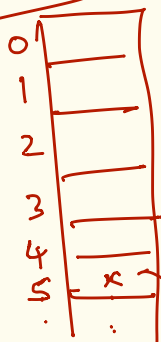


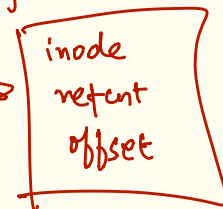
process abstraction.

- PCB

- system calls — fork, exec, wait, exit, waitpid,
getpid, getppid signal, ...# the file-based IO model~ in UNIX-like OSes the file / file-descriptor is used
to reference all IO abstractions — files
sockets~ open, read, write, } system
dup, dup2, pipe } callsuse or return
a file descriptordevices
shared memory
pipes
fifo ...[Q] what is the fd?all open "files"
of a process are
indexed in a fd tablefd table (array) ← is part
of the PCBint fd = open("hw.txt", O_RDWR);read(fd, buffer, num of bytes);
to read

↑ system call

file object

→ in-memory
structure
to
store
info.
about
the file

① `open(hw.txt)`
`open(hw.txt)`

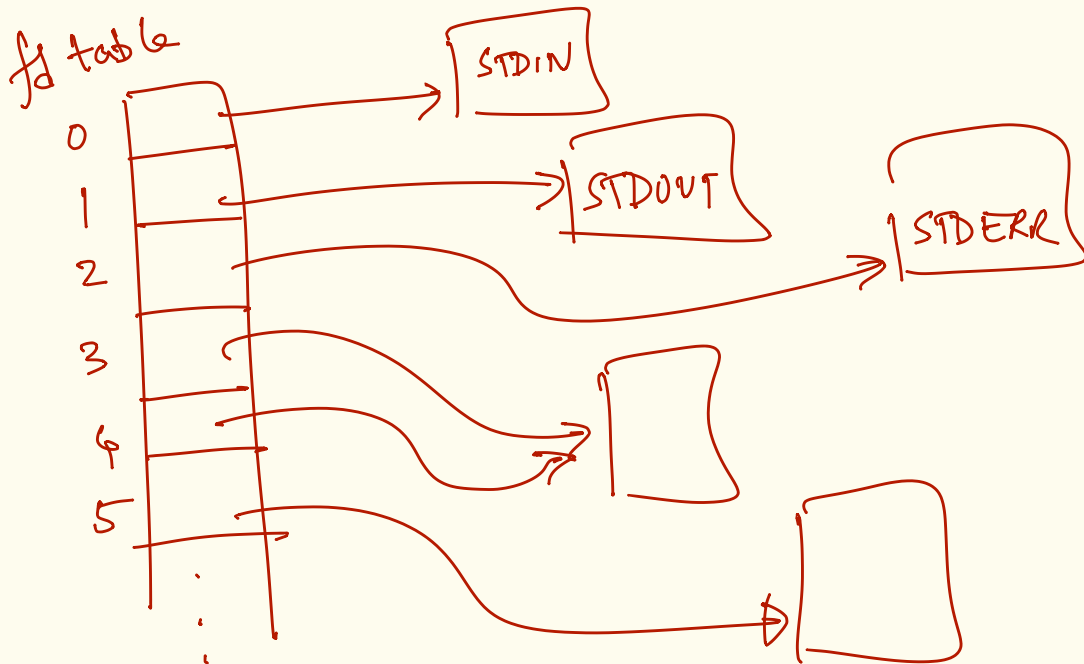
two fds
two file objects

② `new fd = dup(fd);`

both `fd` & `new fd`
point to same file
object.

④ `fd = open()`
`read(4, ...)`
`...`
`close(fd)` ④
`read(4, —)`

fail!



fork: duplicates a process!

— creates new PCB
copies most of PCB — by default
includes
fd table!

