Processes Continued

- Context Switch ✓
  - what state to save + how

- Scheduling
  - i.e., when to context switch
  - preemptive, cooperative

- Interrupts, Exceptions
  - system calls ★
LAST LECTURE

What state is needed to restore "virtual" machine?

Machine Abstraction
- Memory
  - virtual memory
- Processor
- Disk (global / shared)
- Network
- Isolated Machine
  + time slicing (virtual / physical)
Process State \( \langle \text{proc}, \text{task_struct}, \text{PCB} \rangle \)

1) for virtual memory?
   - pointer to page table
   - flush TLB (tags)

2) for file system?
   - file table
   - cwd, Dir

3) for time slicing?
   - trap frame
   - registers: data + other CPU context

4) Kernel stack

Box \( \langle \text{Page} \rangle \)

Vec \( \langle \text{FileInfo} \rangle \) or

\[ \text{FileInfo} ; \text{MAX FILES} \]

TrapFrame \( \overset{\text{ready, waiting, \& zombie}}{\rightarrow} \)

- scheduler status

\( \# \) metadata (pid, parent)
When does a context switch occur?
A: When it's time to switch processes!
Q: How?
(if preemptive, then usually)

**Interrupts!**
(= exceptions)

(... otherwise on yield())
(... but also on blocking calls)
Interrupts & Exceptions

- Vector 0
- Vector 1
- Vector 2
- ... Vector n

Interrupt vector table

```
  0
  1
  2
  ...
```

- Division by zero
- Page fault (COW)
- Configurable (timer)

```
fn(f(a, b, c))
```

```
iret()
```

```
"Trap frame" stack
```

Address to jump to
Information required...
The Context Switch

```
struct process {
    page_table: ..., file_table: ...,
    cwd: ..., trap_frame: ..., stack: ..., status: ...,
}
```

```
PC A
(currently running)
```

```
PC B
chosen to run next
```

A save B restore
Scheduling

Which process (in general: “thing”) do I run next?

Where do I run the next process? thing?

- May have multiple CPUs.
- CPUs may be heterogeneous.
- May even want to schedule on not CPUs, i.e. GPU, accelerators, etc.