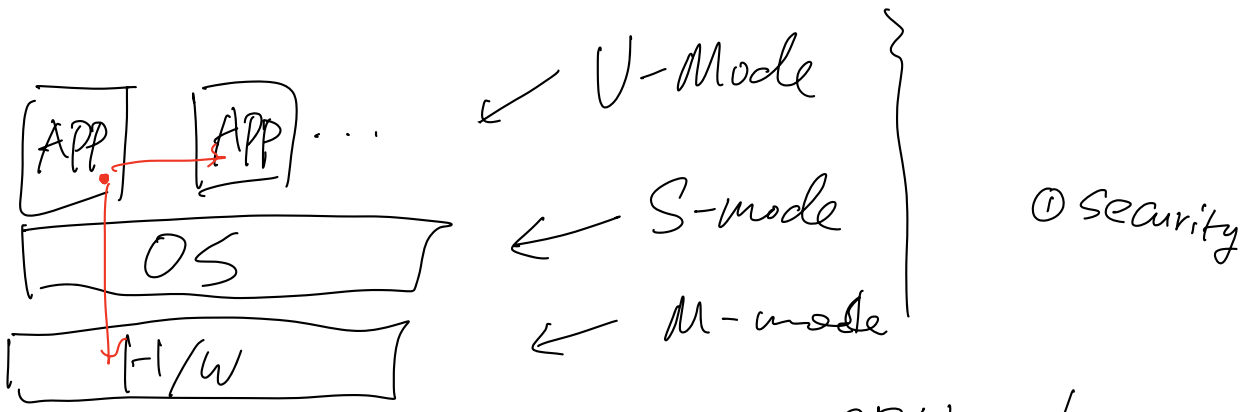
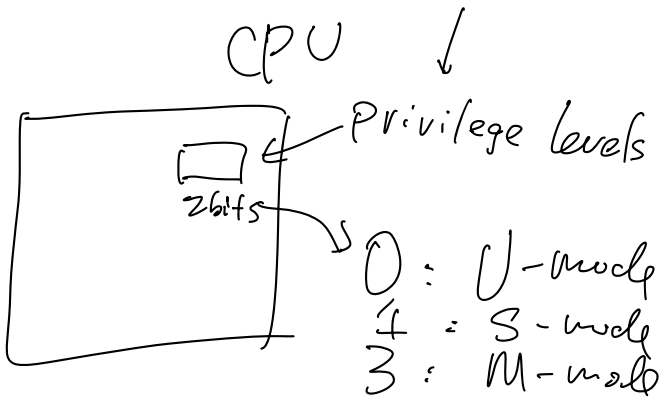
Week 7.a
CS6640
10/16 2023
https://naizhengtan.github.io/23fall/

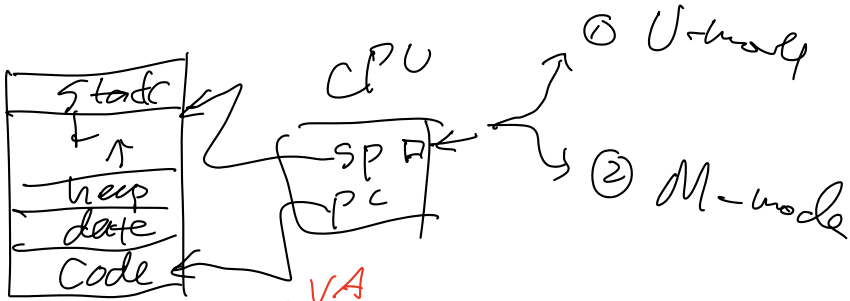
1. CPU privilege levels
2. Virtual machines
3. Trap-and-emulate virtualization

