# NOTES: Operating Systems

# Operating Systems

- Virtualization: every program can operate as if it's the only program running
- Concurrency: multi-processor distributed systems
- Persistency: storage, filesystem

## Processes

- A process is an instanfce of a running program afforded by the abstract machine by the OS
- A process is stateful—it is defined by the program counter, all CPU registers, and all memory

### Stopping Processes

- A process can be stopped by an asynchronous interrupt—can happen at any time
- A process can be stopped by a syncronous exception, which is invoked by certain instructions
    - TRAP by specific instructions
    - FAULT by illegal instructions (i.e. divide by zero)
    - ABORT by the OS
- The OS cannot save program state after an interrupt because it needs to use registers itself. The program also should not be responsible for saving its own state because it could be interrupted at any time. Instead, the hardware (CPU) and OS collaborate to preserve the register content. Exception handling is defined by the architecture.
    - Usually, state is saved to the stack

### Scheduling

- Every so often (usually every ms), the OS sends the current program a **system timer interrupt**
- After the system timer interrupt, the OS reevaluates which process to continue

## Privilege

- CPU has various operational modes
    - User mode: normal program execution
    - Kernel mode: OS execution
- Certain instructions are only allowed in higher privileged modes
- There are special instructions for mode transition
- Memory is segmented into user and kernel space
    - OS can access all memory
    - User programs can only access their own memory

### Kernel Memory

- The kernel has its own memory and its own stack
- When the CPU changes between user/kernel mode, the stack pointer changes to the kernel or user stack

## System Calls

- A system call tells the OS to perform an action for the program
- ex. `read`, `write`, `open`, `close`, `fork`, `exec`, `wait`, `exit`
- System calls are an abstraction for the kernel, and allow the program to be portable from one OS to another