

CSE 120: Principles of Operating Systems

Lecture 11

I/O System Software

November 18, 2003

Prof. Joe Pasquale

Department of Computer Science and Engineering
University of California, San Diego

© 2003 by Joseph Pasquale

1

Before We Begin ...

Read Chapter 11 (on I/O Systems)

2

What is I/O

Input/output between CPU/memory and I/O devices

Problems

- So many different types of I/O devices
- Wide range in speed, operation, data transfer units

Questions

- How does a process initiate I/O?
- How is synchronization achieved?
- How is data transferred?

3

Background on I/O Hardware

I/O bus interconnects CPU, memory, I/O devs

A device generally connects to the bus via a controller

Controller has registers that CPU can read/write via

- I/O instructions
- memory instructions (memory-mapped)

Data transfer: programmed I/O vs. DMA

Synchronization: polling vs. interrupts

4

Classifying Devices

So many different types of devices

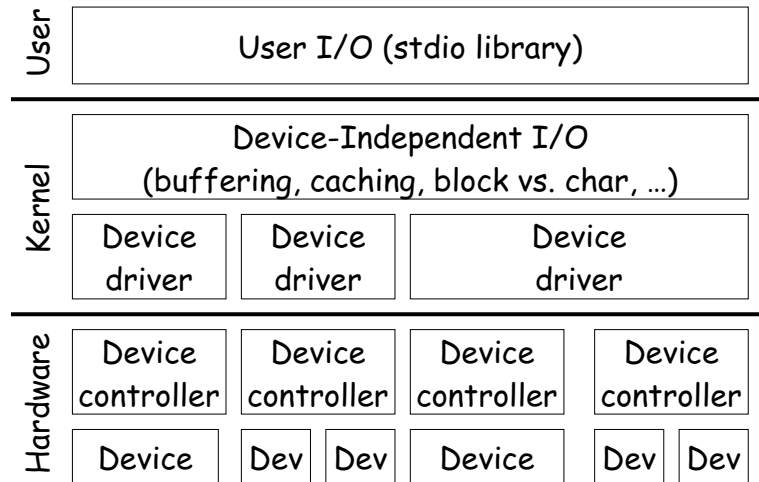
- classify by shared characteristics
- imposes structure: shared code, lower complexity

Dimensions

- character stream vs. block
- sequential vs. random-access
- synchronous vs. asynchronous
- sharable vs. dedicated
- speed of operation

5

I/O Layered Software Structure



6

Device Drivers

Encapsulates device-dependent code

Generally must implement a standard interface

Code contains device-specific register reads/writes

Interrupt handlers

- Handler executes when I/O operation completes
- May be converted to a "software interrupt"
- Updates data structures
- Wakes up process that was waiting

7

Device-Independent I/O Software

Uniform interfacing for device drivers

Naming, protection

Buffering, caching

Device-independent block size

Storage allocation for block devices

Locking (dedicated devices)

Error handling

8

User-Space I/O Software

Richer or simplified interface

- e.g., putchar/getchar vs. read/write

User-level buffering

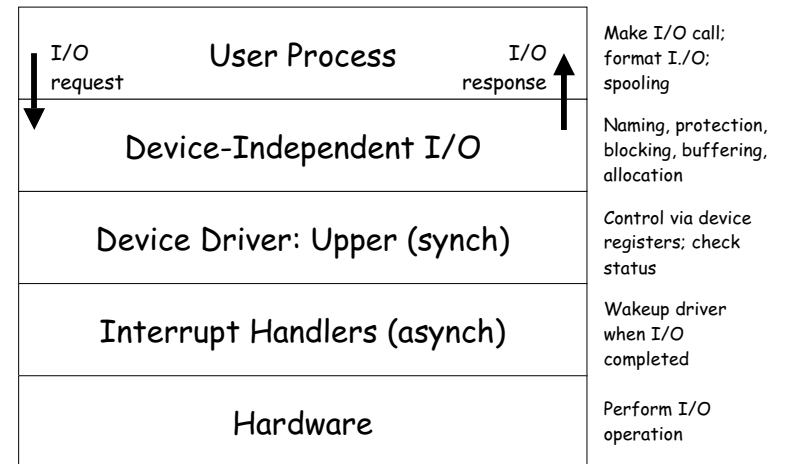
- e.g. stdio library

Spooling daemons

- Printer

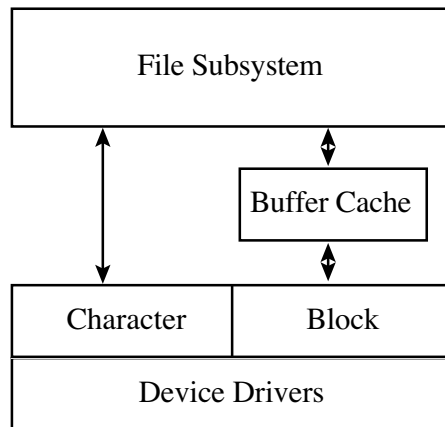
9

Overall Operation



10

Example: Unix I/O



11

Unix: I/O System Calls

Via file system interface

- open
- close
- read
- write
- ioctl

12

Unix: Buffered vs. Unbuffered I/O

Buffered I/O

- system buffers: buffer cache, character queues

Unbuffered I/O

- data transferred directly between proc and dev
- can be via DMA

13

Unix: Standard I/O Library

fopen, fread, fwrite, fprintf, fscanf, fclose

Private buffer kept in user space

Minimizes the number of I/O system calls

14

Unix: Block vs. Character Devices

Block

- transfer units in fixed-size blocks
- blocks are addressable (random access)
- kept in buffer cache

Character

- transfer units in variable-size sequence of bytes
- used for all non-block devices
- linked list of character queues

15

Unix: The Buffer Cache

Cache of recently used blocks

Each block has an address (device #, block #)

Three lists

- device list (hash table + linked lists)
- free list
- driver I/O queue

LRU is used for replacement

16