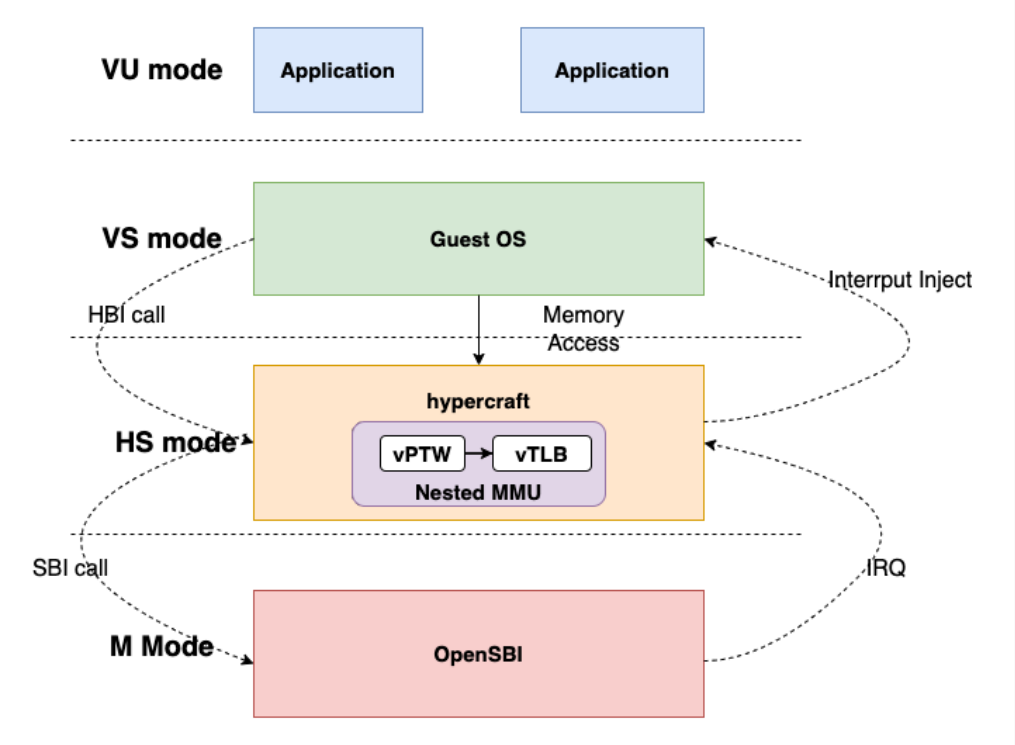
RISC-V I/O Virtualization Technology

- Emulate
  - **纯软件模拟，甚至可以模拟不存在的设备**
  - **平台稳定，不需要特殊的硬件支持**
  - **性能低**
- Passthrough
  - VM **独占** Guest
  - **性能高，实现简单**
  - **需要大量设备（假设有** 100 **个** VM）

