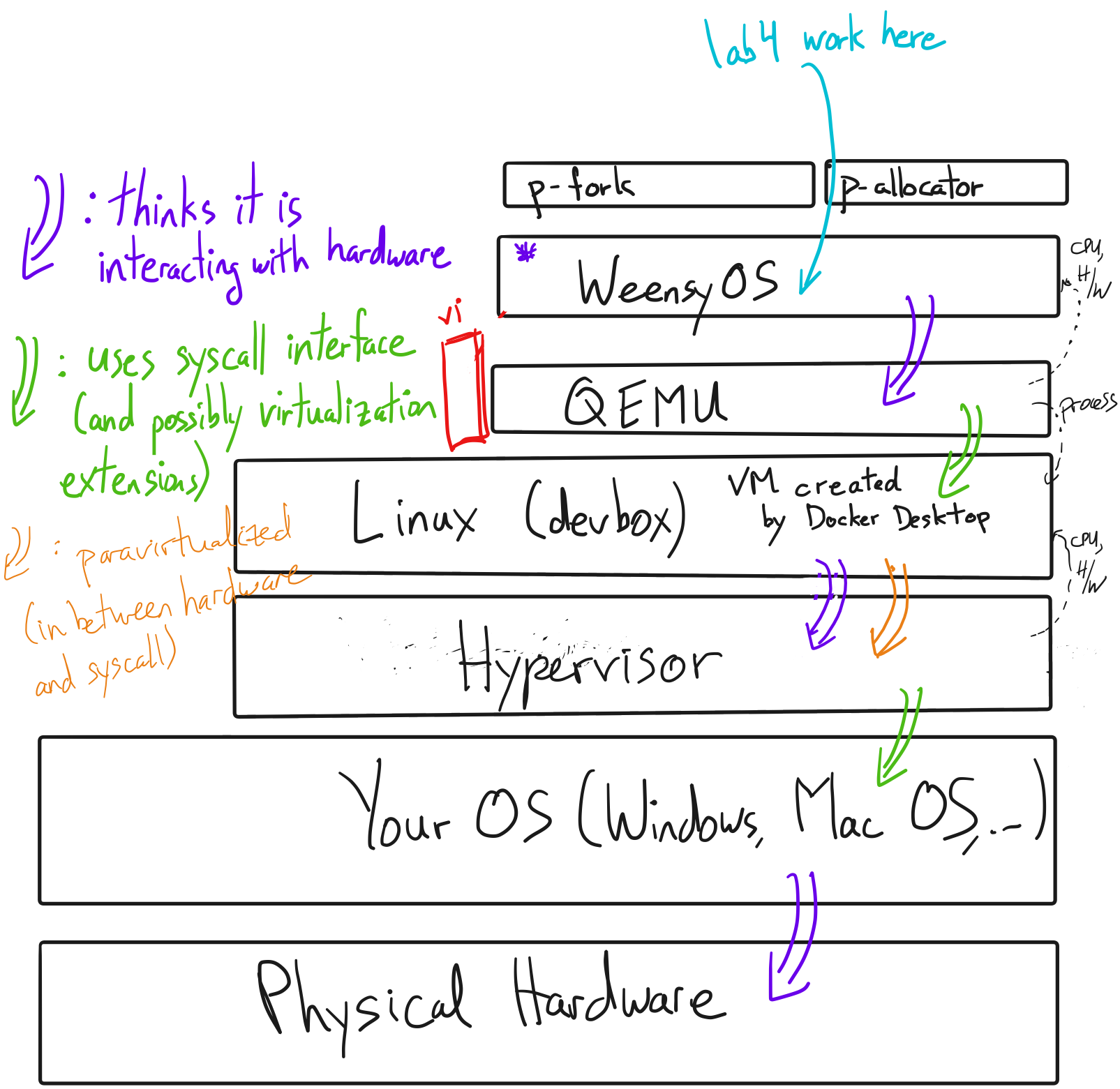
- [ ] 1. Last time
- [ ] 2. Weensy OS walkthrough
- [ ] 3. Weensy OS context switches
- [ ] 4. gdb



Hints:

