Lecture 32   process mgmt & scheduling
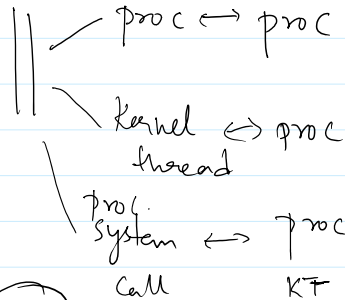09 October 2018   10:39

① first - level user-level process

② creating child processes & returning
   control to user space

③ context s/w mechanism / pcb

④ scheduler w/ scheduling policy

|| ⟍ proc ⟷ proc
   ⟍ kernel ⟷ proc
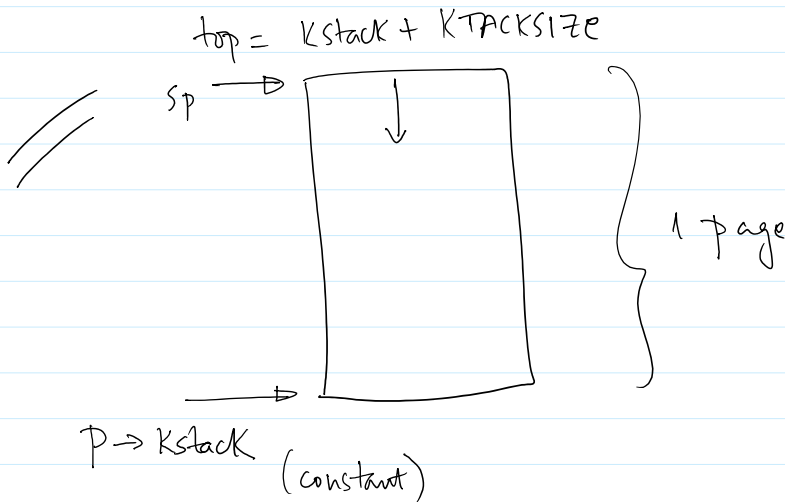   thread
   proc.
   system ⟷ proc
   call   KT

relook at the PCB

struct proc {   XV6

    char * kstack;   ——— // kernel stack ptr. used
                                           when in kernel mode.

    struct trapframe * tf;   copied
                                                    to esp
                                                    regs.
    struct context * context;
                                              user
    :   context
    char * pgdir ; ————————→ CR3   when s/wing
}   to
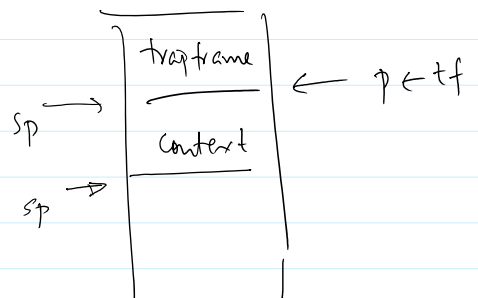   kernel mode

                                          store kernel context
                                          on s/wing between
                                          processes

    top = Kstack + KTACKSIZE

 sp → ┌──────────┐
       │   ↓   │
       │   │ } 1 page
       │   │
       │   │   ⊗ system call
       └──────────┘
                                                  ┌──────────┐
 → P → Kstack   sp ──→   │ trapframe │ ← p ← tf
         (constant)   ├──────────┤
                                                  │ context │
                                          sp →   └──────────┘
                                                       │

sp

$p \Rightarrow$ kstack

#(2)

— when is the scheduler invoked?

— what happens on trap?

— what happens on new process?

exit   timer   yield   exception
        ~~yield~~ ~~yield~~ error
                  sleep   ~~Trap~~

sched ( )