Approach 2 : virtualizing the CPU/MMU

➡ Observations - most instructions are the same regardless of processor privileged level e.g. `incl %eax`

Why not just give instructions to CPU to execute?

- Problem - safety
  How to prevent privilege instructions from interfering with hypervisor and other OSes?
- Solution - use protection mechanisms already in CPU

➡ "Trap and emulate" approach

- run virtual machine's OS directly on CPU in unprivileged user mode
- privileged instructions trap into monitor and run simulator on instruction

Virtualizing interrupts

OS assumes to be in control of interrupts via the interrupt table

So what happens when an interrupt or trap occurs in a virtual environment?

➡ The VMM handles the interrupt (in kernel mode) using the "virtual" interrupt handler table of the running OS

✓ Some interrupt can be shadowed