Lecture #5 CS 347 16/8/2023

recap: process abstraction

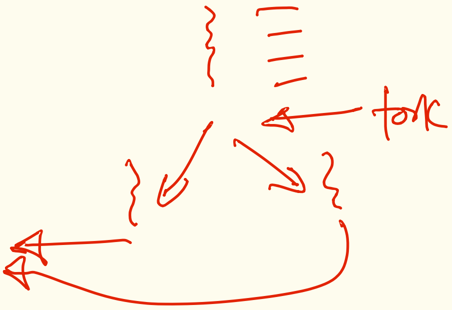
- PCB < pid, ppid, state / memory info.
- OS game plan → handcraft the first user process & start its execution
  - first user process can consume the OS interface/services.

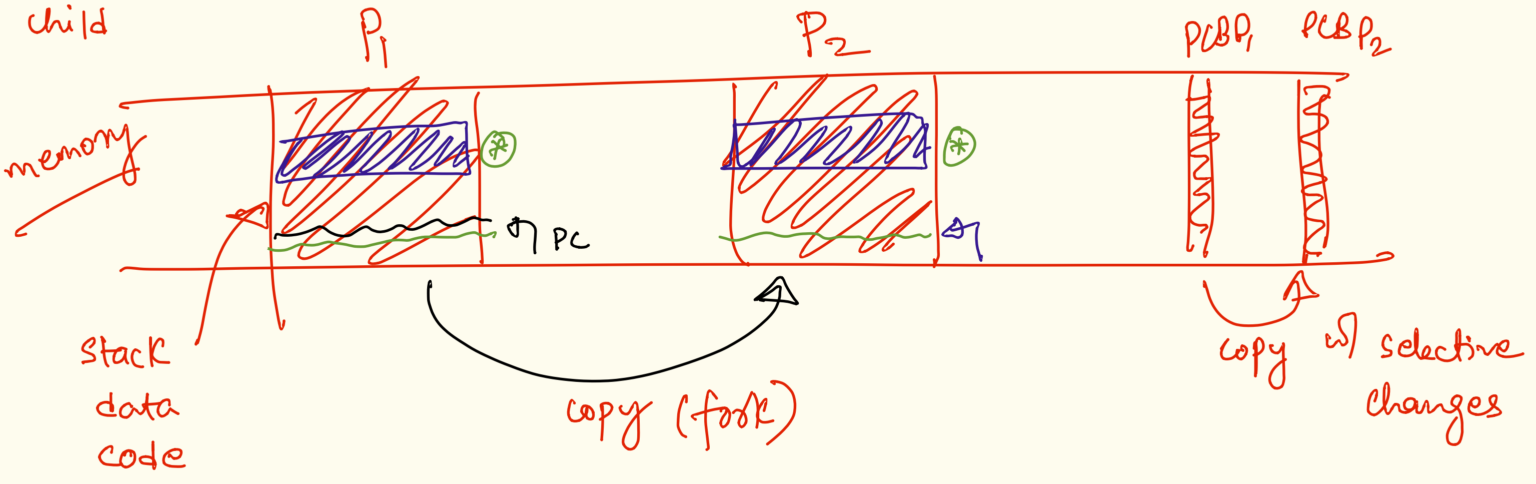
- fork & exec
  - fork
    - duplicates a process
  - exec
    - loads a new program (into a process)

```
int a=23;
fork();
a++;
print(a);
```

returns pid of child in parent

returns 0 in child





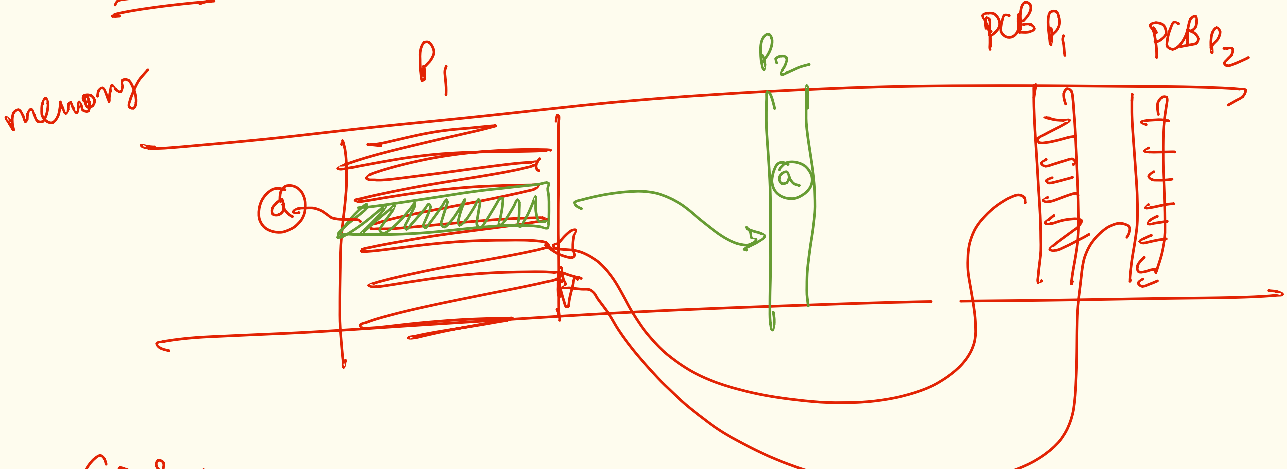
⊛ why fork?

- duplicate & setup a process with custom configurations.
  - default all state is copied (open files etc.)
  - after fork, can set the subset of open files, cwd, etc.
    before
- multi-process parallel work — eg. webservers
- is create a process framework for new programs!

⊛ exec

- loads a new program (in a process) — from disk into process memory
  - code
  - data
- cleans up context (cpu regs)
- sets PC to first instruction — stack & heap

COW — copy-on-write



COW : - share all memory regions || mark all regions "read-only"

- only make copy per process on a write!

(ii) signals

- OS mechanism for inter-process communication (of events)
- process-level interrupt/signaling mechanism
- usage | OS-support.
  - per process pending signal state. in the PCB
  - before a process is scheduled on the CPU, pending signals are processed!