§2. INTRODUCTION

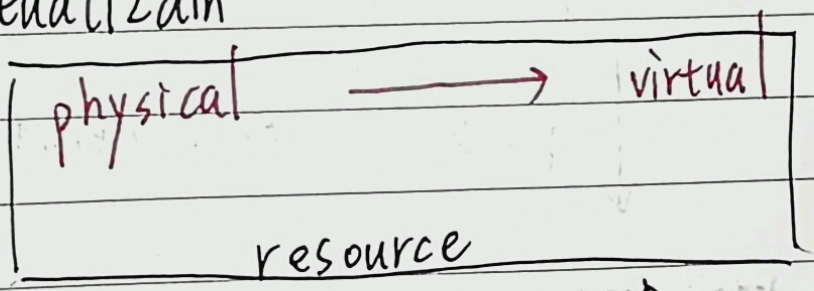
program —run→ execute instruction

① fetch
② decode
③ execute

von Neumann.

2.1

- virtualizain



- role of OS.
  - virtual machine
  - standard library API
  - resource manger share { cpu, memory, device

• 2.1 virtualize CPU

single CPU → seemingly infinite num

many progs. seemingly run @ once

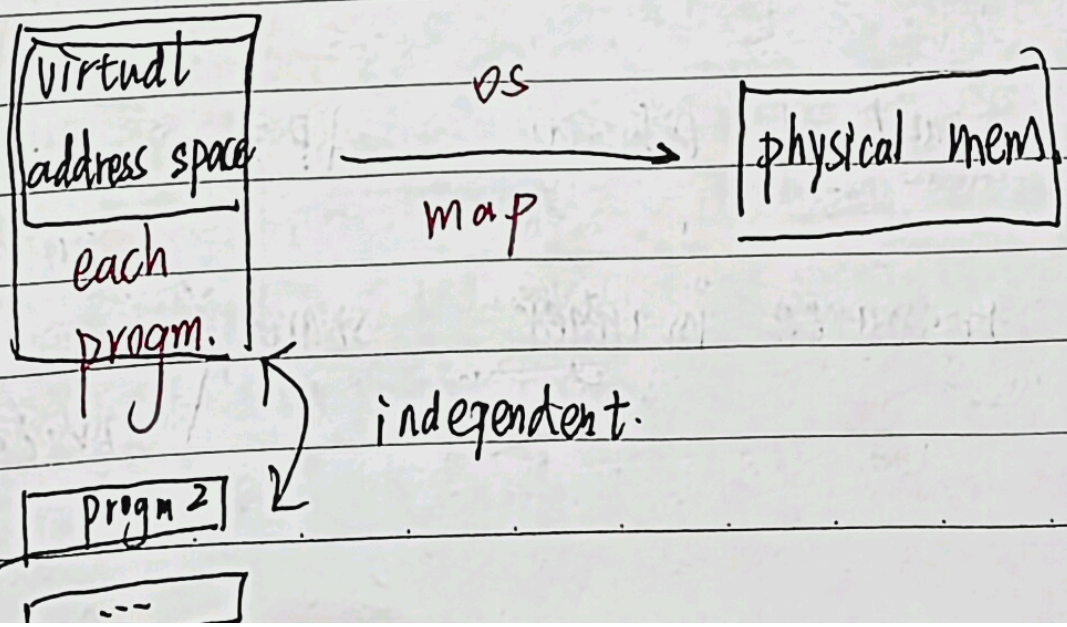
↓

new questions ...

↓ policy ... etc. mechanisms

resource manager

• 2.2 virtualize memory.



• 2.3 concurrency

shared resource ← write ← a thread

① load to register

② update

③ store

not atomic

context switch

(if high concurrency)

thus loop many times.

error

• 2.4 persistent

file system

① where on disk

② maintain metadata

③ I/O, device driver

call API

open()

write()

close()

write file

performance better → delay. buffer

large batch

if crash during writes

write protocol

recover afterward

• 2.5 Design Goals

virtualize concurrency presistent to do.

- abstraction.
- minimize the overheads
- protection → isolation
- reliability (non stop)
- energy - efficient
- security
- mobility

• 2.6 Some History

- Early OS.

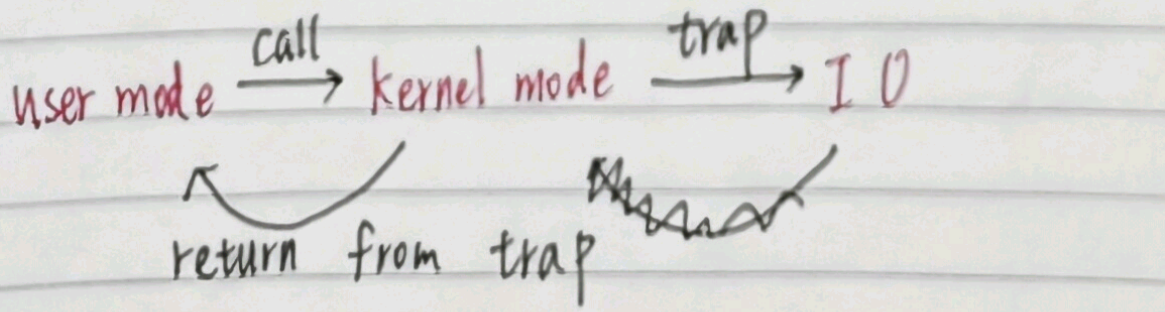
  library → API

  batch processing (human operator)

- Protection

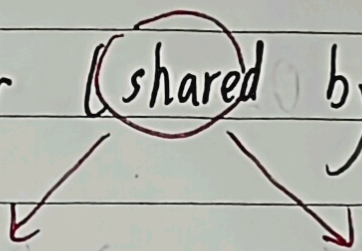
  procedure call → system call

  ~~user mode → trap kernel mode~~



- Multi Programming

  mini computer (shared by workgroup)

  memory protect concurrency

- Modern Era.

  personal computer.

  DOS. MacOs → windows NT

  lessons in mini computer