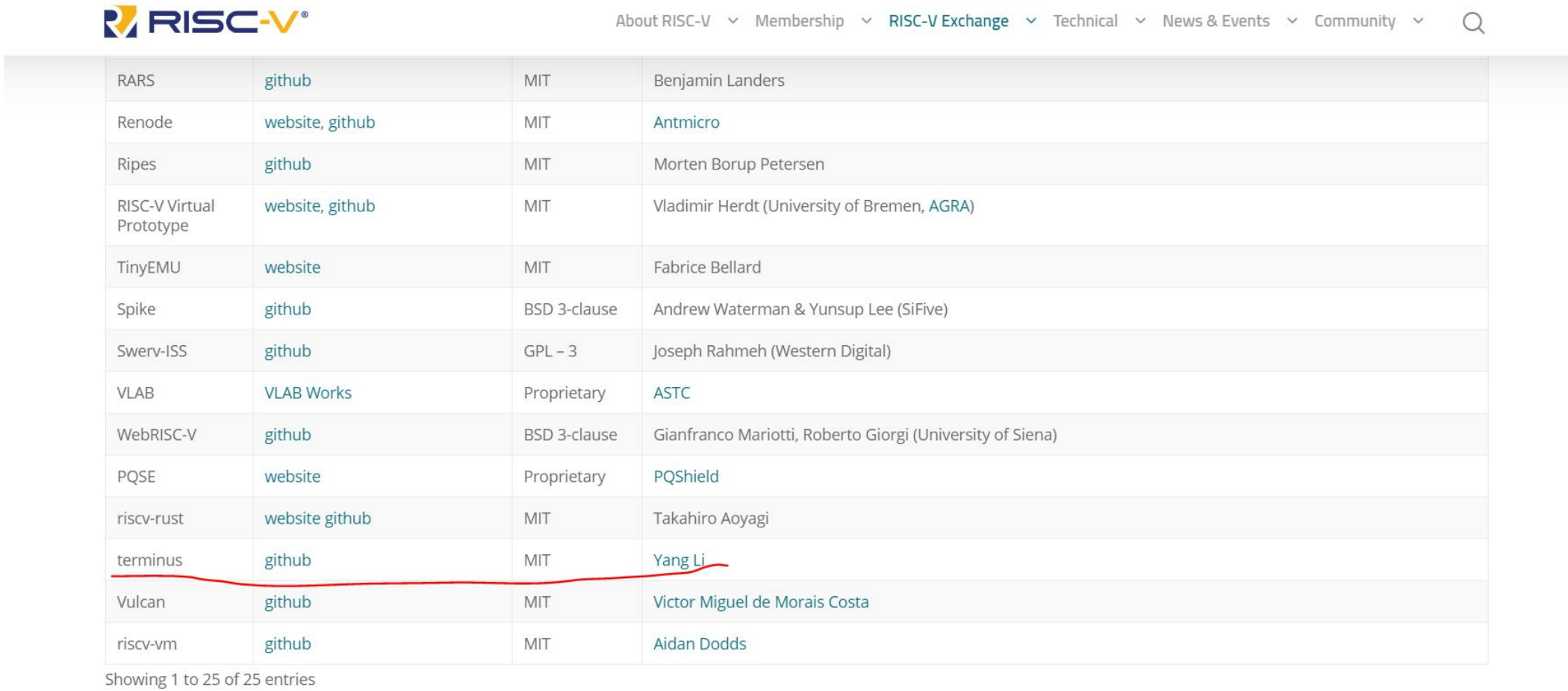
Terminus

A riscv isa simulator in Rust.

