Week 6.b
CS6640
10/12 2023
https://naizhengtan.github.io/23fall/

1. kernel ~= three handlers
2. egos syscall implementation
3. egos exception handling
---

pid2idx / idx2pid ← #define

Last time:

CPU ↳ earth.S : _enter] → earth.elf
↳ earth.c : main
[init HW]

↳ grass.S : _enter] → grass.elf [128KB]
↳ grass.c : main
[syscalls]

↳ app.S : _enter →
↳ sys_proc.c : main
⋮

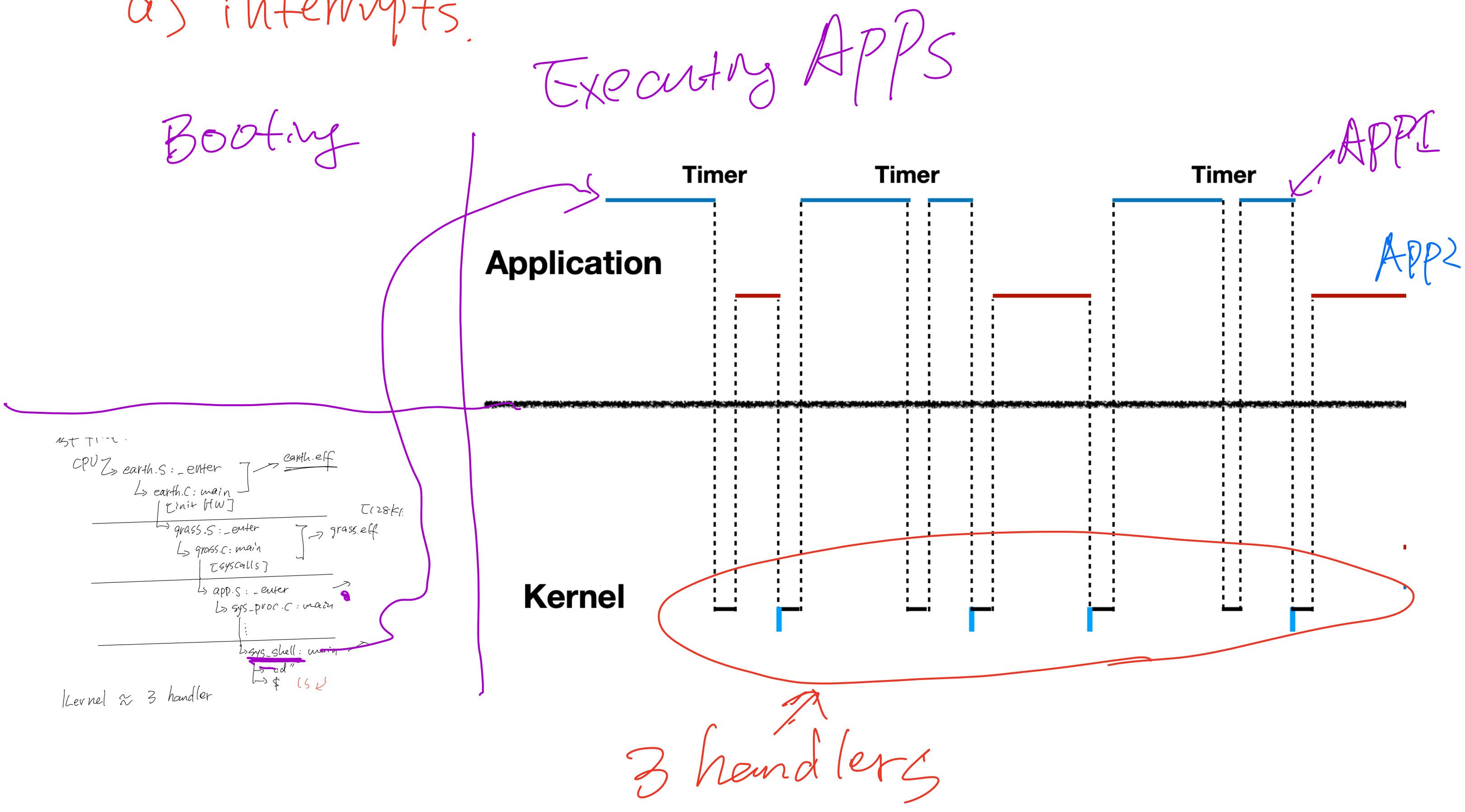
↳ sys_shell : main →
↳ "cd"
↳ $ ls

1. Kernel ≈ 3 handler

Q: What're the 3 ways to trap to kernel?

interrupts, exceptions, syscalls

a) interrupts.



Q: How does a CPU know what to execute when a interrupt is triggered?

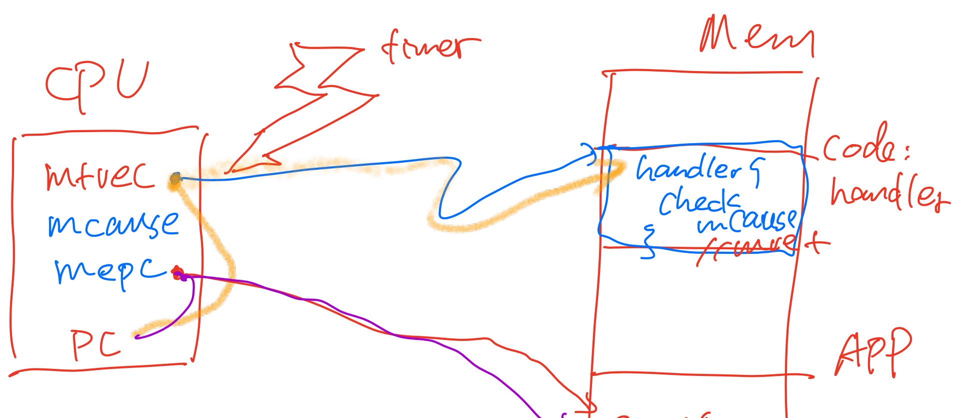
mtvec

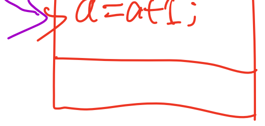
Q: How does kernel know which interrupt has been triggered?

