Ch16. RV64 OS 移植

16.0 引言

- 大概讲一下要做的事情是自下而上搭建一套软件栈，使得 OS 可以跑起来。

16.1 第一行指令（boot stage 0）： bootloader

16.1.1 bootloader 简介

- bootloader 放在哪里?
- bootloader 有哪些作用?

16.1.2 k210/Ariane bootloader 实例分析

- K210/Ariane 上 bootloader 的实例分析

16.1.3 与 BIOS/UEFI 的比较

- 与广为人知的 bootloader 如 BIOS/UEFI 之间的比较

16.2 准备好内核运行环境 SEE：OpenSBI/RustSBI/BBL

16.2.1 特权级运行环境 SEE 与二进制接口 SBI

- 介绍特权级运行环境与相邻特权级之间的二进制接口（以 SEE/SBI 为例）

  主要介绍相关概念、这样设计的好处

  讲解二进制接口 ABI 和函数调用 API 之间的关系（含 RISC-V 调用规范）

16.2.2 SBI 实现所提供的功能

- 介绍一个合格的 SBI 实现应该实现哪些功能

  1. 寄存器初始化，准备好 S 态运行环境；
  2. 负责 M 态 trap 处理；
  3. 为 S 态代码执行以 ecall 的方式提供服务

16.2.3 开源 SBI 实现代码分析

- 分析开源 SBI 实现（以 OpenSBI 为例）是如何做好上述三点功能的

16.3 实现硬件抽象层 HAL

16.3.1 初始化

- 为不同的 hart 分别设置启动栈，跳转到内核入口点
- 启动核进行内核初始化并阻塞其他核，初始化结束后通过 ipi 提醒其他核完成简单初始化

16.3.2 内存管理模块

- 分析内存管理模块要对上提供哪些接口，如何实现

16.3.3 中断处理

- 分析中断处理模块要提供哪些功能，如何实现
- 如何跳转到中断处理：stvec 设置为 direct 模式还是 vector 模式
- 中断帧内应防止什么内容；中断前后上下文保存/恢复
- 中断嵌套；中断代理（与 SEE 相关）

16.4 驱动程序开发

应该是硬件抽象层的一部分，但是很有代表性所以作为单个章节来讲述。

16.4.1 与设备进行交互：设备寄存器

- 分成状态/数据寄存器，以 MMIO 的形式访问

16.4.2 与设备进行交互：外部中断

- 中断不一定优于轮询
- 介绍 PLIC 的中断分发机制
- 结合 PLIC 叙述一次中断从产生到被处理的详细流程

16.4.3 嵌入式系统常见外设简介

- 串口：通用异步收发器 UART

- 外存设备：如 Flash/sdcard 等；
- 总线设备：如 SPI/APB/USB 等；
- 其他设备如 GPIO/PLL 等。

16.4.4 驱动框架分析

- 分析一种可行的驱动框架（以 rCore 为例）来加深理解。

16.5 应用程序移植

- 讲解用户程序编译构建过程中底层库（如 libc）所起的作用：即将系统调用进行封装提供方便易用的接口

- 大致讲一下如何通过实现所需的系统调用来支持运行基于 libc/musl-libc/tiny-libc 开发的 C 程序

16.6 在 Ariane 开源软核上运行 Linux

16.6.1 环境配置

- 安装串口终端软件 SecureCRT
- 安装工具链

16.6.2 编译安装（参考 ariane-sdk）

- 编译镜像（从 Makefile 详细分析编译流程）
- 准备 sdcard 并进行 gpt 分区
- 将镜像烧写到 sdcard 上

16.6.3 运行 Linux

- 若干命令确认 Linux 的版本或其他元数据
- 玩俄罗斯方块游戏 Tetris
- 通过 busybox 访问文件系统
- 运行其他用户程序

16.7 总结

- 回顾本章知识点并给出总结概要。