Hardware – OS & OS- Application interface
Today

- HW-OS interface
- OS-App interface
- Protection
How can an editor use a keyboard?

Diagram:
- Keyboard
  - Keyboard controller
    - Memory
      - OS
        - Keyboard driver
      - Editor
    - CPU
A modern computer system
HW-OS interface

- CPU
- Memory
  - OS
    - Device driver
  - Application
- Device controller
- Device
HW-OS interface

● Device Controller:
  ● A set of chips on a plug-in board.
  ● It has local buffer storage and/or a set of special purpose registers.
  ● Responsible for moving data between device and registers/buffer.
  ● Responsible for making data available to the device driver.
HW-OS interface

- Device Driver:
  - Belongs to the OS.
  - Communicates with the device controller.
  - Presents a uniform interface to the rest of the OS.
Driver to Controller

- Memory-mapped I/O
  - Device communication goes over the memory bus
  - Reads/Writes to special addresses are converted into I/O operations by dedicated device hardware
  - Each device appears as if it is part of the memory address space

- Programmed I/O
  - CPU has dedicated, special instructions
  - CPU has additional input/output wires (I/O bus)
  - Instruction specifies device and operation

- Memory-mapped I/O is the predominant device interfacing technique in use
Controller to Driver

- **Polling**
  - CPU constantly checks controller for new data
  - Inefficient

- **Interrupts**
  - Controller alert CPU for an event
  - Interrupt driven I/O

- **Interrupt driven I/O** enables the CPU and devices to perform tasks concurrently, increasing throughput.
Example: Reading data from disk

- Disk’s device driver (in OS) executes a read command (memory-mapped I/O).
- CPU writes the read’s descriptor to the disk’s controller register.
- CPU executes another computation.
- The disk asynchronously performs the read operation.
- When the read operation completes, by putting the requested data in disk controller’s buffer, the device controller interrupts the CPU (Interrupt driven I/O).
- The CPU stops the current computation.
- The CPU transfers the execution to the disk’s device driver (which was waiting for this read to complete).
- The disk’s device driver executes by moving the requested data from disk controller’s buffer to memory.
- On completion, the CPU resumes the interrupted computation.
- BUT, this would incur high-overhead for moving bulk-data. One interrupt per byte!
Direct Memory Access (DMA)

- Transfer data directly between device controller and memory.
- No CPU intervention required for moving bytes.
- Device raises interrupts solely when the block transfer is complete.
- Critical for high-performance devices.
OS-App interface

- **Driver to Application:**
  - Pass data from OS memory space to application memory space.

- **Application to Driver:**
  - **System Calls**
  - Like calling a routine of the OS.
  - Examples: print a character, send a packet, read a block from disk.
Protection

- OS is necessarily trusted to do anything with the hardware.
- Applications are untrusted; they should not have complete control of the hardware.
- For example, applications should not be able to:
  - access memory owned by other application, or
  - disable interrupts.
- CPU needs to distinguish whether an instruction is executed on behalf of the OS or on behalf of an application.
Dual-mode operation

- Use a *privilege mode* bit in CPU.
  - On *user mode*, CPU checks whether each instruction is allowed.
  - On *kernel mode*, CPU applies no check.
Example: changing privilege mode

System call, or
Interrupt, or
Processor exception

User Mode

Kernel Mode

Return
Synchronous VS asynchronous events...

... with respect to the execution of an application.

- **Synchronous**
  - Events triggered by the execution of the application.
  - Example: systems call, process exception (i.e. division by 0).

- **Asynchronous**
  - External events; not triggered by the application.
  - Example: intervals.

- In both cases, CPU stops executing the application, saves the execution state, and executes the corresponding handler in the OS.
Today

- HW-OS interface
- OS-App interface
- Protection
Coming up...

- Next lecture: Processes and Threads
- HW1:
  - HW-OS-App interface
  - Due on Monday, 10pm.
- No in-class exam next week.