# Approach 1 : complete machine simulation

➡ Simplest VMM approach, used by B*ochs*

Build a simulation of all the hardware

- CPU – a loop that fetches each instruction, decodes it, simulates its effect on the machine state (no direct execution)
- Memory – physical memory is just an array, simulate the MMU on all memory accesses
- I/O – simulate I/O devices, programmed I/O, DMA, interrupts

◉ Too slow!

- CPU/Memory – **100x slowdown**
- I/O Device – 2x slowdown

➡ Need faster ways of emulating CPU/MMU

# 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