Rust China Conf 2020

Shenzhen, China

2020conf.rustcc.cn

DatenLord 高性能用户态存储

自我介绍

回国创业

2014 年回国创业
国内最早推广容器 PaaS 技术的创业公司之一

Google 美国总部

担任美国 Google 广告部门架构师
负责广告大数据分析

教育背景

美国 George Mason 大学博士 2007-2011
北京大学硕士 2004-2007
北京航空航天大学学士 1998-2002

技术与科研成果

拥有云计算领域两项专利、多项软著
发表十余篇论文，累计引用近千次
入选腾讯 TVP

为什么要做用户态存储

高性能

轻量

软件加速

- 自行实现 IO 调度，规避内核性能瓶颈
  - 避免上下文切换成本
    - Linux 每秒钟上万次上下文切换
    - 一次上下文切换时间代价等于一次高速闪存 IO 访问
  - 异步 IO
    - 用户态异步 IO 任务调度
    - 异步系统调用
- 自动化分层存储，绕开内核存储管理
  - 二八原则
    - 80% 的 IO 访问集中在 20% 的数据上
  - 内存不再是紧缺资源，充分利用内存缓存数据

IO 相关操作	延迟	倍数
CPU 执行一条指令	0.4ns	1
内存一次 IO 访问	100ns	250
进程上下文切换	20us	50K
NVMe SSD 一次 IO 访问	25us	62.5K
普通 SSD 一次 IO 访问	150us	375K
机械硬盘寻址	1ms	2.5M

硬件加速

- 片上系统/System on Chip
  - 可编程硬件 / 专用芯片
  - Offloading
    - 纠删码
    - 编解码
    - 加密
    - 压缩
- 非易失内存
  - Intel Optane

对 Rust 进行系统编程的体会

- 优点
  - 强规约
    - 类型安全、内存安全
    - 代码风格规约
  - 函数式
    - 提升代码可读性
    - 提升编译器潜在优化空间
- 缺点
  - 宏过于灵活，可以改变语法
  - 异步场景下生命周期过于复杂
  - FFI 太容易出错
    - 写 Kernel module 还不太友好
  - 错误处理和日志处理 verbose

Rust 能否用于硬件编程

- 硬件编程的特点
  - 连通性
  - 并行性
  - 时间性
    - 时钟 / 脉冲信号上下沿驱动
- 高阶综合
  - LLVM 把 C/C++ 等高级语言编译成硬件描述语言 HDL
- Rust 用于高阶综合
  - 函数接口封装、调用
  - Rust 异步
  - 需要实现
    - 监测时钟 / 脉冲信号上下沿
    - 事件驱动机制

Rust 语言特性对硬件编程的帮助

- 非法地址、整数溢出
  - 硬件不会报错
- 复位初始化 / 赋初值
- 高阶综合优化
  - 人工编译优化
  - 循环展开 / 嵌套循环展开
  - 循环并行化 / 流水线化
  - 函数调用并行化 / 流水线化
- Rust 保证内存访问安全
  - 寄存器地址
- Rust RAII 机制
- Rust 函数式高阶综合
  - 函数式编程对编译优化更友好
  - 自动编译优化？

谢谢

For Extreme Performance,
Think in Hardware.

RUST CHINA CONF 2020

首届中国 Rust 开发者大会

2020.12.26–27 深圳

RUST CHINA CONF 2020

首届中国 Rust 开发者大会

2020.12.26–27 深圳

RUST CHINA CONF 2020

首届中国 Rust 开发者大会

2020.12.26–27 深圳