

2025 春夏 · 开源操作系统训练营汇报

ArceOS / oscamp ——x86-64 端口阶段性总结

莫天昊

2025-06-21

GitHub: <https://github.com/MTttth/oscamp>

1. 序言
2. 工作记录
3. 收获与反思
4. 未来展望

序言

选择 x86-64 的动机

- **Unikernel 思想**——用户态即内核子集，挑战传统 Ring 3/o 隔离
- **宏内核工程学**——调度、内存、驱动大一统，需要在 x86-64 上完整验证
- 本阶段目标：移植 oscamp 到 x86-64+ 补完 x86_rtc 文档/测试

工作记录

第 1 周-学习与规划

- 研读 ArceOS Backbone 设计
- 参照 Risc-V 制定迁移路线：上下文切换 → Trap 机制 → Syscall 通路

上下文切换 & Trap

context.rs

- 保存 RBX RBP R12-R15 CR3 RFLAGS
- context_switch: swapgs+ 双向 iretq
- enter_uspace: 手工构造 iretq 帧
- 在切换用户态前确保 TSS.rspo 指向的分配的内核栈顶

trap.S / trap.rs

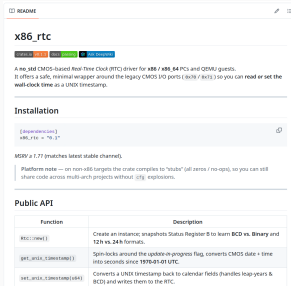
- 256 条 IDT: #PF、#UD、IRQ0、0x80……
- trap.rs 根据向量号派发到 handle_page_fault / handle_irq / handle_syscall。

系统调用通路

- 采用 **SYSCALL/SYSRET** 替代 INT 0x80
- SysV ABI: RAX = nr, RDI RSI RDX R10 R8 R9 传六参
- 在汇编里把寄存器序列化到栈，统一传给 x86_syscall_handler(), Rust 统一处理，退出时 swapgs→sysretq

完善 x86_rtc

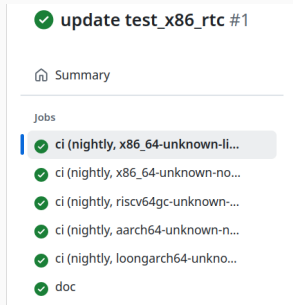
- 重写 README
- 添加了测试



The screenshot shows the README for the `x86_rtc` crate. It includes a description of the crate as a no_std CMSD-based Real-Time Clock (RTC) driver for x86 / x86_64 PCs and QEMU guests. It offers a safe, minimal wrapper around the legacy CMOS I/O ports. The installation section shows the crate name `x86_rtc = "0.3"` and the MSRV `1.77`. The public API section contains a table with four functions: `rtc::new()`, `get_unix_timestamp()`, `spin_locks_around_update_in_progress_flag()`, and `set_unix_timestamp_bcd()`.

Function	Description
<code>rtc::new()</code>	Create an instance; snapshots Status Register B to learn BCD vs. Binary and 12h vs. 24h formats.
<code>get_unix_timestamp()</code>	Spin-locks around the update-in-progress flag, converts CMOS date + time into seconds since 1970-01-01 UTC.
<code>set_unix_timestamp_bcd()</code>	Converts a UNIX timestamp back to calendar fields (handles leap-years & BCD) and writes them to the RTC.

README 文档示意



The screenshot shows the CI pipeline results for the `update test_x86_rtc #1` job. The summary section shows a list of jobs, all of which passed (indicated by green checkmarks). The jobs are: `ci (nightly, x86_64-unknown-li...`, `ci (nightly, x86_64-unknown-no...`, `ci (nightly, riscv64gc-unknown-...`, `ci (nightly, aarch64-unknown-n...`, `ci (nightly, loongarch64-unkno...`, and `doc`.

CI 流水线结果

收获与反思

收获与反思

1. 对 Unikernel 和宏内核有了更多的理解
2. 进一步的掌握了开发 os 内核的一些现代工具
3. 由于时间的问题没能继续深入

未来展望

下一步路线

- **Transparent HugePages**

接入 khugepaged, 降低 TLB miss

- **vDSO 加速**

共享时间页实现 `__vdso_clock_gettime`

谢谢！