

OS has to be sole owner
coordinator
 to manage all resources

one rule
 that cannot be
 violated.

two building blocks
 of all OS design -

(i) 2DE — limited direct execution.

Use ISA-supported privilege levels to
 setup user & kernel mode to control
 supervisor access to
sensitive instructions
 & runtime behavior.

(ii) interrupt-driven / interrupts.
 execution

Q what are interrupts?

- an event (ISA-supported) that stops/pauses/interrupts the current sequence of instructions
- pause after current instruction is executed.
- slow to handling the interrupt!

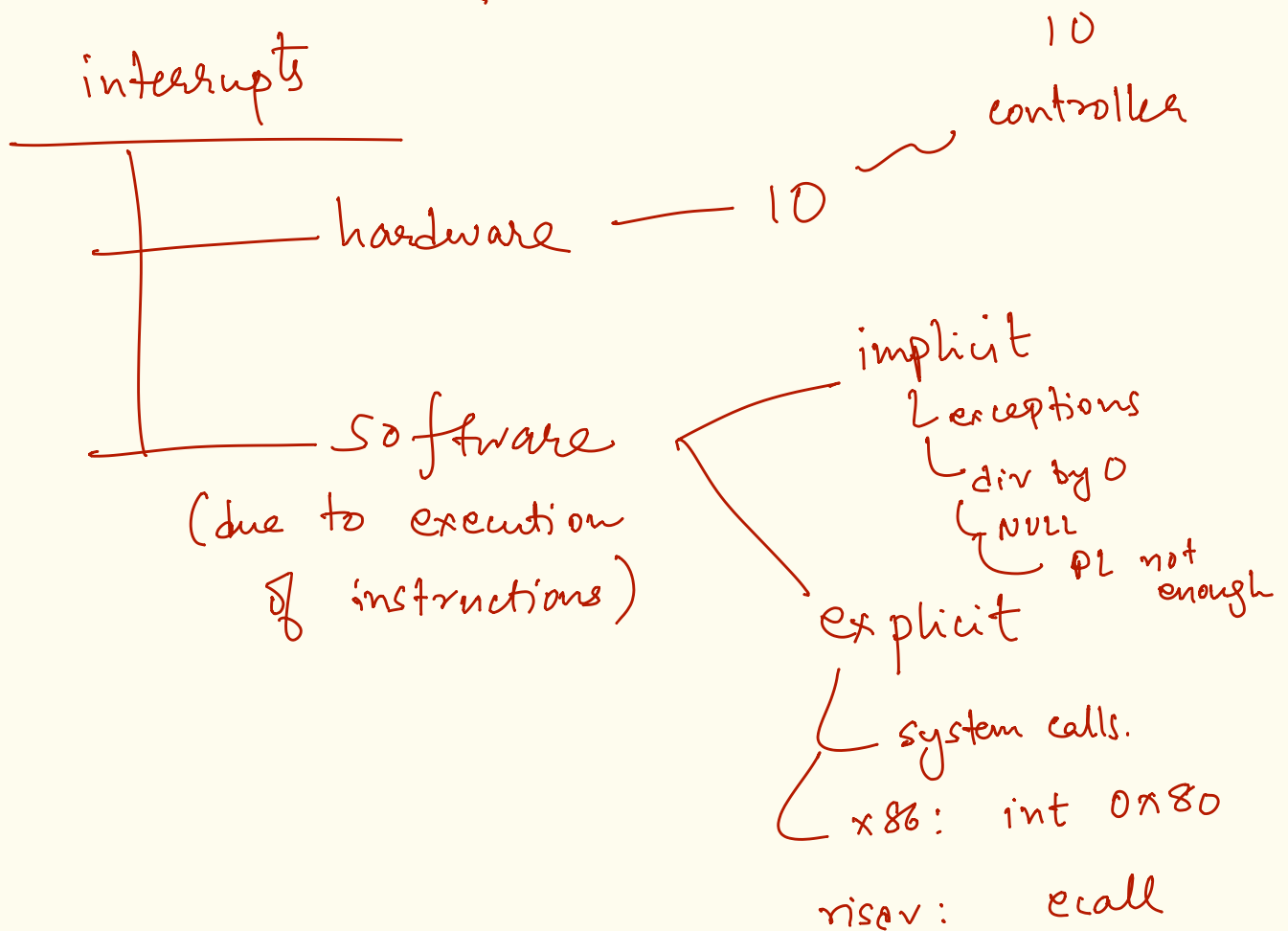
Lab1 (due)
 Lab2 (available)
 Lab session
 (coming up)
 TA office hours
 (TBA)
 - today 2.30pm

Q why interrupts?

- the world is non-deterministic! (in time axis)
- all IO is non-deterministic.
- IO events are signaled via interrupts
- KBD, n/w, disk, ...