github.com/shady831213

Terminus

https://riscv.org/exchange/software



RARS	github	MIT	Benjamin Landers
Renode	website, github	MIT	Antmicro
Ripes	github	MIT	Morten Borup Petersen
RISC-V Virtual Prototype	website, github	MIT	Vladimir Herdt (University of Bremen, AGRA)
TinyEMU	website	MIT	Fabrice Bellard
Spike	github	BSD 3-clause	Andrew Waterman & Yunsup Lee (SiFive)
Swerv-ISS	github	GPL – 3	Joseph Rahmeh (Western Digital)
VLAB	VLAB Works	Proprietary	ASTC
WebRISC-V	github	BSD 3-clause	Gianfranco Mariotti, Roberto Giorgi (University of Siena)
PQSE	website	Proprietary	PQShield
riscv-rust	website github	MIT	Takahiro Aoyagi
terminus	github	MIT	Yang Li
Vulcan	github	MIT	Victor Miguel de Morais Costa
riscv-vm	github	MIT	Aidan Dodds

Showing 1 to 25 of 25 entries

Terminus

- https://github.com/shady831213/terminus

- [x] RV32/64I
- [x] MADFC
- [x] M/S/U priviledge
- [x] Pass all riscv_tests
- [x] CLINT and Timer
- [x] HTIF console
- [x] FDT generation
- [x] Multi Cores
- [x] Boot Linux
- [x] Emu mode binary
- [x] Boot Linux(smp)
- [ ] Publish to crate.io
- [x] PLIC
- [x] VirtIO console
- [x] VirtIO disk
- [x] VirtIO network
- [x] framebuffer
- [x] VirtIO keyboard
- [x] VirtIO mouse
- [ ] debug mode
- [ ] other extensions(b, v ...)

Why Terminus?

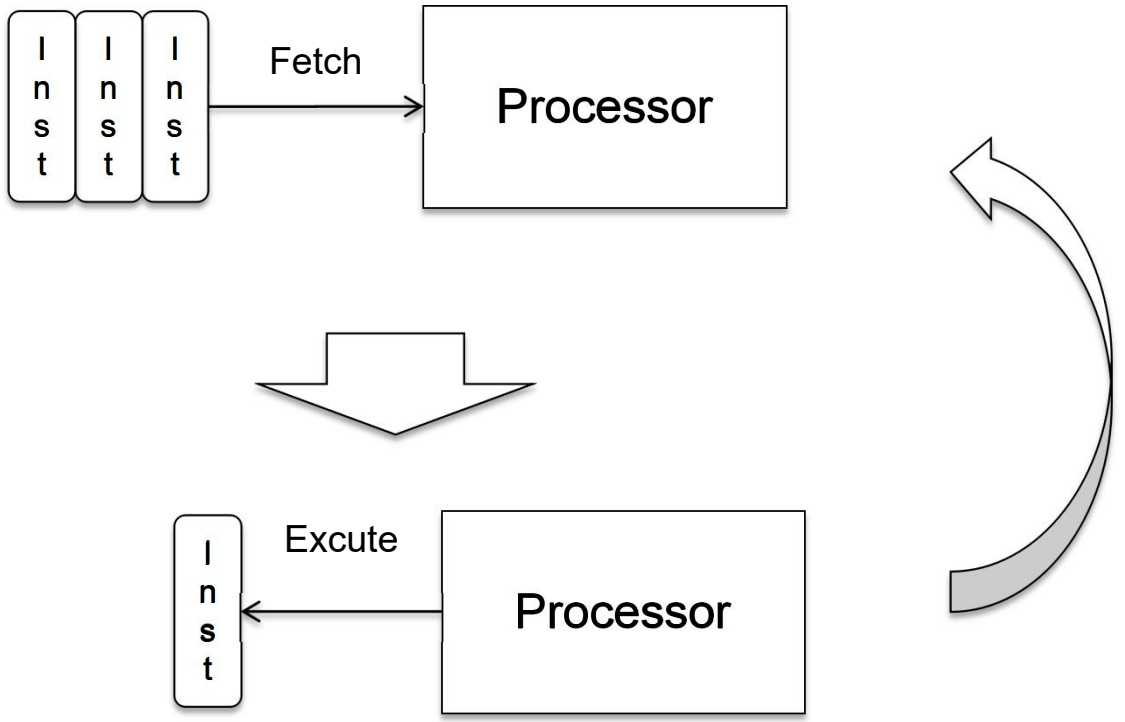
- Just for fun，疫情期间找点事情
- 学一手Rust, 写个不是太简单的东西
- 更深理解RISCV的体系结构，动手做一遍
- https://bellard.org/tinyemu

Why Rust?