- processes: files matching `p-*.c`
- kernel code: files matching `k-*.c`, `k-*.s`
- system calls and returns

```
void *sys_mmap(void); /* cousin of mmap() */
```

sys_page_alloc();

look in process.h

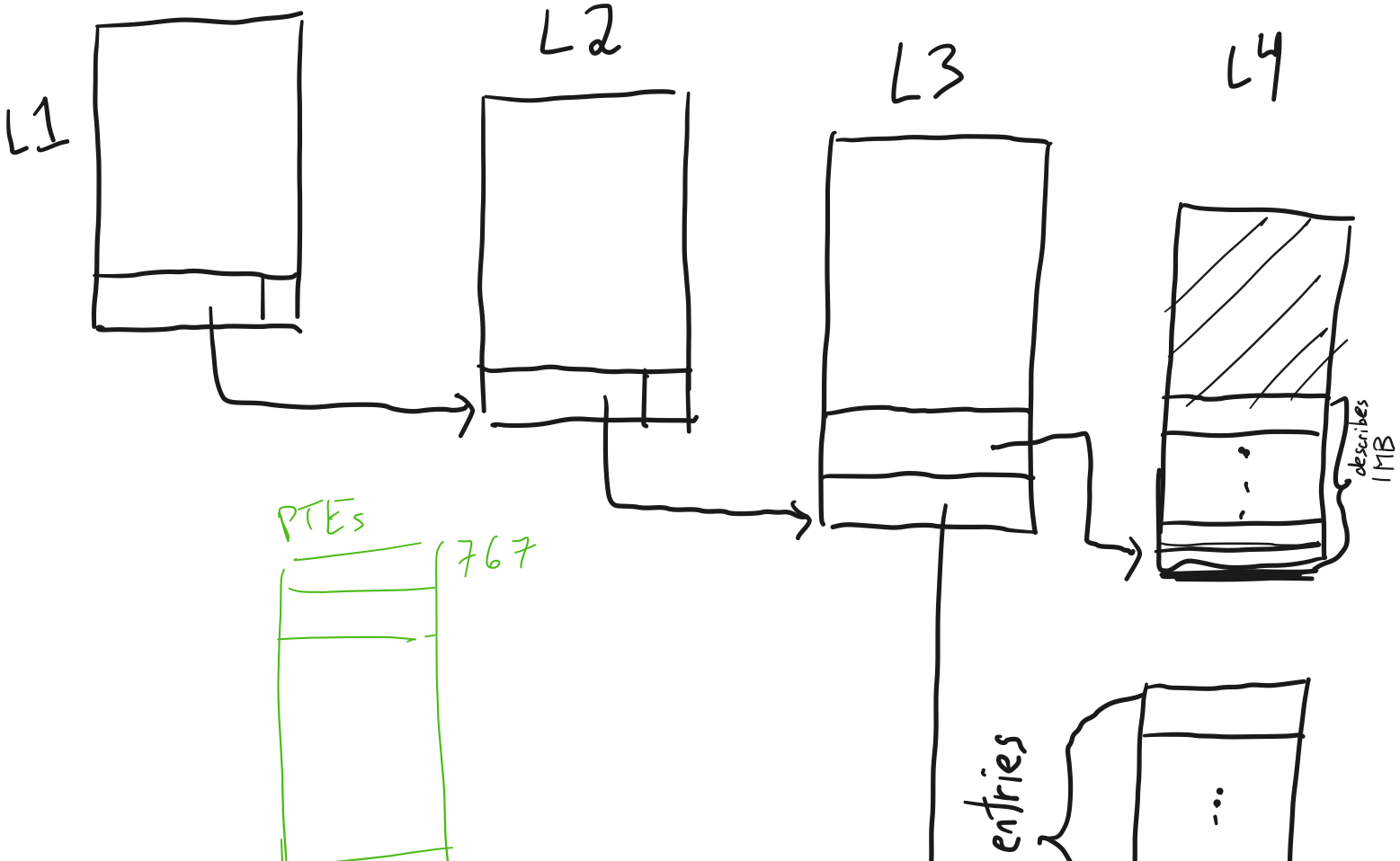
kernel returns: exception_return();
%rax contains return value of system call