#

interrupts



#

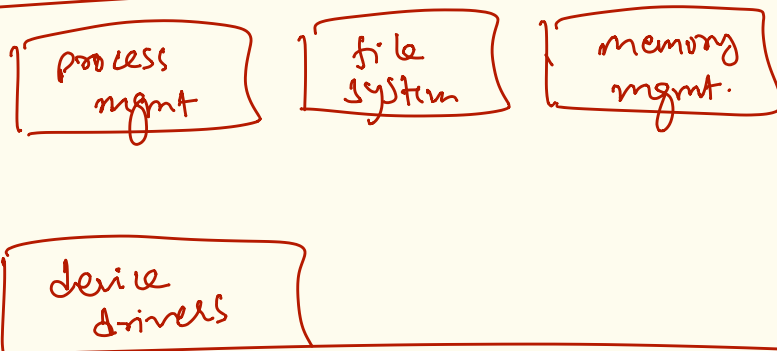
hw shell email

user mode

libc

kernel mode

OS



OS abstractions

process & threads
file
sockets
signals
users, groups

hw

cpu, net, mem, disk

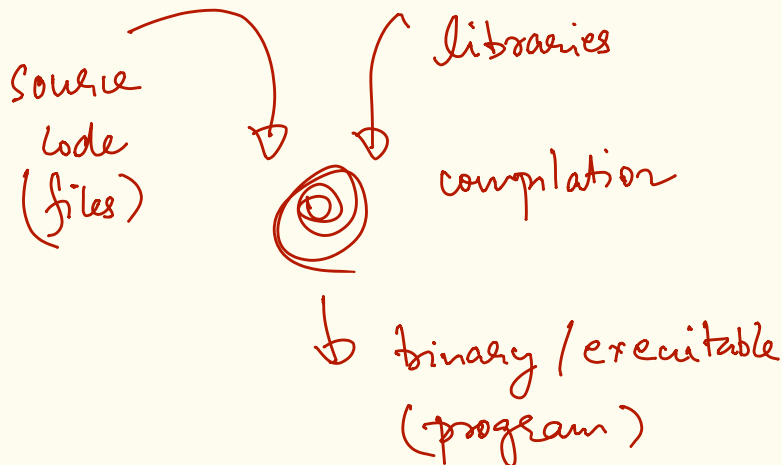
#

process abstraction

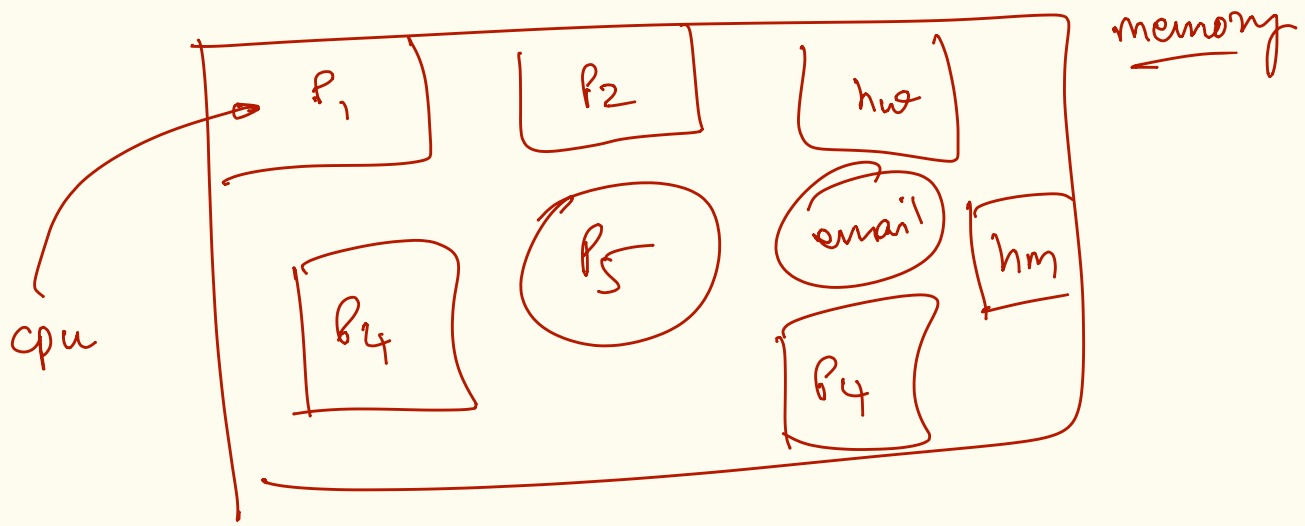
program

vs

process



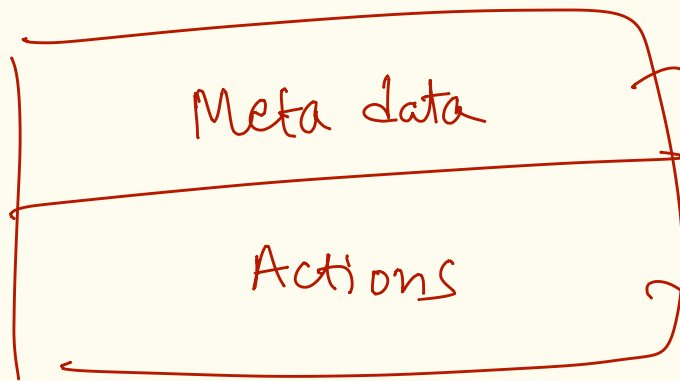
- instance of program execution
- entity that consumes resources
- decouples logic development encoding from execution of work.



cab-hailing service

cabs, traffic, routes, cab locⁿ, ... } metadata

every (OS) abstraction can be viewed as two-block view



process abstraction

PCB process control block
per process ~ task_struct (linux)
(xv6) proc

scheduling
memory alloc
open/close/read files...

⑧

the PCB