hypercraft Overview

hypercraft Overview

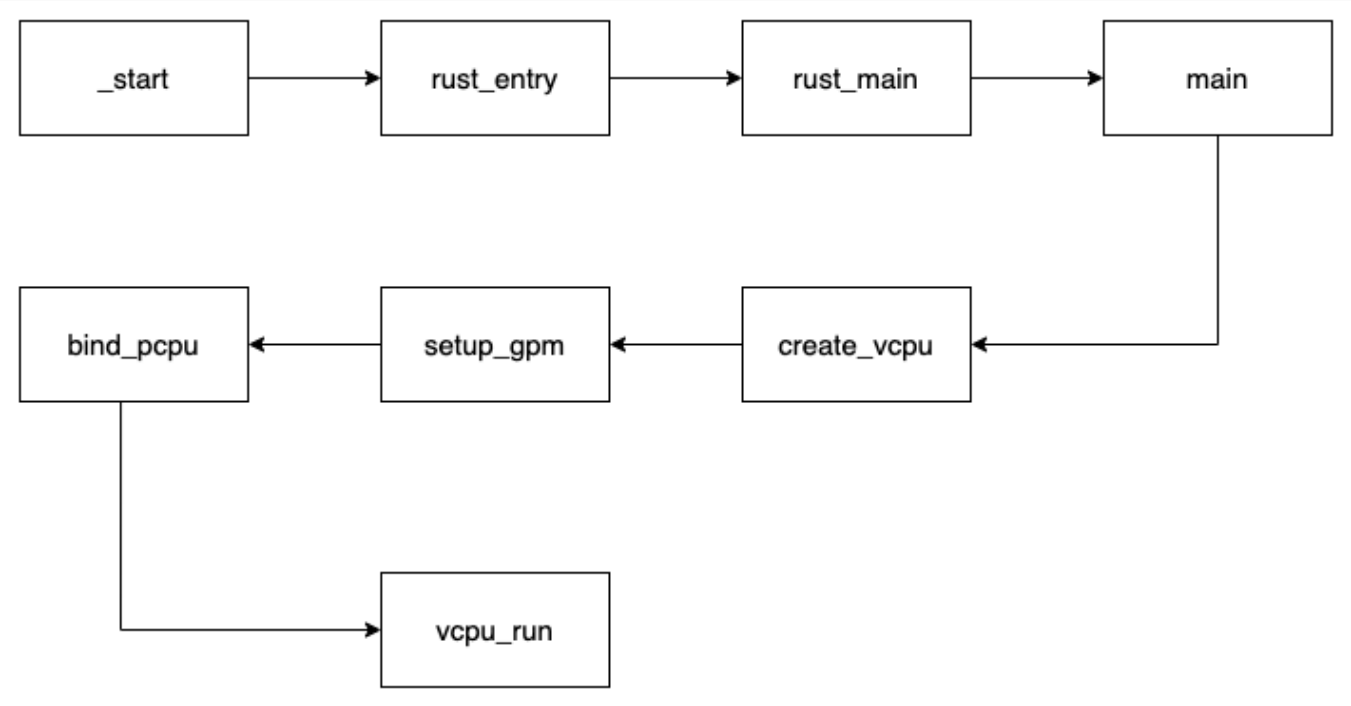
hypercraft 架构



- 基于设备树的配置
- RISC-V H Extension 辅助虚拟化
- 两阶段页表翻译
- 异常代理与中断转发 (PLIC 模拟)
- 设备透传
- 可运行 rCore-Tutorial-v3, RT-Thread 以及 Linux

hypercraft Boot Flow

arceos + hypercraft **启动流程**



hypercraft CPU 虚拟化

- 硬件实现了 H extension，CPU 自动为 S-mode 维护 vs-<xyz> 寄存器，当 guest 写入 s-<xyz> 寄存器时硬件自动将其写入 vs-<xyz> 寄存器。
- 当发生 vmentry/vmexit 的时候需要保存必要的上下文。在 hypercraft 中 vcpu_run 方法中使用一个 loop 用来不断处理 vmentry/vmexit，无需每次都跳到不同的地址，增加可维护性与程序局部性。
- pcpu 与 vcpu 1:1 映射，无需 cpu 进行调度，更为简单高效。为了支持多 guest，未来可能会在每个 vcpu 中加入调度队列。

hypercraft CPU 虚拟化需要保存的最小寄存器

Field	Description	Save/Restore
zero	Zero register	---
ra	Return address register	Trap Entry/Exit
sp	Stack pointer register	Trap Entry/Exit
gp	Global pointer register	Trap Entry/Exit
tp	Thread pointer register	Trap Entry/Exit
a0-a7	Function argument registers	Trap Entry/Exit
t0-t6	Caller saved registers	Trap Entry/Exit
s0-s11	Callee saved register	Trap Entry/Exit
sepc	Program counter	Trap Entry/Exit
sstatus	Shadow SSTATUS CSR	Trap Entry/Exit
hstatus	Shadow HSTATUS CSR	Trap Entry/Exit
sp_exec	Stack pointer for traps	Trap Entry/Exit

hypercraft Memory Virtualization

hypercraft 内存虚拟化

- 基于 arceos page_table crate 扩展了 G-stage，用于建立客户操作系统页表，根据 RISC-V H Extension，客户根页表大小应为 16 KiB 并且需要进行 16 KiB 对齐。
- 目前将第二阶段页表的标识位全部设置为 RWXU，维护起来较为简单，但可能并没有很高的安全性，最终需要修改。
- 在 app 层面类似 KVM 可以由用户自己进行内存映射。

