CSE 4/521 Introduction to Operating Systems

Lecture 25 – I/O Systems II

(Transforming I/O Requests to Hardware Operations, Performance)

Summer 2018

Overview

Objective:

- Discuss how requests are handled in the hardware.
- Provide details of the performance aspects of I/O hardware and software

- Transforming I/O Requests to Hardware Operations
- Performance

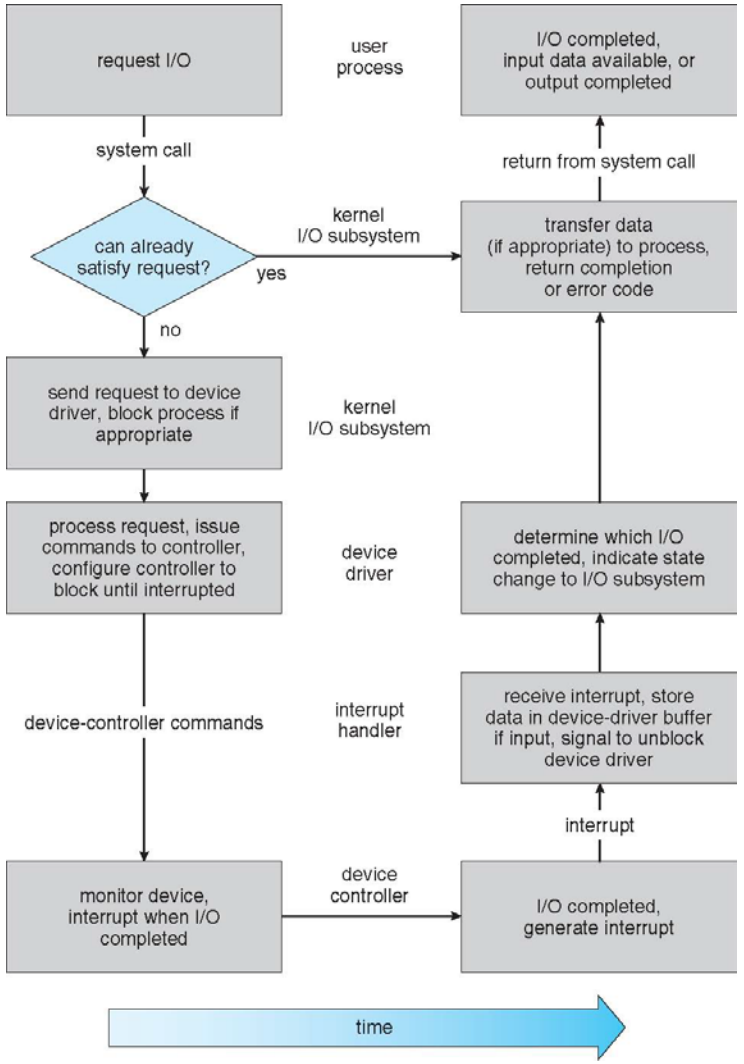
Overview

- Transforming I/O Requests to Hardware Operations
- Performance

Transforming I/O Requests to Hardware Operations

- Consider reading a file from disk for a process:
  1. Determine device holding file
  2. Translate name to device representation
  3. Physically read data from disk into buffer
  4. Make data available to requesting process
  5. Return control to process

