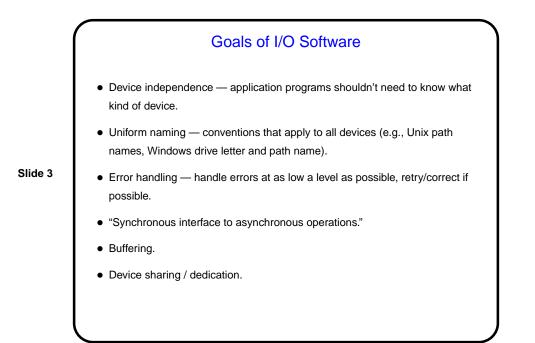
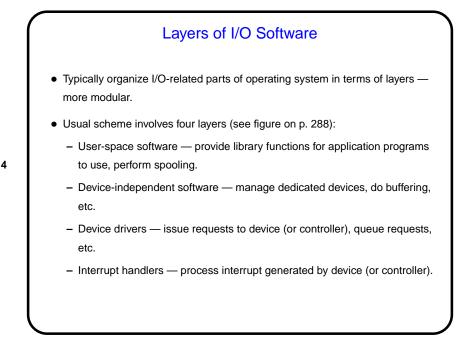
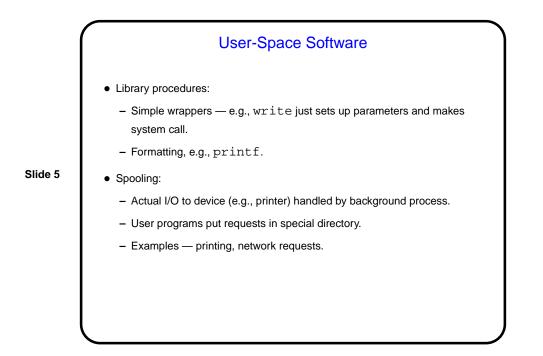
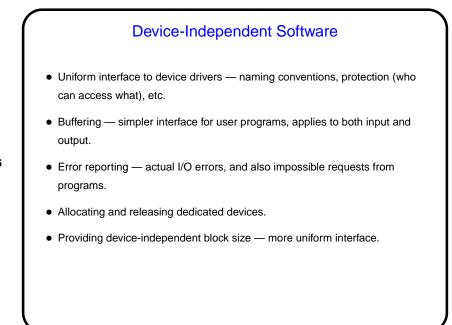


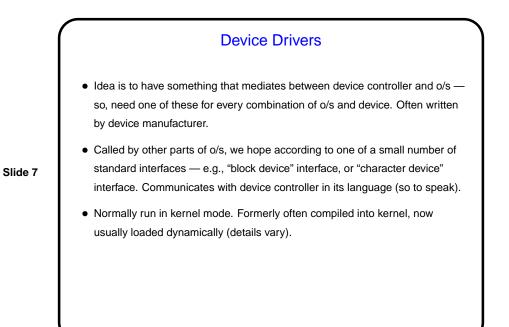
I/O Subsystem, Continued Goal of this part of the o/s is like for other parts — manage physical resources on behalf of (all) user programs, and provide a nicer "virtual machine". Hardware is diverse and complicated, so in discussing it we try to find basic principles, talk about what's available to o/s-level programmer.







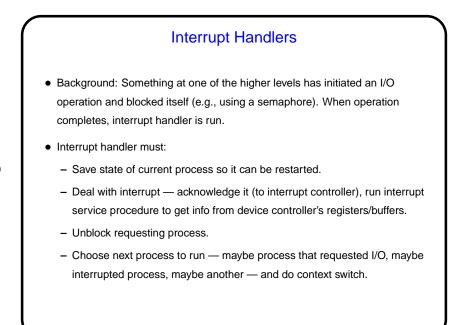


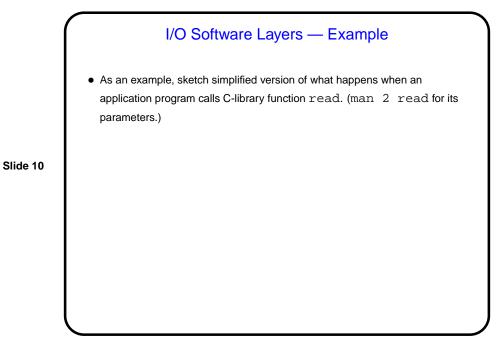


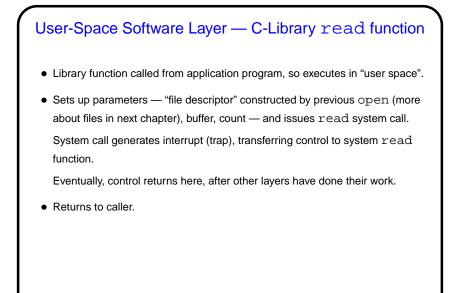
Device Drivers, Continued
When called, must:

Check that parameters are okay (return if not).
Check that device is not in use (queue request if it is).
Talk to device — may involve many commands, may require waiting (block if so).

Check for errors, return info to caller. If there are queued requests, continue with next one.

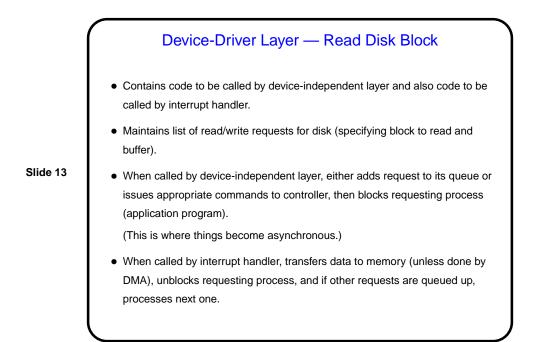






Device-Independent Software Layer — System read Function Invoked by interrupt handler for system calls, so executes in kernel mode. Checks parameters — is the file descriptor okay (not null, open for reading, etc.)? Returns error code if necessary. If buffering, checks to account the request can be obtained from buffer. If account of the second seco

- Slide 12
- If buffering, checks to see whether request can be obtained from buffer. If so, copies data and returns.
- If no buffering, or not enough data in buffer, calls appropriate device driver (file descriptor indicates which one to call, other parameters such as block number) to fill buffer, then copies data and returns.



Interrupt-Handler Layer — Read Disk Block

- Gets control when requested disk operation finishes and generates interrupt.
- Gets status and data from disk controller, unblocks waiting user process.
 At this point, "call stack" (for user process) contains C library function, system read function, and a device-driver function. We return to the device-driver function and then unwind the stack.