- 体验一门新的语言
- 快
- 方便的构建工具，容易使用的单元测试框架
- 轮子
- 泛型，宏，模式匹配，闭包。。。

How?

- Processor 是什么?
  - 状态
    - 寄存器
    - csr
    - pc
    - icache, dcache, tlb..
    - ...
  - 改变状态的操作
    - 指令
    - 中断
    - ...



How?

```rust
pub struct Processor {
    xreg: [u64; 32],
    pc: RegT,
}
```

```rust
pub struct Instruction(Box<dyn InstructionImp>);
```

```rust
pub trait Execution {
    fn execute(&self, p: &mut Processor) -> Result<(), Exception>;
}
```

How?

- 指令如何描述?
  - 我希望是:
    - 指令改变processor 的一系列状态
    - 每条指令应该是可以独立描述的
    - 每条指令的属性应该是集中定义的
    - 要方便添加新的指令
    - 指令应该是无状态的
    - 相似的指令应该是可以被抽象的

How?

- 指令如何描述?

```rust
trait Jump: InstructionImp {
    fn jump<F: Fn(&ProcessorState, Wrapping<RegT>) -> Wrapping<RegT>>(&self, p: &mut Processor, target: F) -> Result<(), Exception> {
        let offset: Wrapping<RegT> = Wrapping(sext( value: ((self.imm(p.state().ir()) >> 1) << 1) as RegT,  len: self.imm_len()));
        let t = target(p.state(), offset).0;
        if let Err(_) = p.state().check_extension( ext: 'c') {
            if t.trailing_zeros() < 2 {
                return Err(Exception::FetchMisaligned(t));
            }
        } else if t.trailing_zeros() < 1 {
            return Err(Exception::FetchMisaligned(t));
        }
        let pc = t;
        p.state_mut().set_pc(pc);
        let rd :u32  = self.rd(p.state().ir());
        let value = *p.state().pc() + 4;
        p.state_mut().set_xreg( id: rd, value);
        Ok(())
    }
```

```rust
#[derive(Instruction)]
#[format(I)]
#[code("0b?????????????????000?????1100111")]
#[derive(Debug)]
struct JALR();

impl Jump for JALR {}

impl Execution for JALR {
    fn execute(&self, p: &mut Processor) -> Result<(), Exception> {
        self.jump(p,  target: |state, offset| { offset + Wrapping(*state.xreg(self.rs1(state.ir()))) })
    }
}
```

```rust
#[derive(Instruction)]
#[format(J)]
#[code("0b?????????????????????????1101111")]
#[derive(Debug)]
struct JAL();

impl Jump for JAL {}

impl Execution for JAL {
    fn execute(&self, p: &mut Processor) -> Result<(), Exception> {
        self.jump(p,  target: |state, offset| { offset + Wrapping(*state.pc()) })
    }
}
```

How?

- 指令如何描述?
  - 怎么实现?
    - PROC MACRO
    - https://github.com/shady831213/terminus/blob/master/proc_macros/src

How?

- Decode?
  - （编码 -> 某个指令实例） -> 应该是个decode tree
  - 有什么特点?
    - 最大深度32
    - 应该是稀疏的
    - 运行时是不变的
    - 是被所有processor共享的
  - 所以我希望:
    - decode tree 应该是单例的
    - decode tree 应该可以支持分布式的节点插入
    - decode tree 应该可以路径压缩
    - decode tree 在建立时完成所有可变操作，而在建立完成后时不变的
  - https://github.com/shady831213/terminus/tree/master/src/processor/decode

How?

- Decode?
  - decode tree 应该可以支持分布式的节点插入？
    - https://crates.io/crates/linkme
    - 把slice放到link到特殊的section，有平台依赖的限制

How?

- 测试?
  - https://github.com/riscv/riscv-tests
  - https://github.com/shady831213/terminus/blob/master/top_tests/riscv_tests.rs

What's next?(If I have time to do :))

- Support Debug
- Support More Extension, b, v...
- Try run other OS...

- Other arch...
- More device...

谢谢！