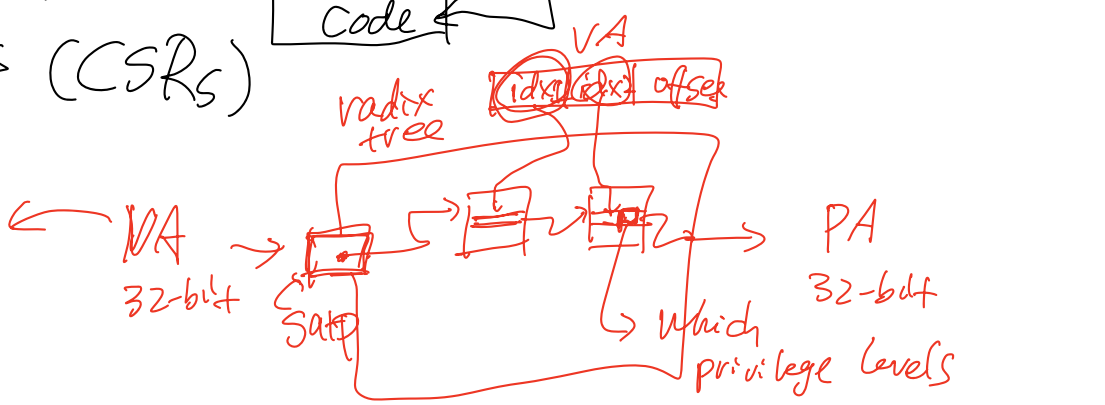


- Why Privilege:
  - security
  - isolation
  - (virtual memory)
  - multiplex resource
  - abstraction



- instructions
  (sfence.vma)
- registers (CSRs)
  (mstatus)
- Memory

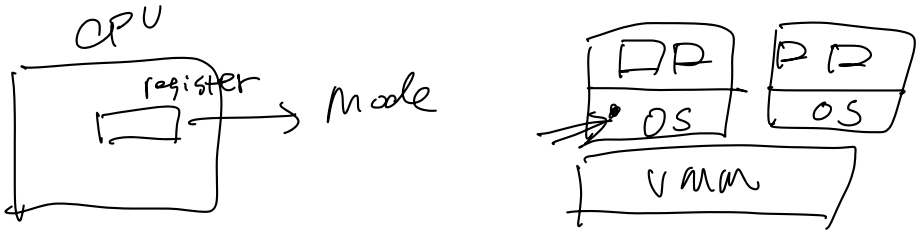




- exception / interrupts
- ecall
- mret

"""
RISC-V deliberately doesn't make it easy for code to discover what mode it is running it because this is a virtualisation hole. As a general principle, code should be designed for and implicitly know what mode it will run in. Applications code should assume it is in U mode. The operating system should assume it is in S mode (it might in fact be virtualised and running in U mode, with things U mode can't do trapped and emulated by the hypervisor).
"""

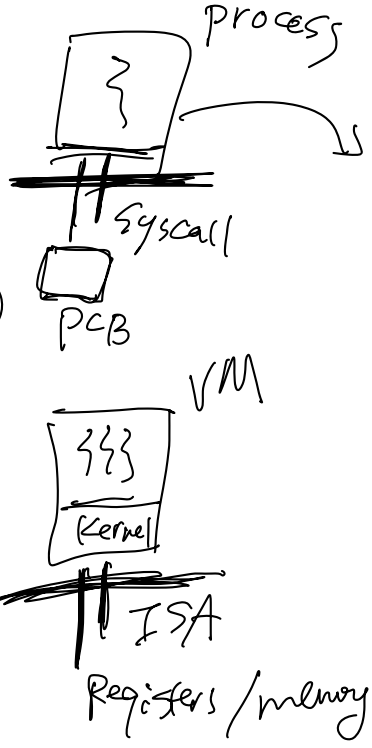
[from https://forums.sifive.com/t/how-to-determine-the-current-execution-privilege-mode/2823]



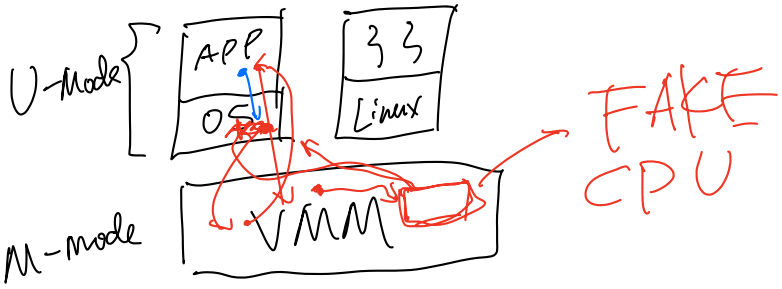
2. VM.