%rdi: arg to syscall

<VPN, PPN>

- you'll use virtual_memory_map() =
  - pay attention to the allocator argument (NULL vs. non-NULL)
  - make sure your allocator initializes the page table
    memset(addr, 0, len);

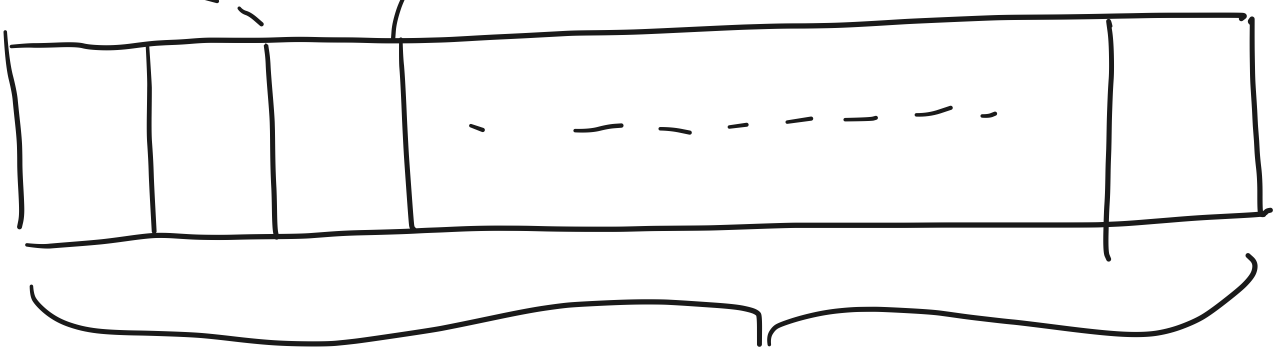
- a process's virtual address space: 3 MB. What's the page table structure?



each entry
describes a mapping
for one page
(4KB)

PCB ≡ struct proc (in kernel.h)

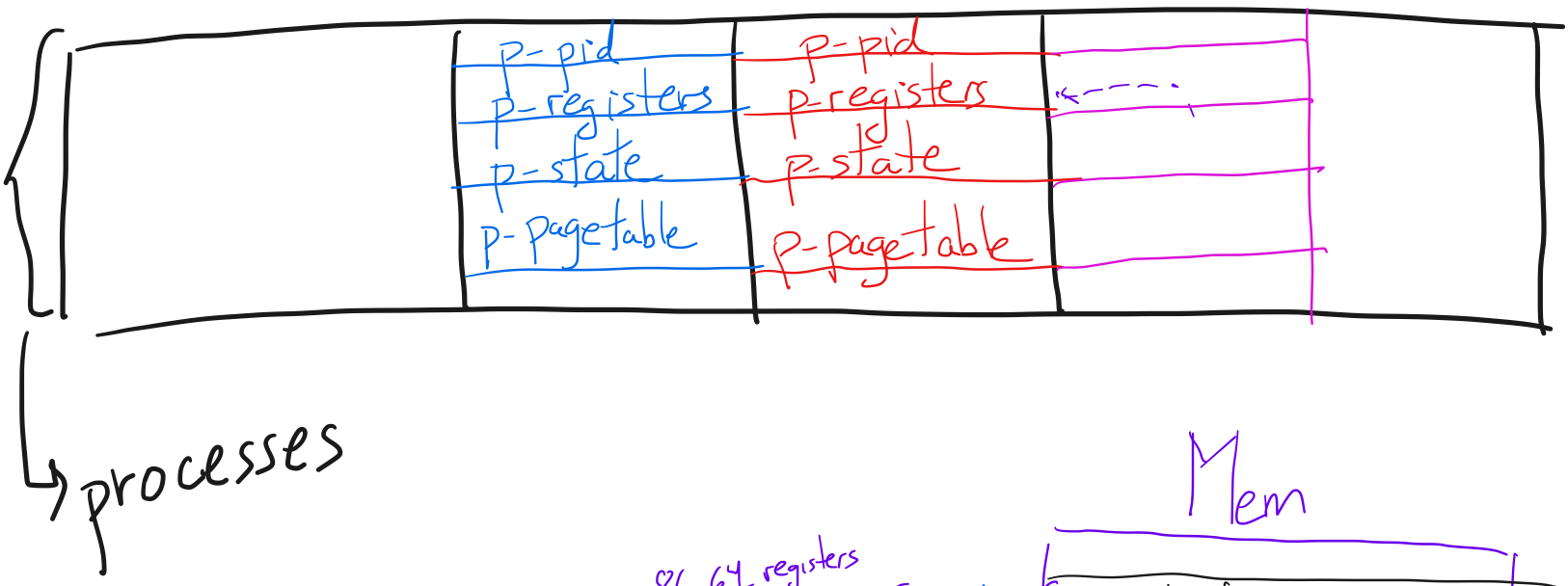
```
struct physical_pageinfo {
    int8_t owner;
    int8_t refcount;
}
```