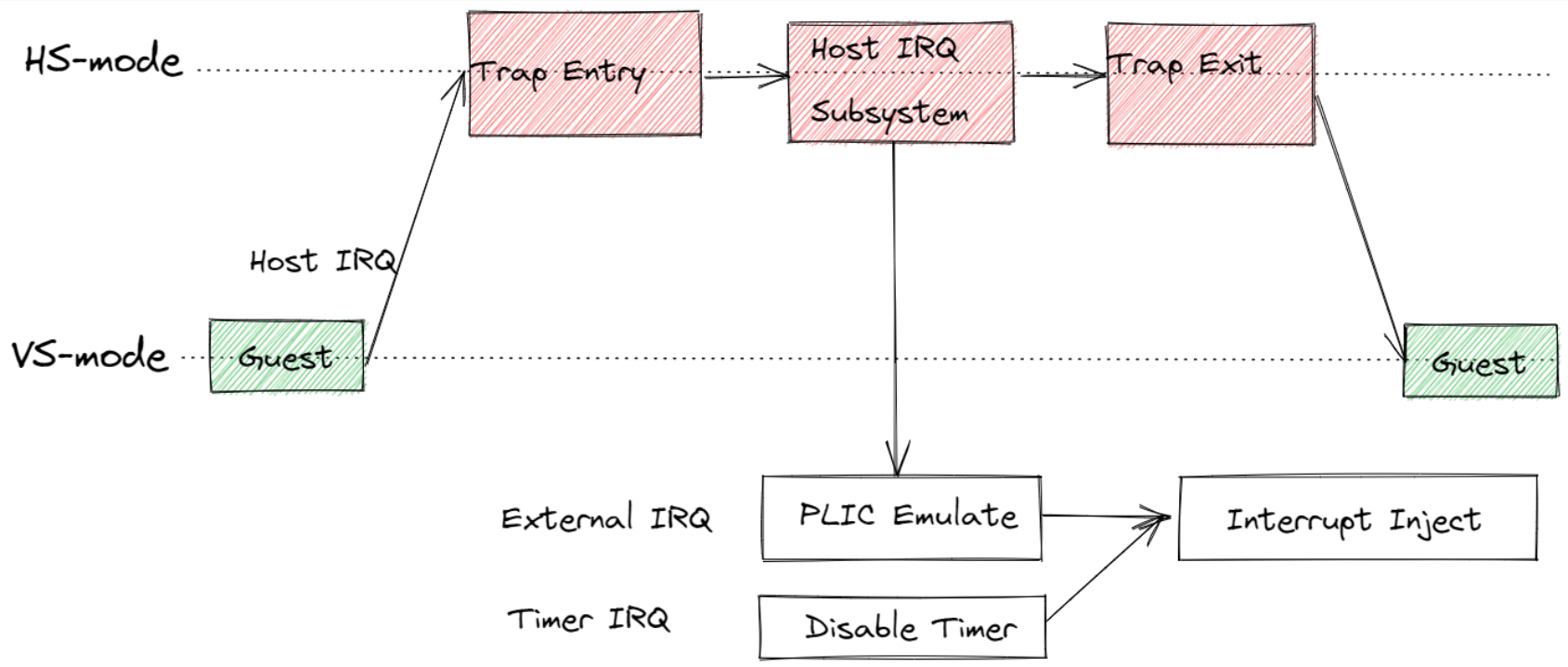
hypercraft Interrupt Virtualization

hypercraft **中断虚拟化**

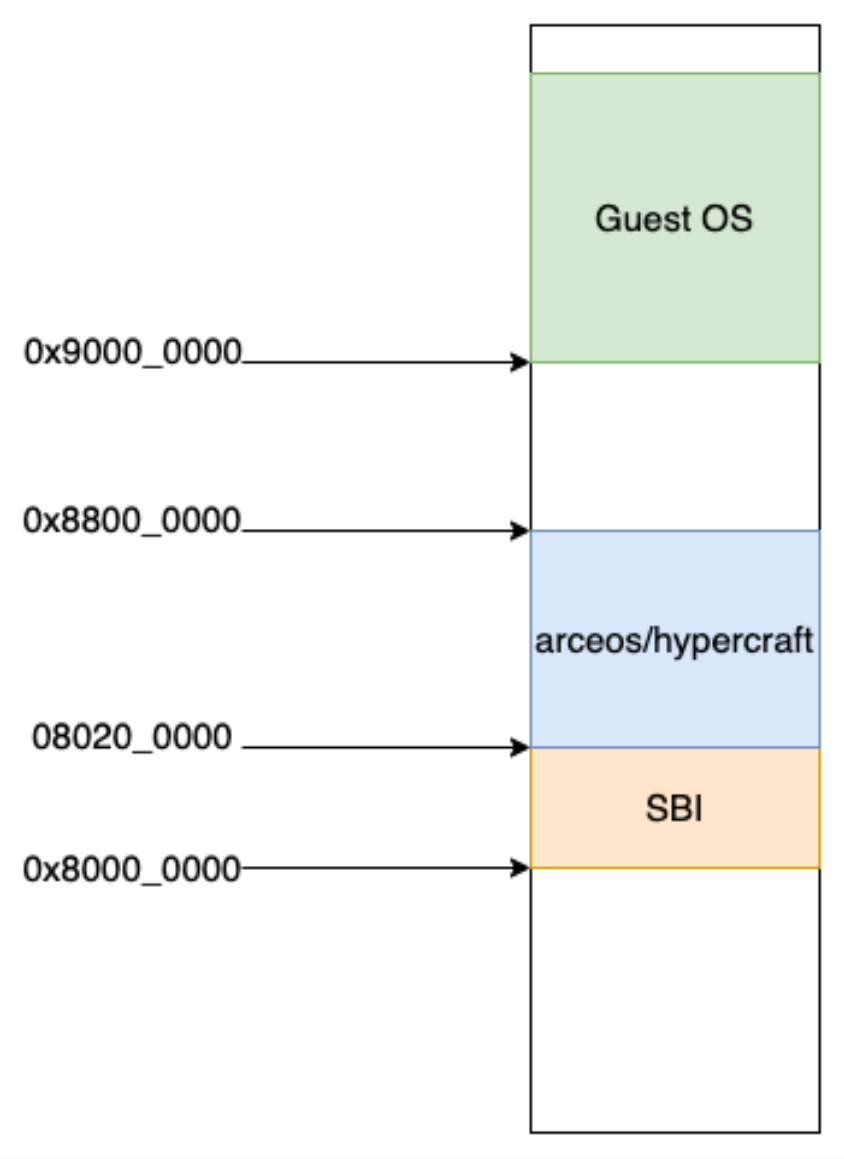
- **当前未实现** AIA，**无法将中断代理到** VS-mode，**需要进行中断注入**。
- **时钟中断**：**接收中断** -> **关闭中断** -> **中断注入** -> **收到** SBI call -> **打开中断**
- **外部中断**：**接收中断** -> **模拟** PLIC -> **中断注入** IRQ ID -> **拦截并模拟写入** PLIC **寄存器操作** -> **告知** PLIC **中断完成**

hypercraft Interrupt Virtualization

hypercraft 中断虚拟化



hypercraft I/O Virtualization



- 当前 RISC-V IOMMU 标准仍然是一个草案，当前 hypercraft 未实现 IOMMU 驱动。
- 基于设备树配置，可使用恒等映射，这样即使不经过 IOMMU 可以直接进行 I/O。

Q&A