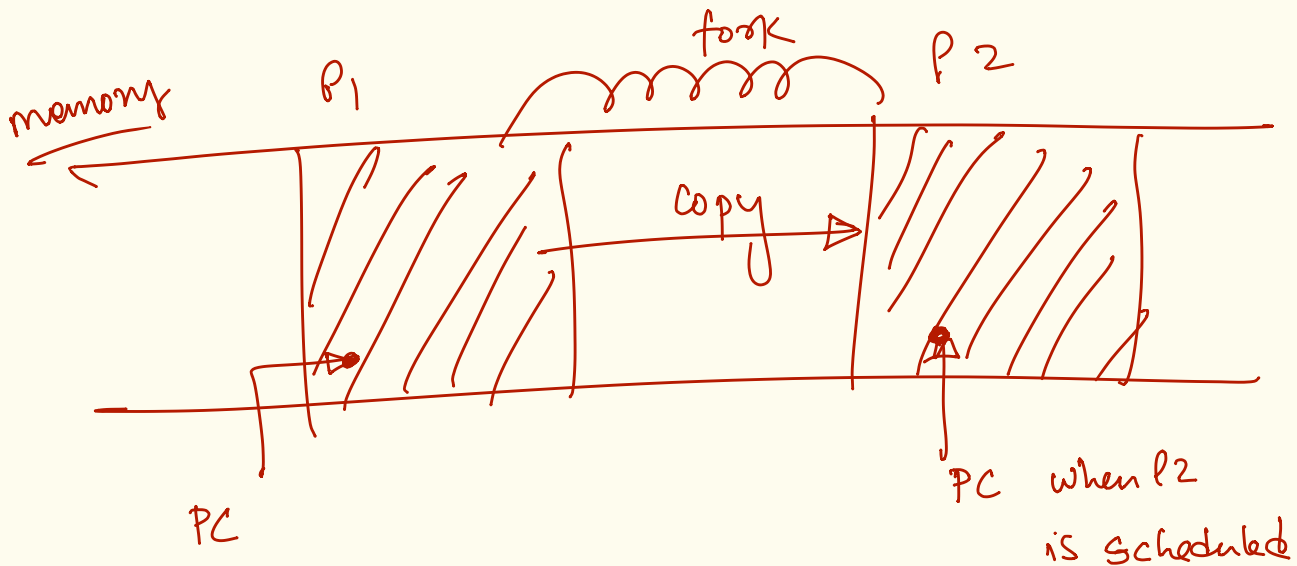
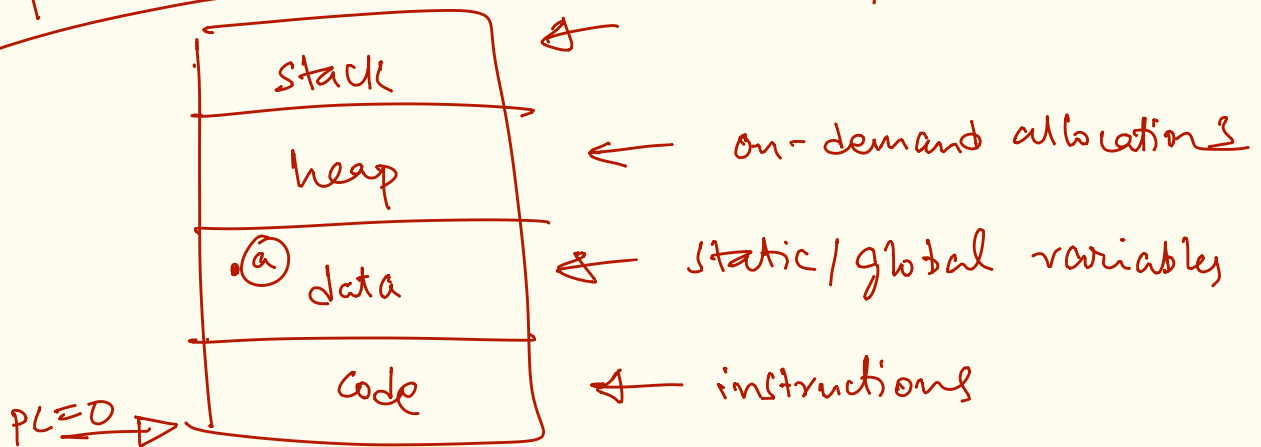
⑧ traditionally, unix-like systems re-use
the first unused fd index for open!

└ 3 → fd = open(...);
read
write
close(fd);

③ ← open();

⑧ Cow — copy-on write optimization.

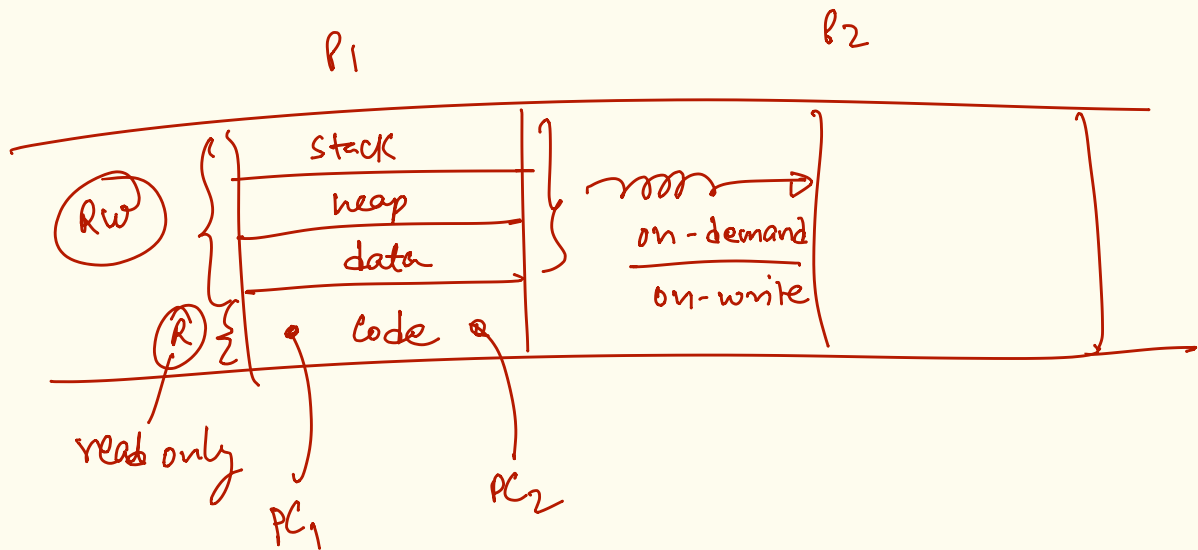
process-view
of memory



challenge : - fork can be costly for large processes

- & fork + exec multiplies all that is copied!

conv: copy-on-write.



- mark all ~~pages~~ read-only memory

- on write → exception → as part of exception handling copies parts of memory (to make

int a=15;

if (fork() == 0)
a = 23;

else
a = 20;

different physical copies).

no Cow	Cow	
23	23	child
20	20	parent