Wednesday, March 15

IO Devices (input/output)

Table of devices and how fast the I/O hardware is

Device controllers

- IO devices have components
  - Mechanical and electronic
- Electronic component controls the device
  - Can handle multiple devices
  - Can have more than one controller per mech component (hard drive)
- Controller's tasks
  - Concert serial bit stream to block of bytes
  - Perform error correction as necessary
  - Make available to main mem

Memory-mapped IO

- Single-bus
  - All mem accesses go over a shared bus
  - IO and ram accesses compete for bandwidth
- Dual-bus
  - Ram access over high-speed bus
  - IO access over lower-speed bus
  - Less competition for bus
  - More hardware (more expensive)

Direct mem access (DMA)

Hardware view of interrupts

- Interrupt controller

IO software goals

- Device independence
  - Programs can access any IO device
  - No need to specify device in advance
- Uniform naming
  - Name file or device as string or int
  - Doesn’t depend on the machine
- Error handling
  - Done as close to hardware
  - Isolate higher-level software
- Synch vs asynch transfers
  - Blocked transfer vs interrupt-driven
- Buffering
  - Data coming off device cant be stored in final destination
- Sharable vs dedicated devices

Programmed io

- Simplest, lots of direct control, but slow
- Printing a page

Interrupt driven io

- Run by system call
- Run by interrupt

Layers of io software

- User software and libraries
- Operating system - kernel
    - device-independent OS code
    - Device drivers
    - Interrupt handlers
- Hardware

Interrupt handlers

- Interrupt handlers are best hidden
    - Drivers starts an IO operation and blocks
    - Interrupt notifies of completion
- Interrupt procedure does its task
    - Then unblocks driver that started it
    - Perform minimal actions at interrupt time
- Interrupt handler must
    - Save registers not already saved
    - Set up context for interrupt service procedure

What happens on an interrupt

- Set up stack
- Ack interrupt controller, enable interrupts
- Copy registers
- Run service procedures
- Pick new process for next
- Set up MMU context for process to run next
- Load new process registers
- Start new process

Device drivers

- Go between device controllers and rest of OS
- Standardized interface
- Drivers communicate with device and operating system
- Provides abstraction

Device independent IO software

- Provides common library routines for IO software
- Helps maintain standard appearance
- Uniform interface
- Common resource pool

Non-standard device

- Diff interface for each driver
- High OS complexity
- Less code reuse

Standard device

- Less OS/driver interface code

- Lower OS complexity
- East to ass new drivers

Buffering device input

- User space
- kernel , copy to user space
- Double buffer in kernel

IO request: what and where

- See slide 19

Disk drive structure - slide 20

- Data stored on surfaces
- Data on concentric tracks
- Data read and written by heads

Disk addressing

- Sequential numbering

Bad blocks

Calculating parity

Disk scheduling

- Use disk hardware efficiently

Clock hardware