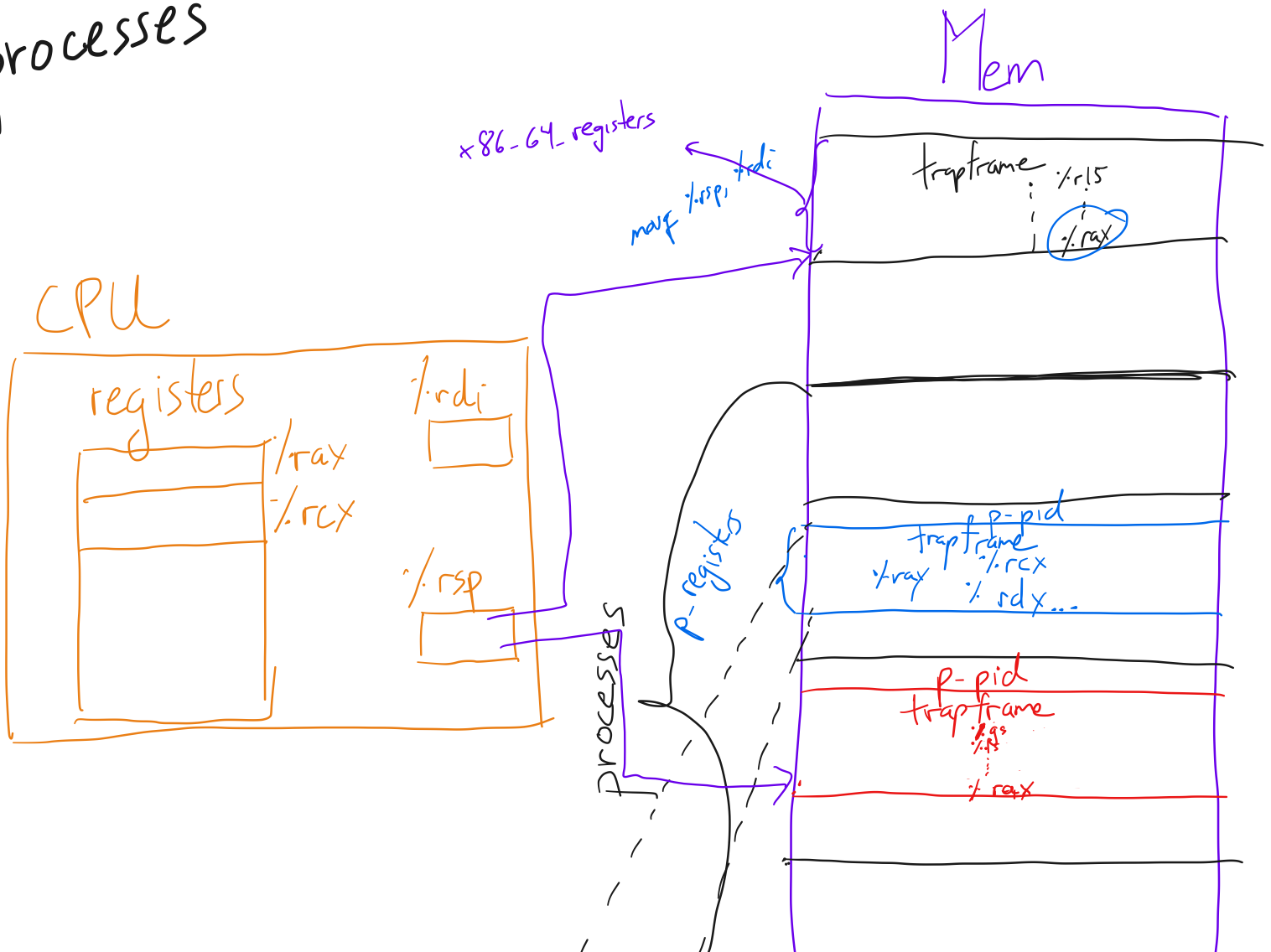


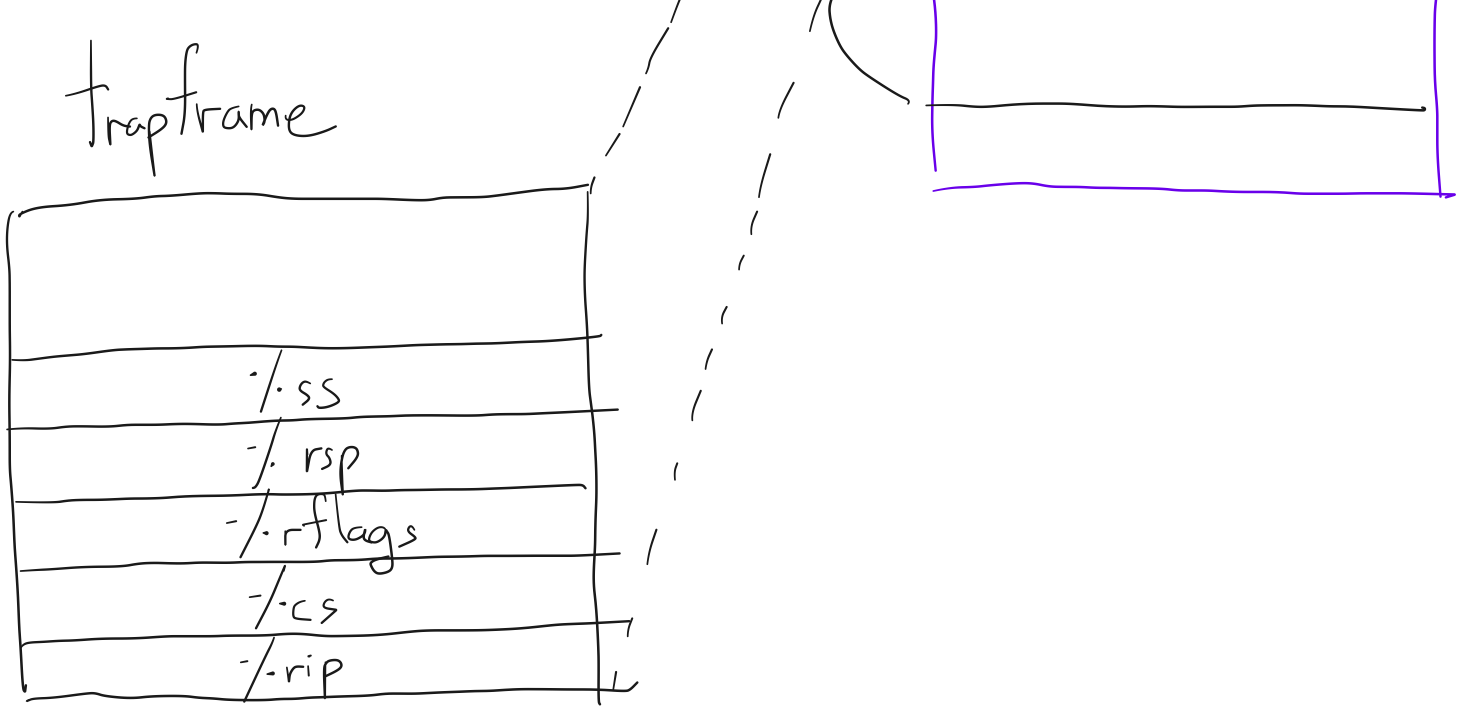
one per physical page

this is not a page table; it is bookkeeping.

Context switches in Weensy OS






trapframe
-/.ss
-/. rsp
-/.rflags
-/.cs
-/.rip