Transforming I/O Requests to Hardware Operations



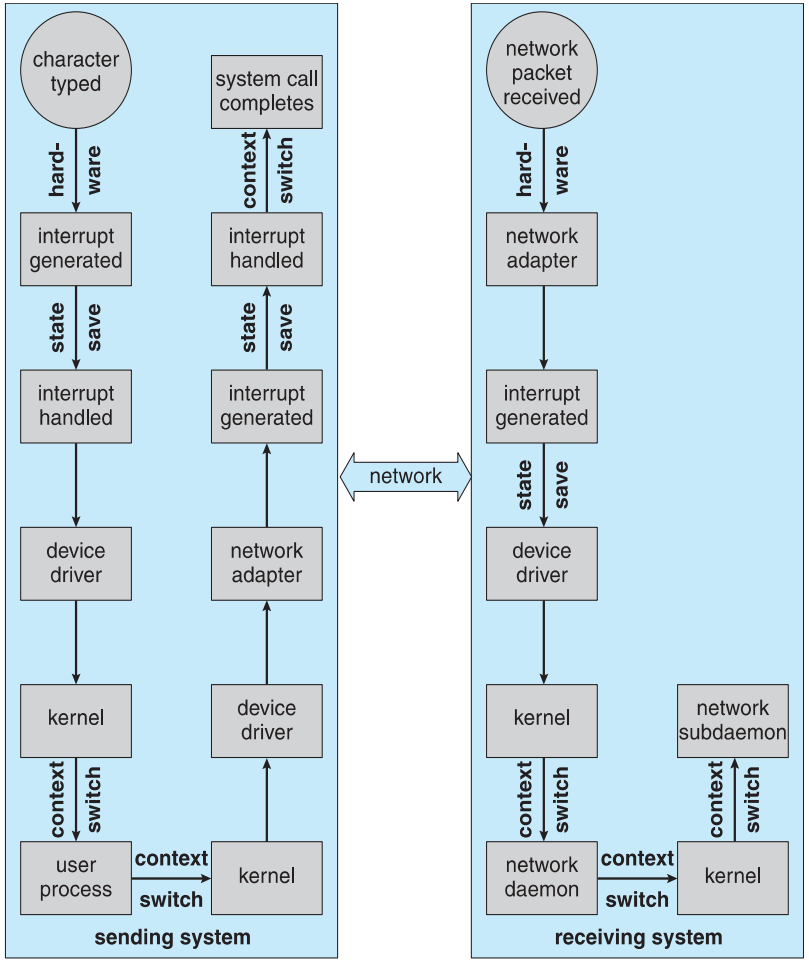
Overview

- Transforming I/O Requests to Hardware Operations
- Performance

Performance

- I/O a major factor in system performance:
  - Demands CPU to execute device driver, kernel I/O code
  - Context switches due to interrupts
  - Data copying
  - Network traffic especially stressful

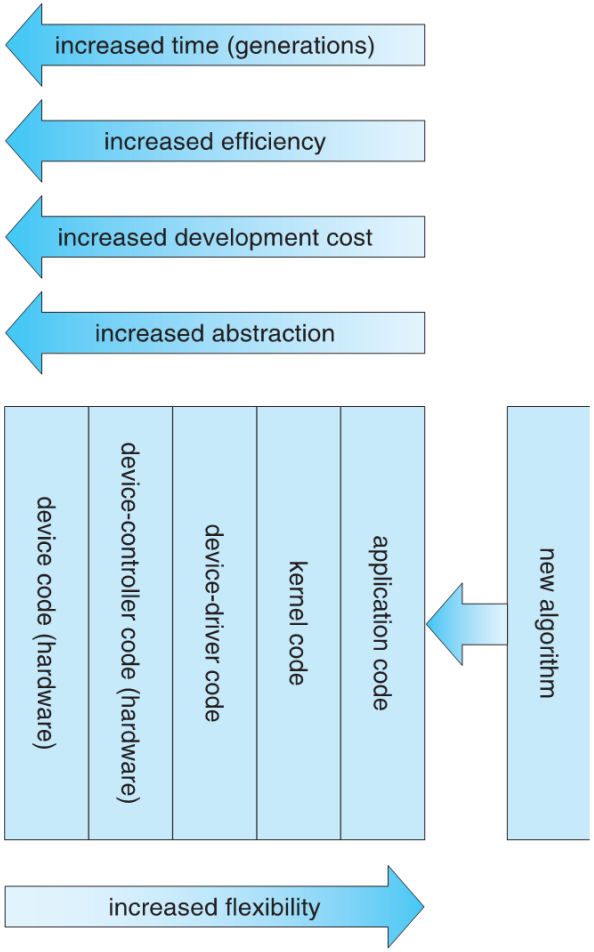
Performance: Inter-process Communication



Performance: Improving Performance

- Reduce number of context switches
- Reduce data copying
- Reduce interrupts by using large transfers, smart controllers, polling
- Use DMA
- Use smarter hardware devices
- Balance CPU, memory, bus, and I/O performance for highest throughput
- Move user-mode processes / daemons to kernel threads

Performance: Device-Functionality Progression



Performance: Power Management

- Not strictly domain of I/O, but much is I/O related
- Computers and devices use electricity, generate heat, frequently require cooling
- OSes can help manage and improve use
  - Cloud computing environments move virtual machines between servers
    - Can end up evacuating whole systems and shutting them down
- Mobile computing has power management as first class OS aspect

Performance: Power Management

- Android implements
  - Component-level power management
    - Understands relationship between components
    - Build device tree representing physical device topology
    - System bus -> I/O subsystem -> {flash, USB storage}
    - Device driver tracks state of device, whether in use
    - Unused component – turn it off
    - All devices in tree branch unused – turn off branch
  - Wake locks – like other locks but prevent sleep of device when lock is held
  - Power collapse – put a device into very deep sleep
    - Marginal power use
    - Only awake enough to respond to external stimuli (button press, incoming call)

Credits for slides

Silberschatz, Galvin and Gagne