↳ simulation of a computer, accurate enough to run an O/S.

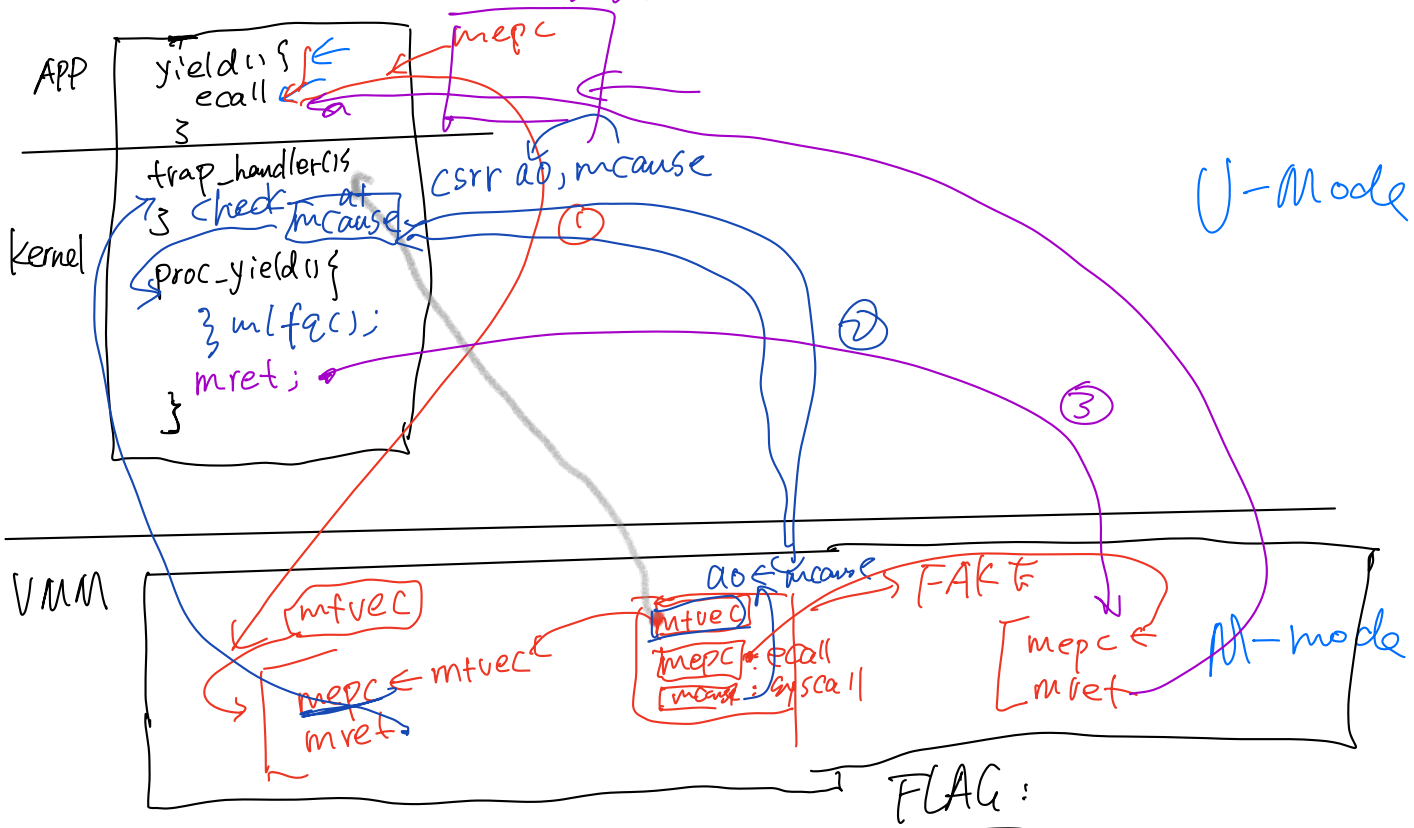
- isolation
- clean setup
- multiple VMs on the same machine
- framework (unified env)
- play games on windows.
- migrate, suspend, resume

