CS 347 Lecture #3

- Lab1 is available.
- Lab2 will be out this week.

von-neumann model.

fetch (from address/location stored in PC)
decode
execute —— (may update PC)
repeat

OS requirements

- useful abstractions
- efficient resource usage
- ~~root~~ robustness / failure handling
- isolation / "contained/restricted" interference

Challenges

~ monopolizing resources

~ unintended ~~state~~ / data/ memory sharing

e.g.: interval timer register ~ regs. available in some ISAs that counts down & on zero stops the current appl'n. on CPU to possibly do something else (execute different application)

Q who/which entity updates the interval timer register?

(#) main invariant / condition for design of all OSes.

⇒ the OS is the sole owner / arbitrator of all resources (hardware resources)
cpu,
disk, network,
memory ...

(#) two main design principles / building blocks of an OS.

(i) limited direct execution ~ LDE (on the CPU) ‖ privilege levels of execution.

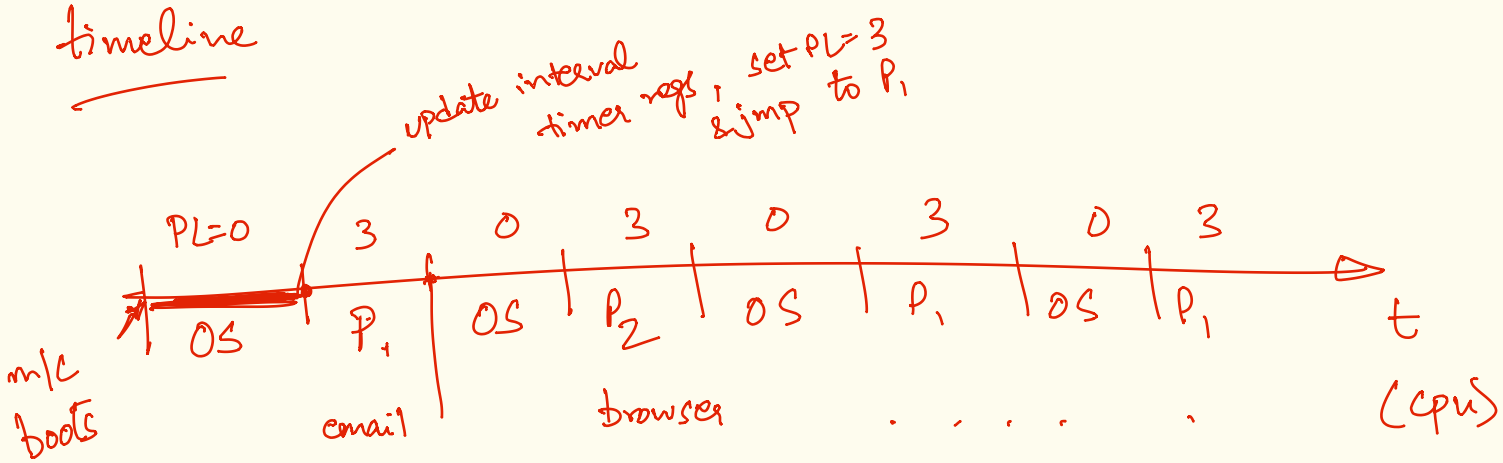
(ii) interrupt-driven execution.

(i) LDE ~ ISA supported feature

~ the CPU executes with a privilege level
~ the PL can be set/updated
~ every instruction of the ISA has a minimum privilege level specified for correct execution semantics.
~ critical / sensitive instructions
└ alter system usage behaviour

if PL ≠ req. min PL
~ error
exception
undefined

timeline



x86 — privilege levels

Ring 3 ——— lowest ~ appl^n's. (user-mode)

2

1

0 ——— highest — OS (Kernel mode)

{ set by 2-bits in a control register

(ii) interrupt-driven execution / interrupts

what is an interrupt?

① ⊢ interrupting / pausing / stop the current sequence of instructions

③ ~ after current instruction executed, the PC changed to jump to some other, pre-defined location

② (*) ⊢ Save state of current execution
- PC
- cpu registers.

mov 23, #b
add #b, #a ← interrupt

③ ~ s/w to PL=0 & jmp to a loc^n in the OS program

why interrupts?

- everything in the world are non-deterministic!
- all IO is non-deterministic (in time)
  - KBD / mouse
  - disk read
  - arrival of n/w packets

types of interrupts

- hardware — IO
- Software (via a program instructions)
  - implicit
    - exceptions
      - div by zero
      - null ptr.
  - explicit