mcause

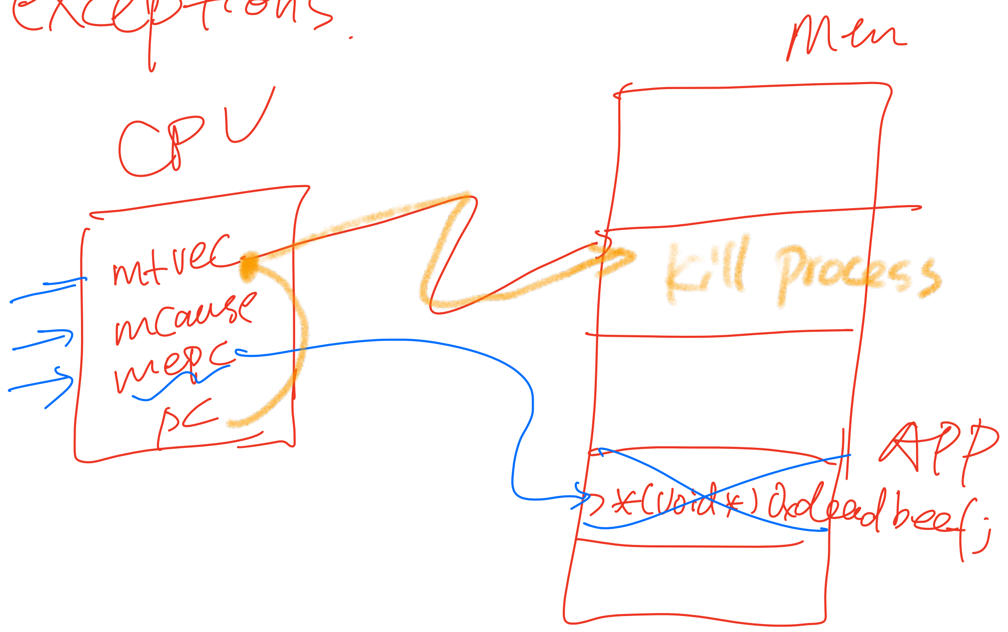
Q: After handling interrupts, where does CPU ~~to~~ jump to?

mepc



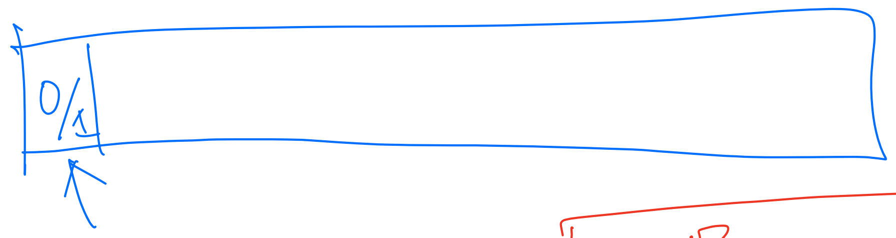


b) exceptions.

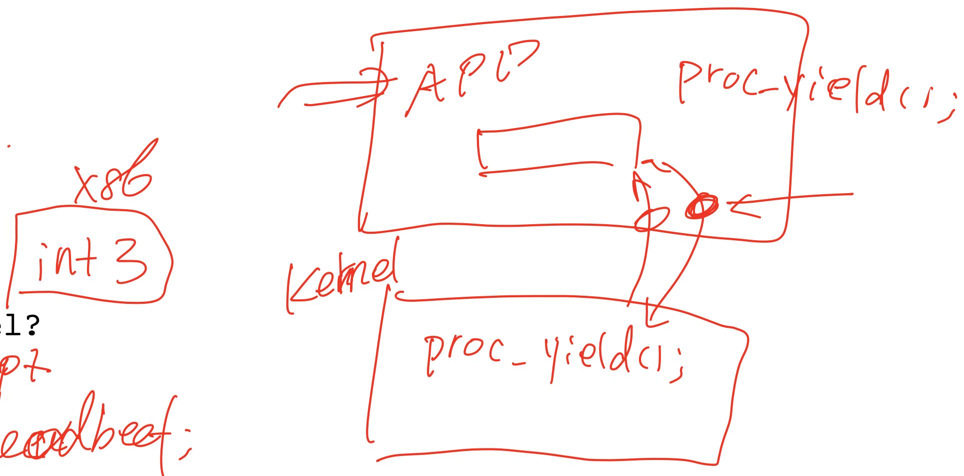


Q: interrupts vs. ~~exe~~ exceptions?

mcause (32bits)



c) Syscalls.



Q1: how to trap to kernel?

- software interrupt
- *(void*) 0xdeadbeef;

Q2: how does kernel understand what the application wants?
That is, what information is needed for handling a syscall?

- syscall type
- arguments
- address to return to (mepc)
- return val

Q3: Where to store the information?

- registers
- well-know memory
- stack

2. egos-2K+ syscall implementation

Q1 : sys_inoke()

Q2 : struct syscall (grass/syscall.h)

Q3 : SYSCALL_ARG

3. exception handling.

Q: How much Mem "malloc" can get?

- 8GB
- 16GB (2)
- 320GB
- 100GB (2) $\longleftarrow$

170 GB