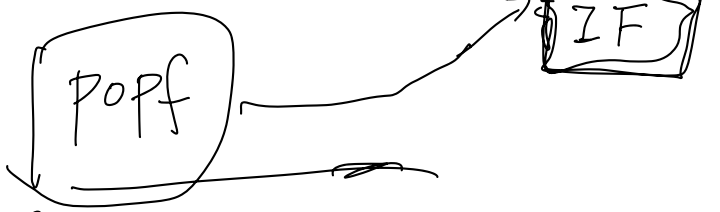


Q: How to implement a VMM?



APP

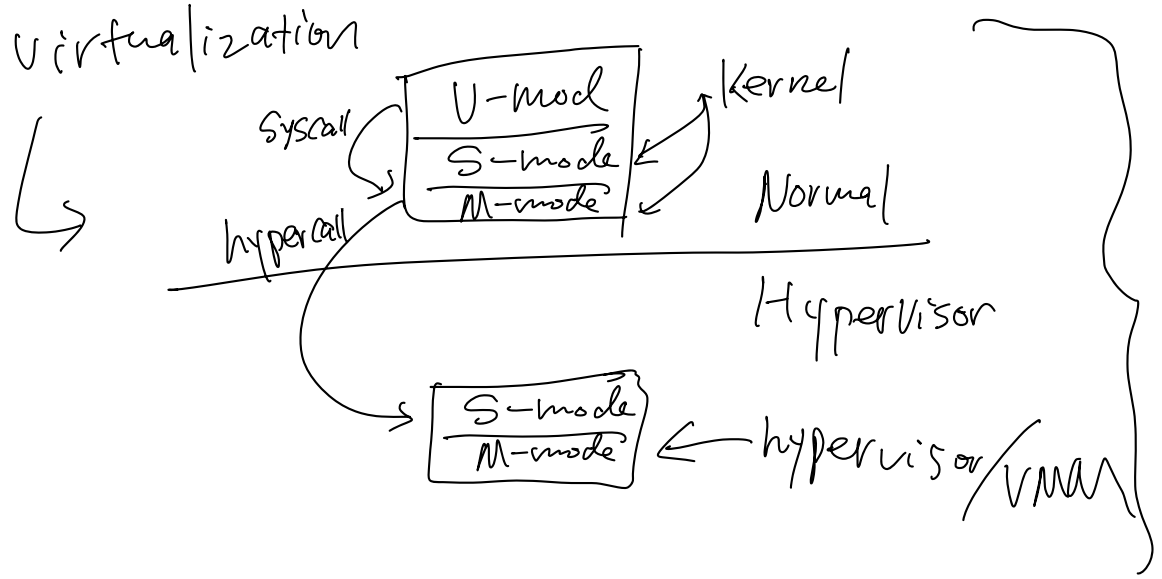




FLAG:

X86:

- VMware
- para-virtualization: Xen
- HW virtualization



Q: what RISC-V state must a trap-and-emulate VMM "virtualize"?
  * all "privileged CPU state"
    CPU state that the guest kernel assumes it can read/write
    but is forbidden by user mode (plus VMM needs to protect for security)
  * all CSR registers (mepc, mtvec, mcause)
  * page table (satp)
  * PLIC/CLINT

  (32 registers and memory are virtualized by processes already)