

Input/Output Subsystem

System Programming
2019-2020 Fall

Input-Output (I/O)

- operating system must control all I/O devices
 - issue commands to devices
 - catch interrupts
 - handle errors
 - provide interface between devices and rest of system

I/O Devices

- main categories
 - block devices
 - character devices
 - network devices
 - clocks and timers

I/O Devices – Block Devices

- block devices
 - fixed sized blocks
 - each block has its own address
 - possible to read/write each block independently
 - can host a file system
 - e.g. disks

I/O Devices – Character Devices

- character devices
 - stream of characters
 - no block structure
 - can transfer arbitrary sized data in single I/O operation
 - not adressable
 - no seek operation
 - e.g. terminals, mice, sound cards, serial / parallel ports, ...

I/O Devices

- I/O units typically consist of
 - a mechanical component
 - an electronic component
 - device controller / adapter
- operating system deals with controller
 - connected over a standard interface

Device Controllers

- controllers have registers to communicate with CPU
 - control register
 - send command to device
 - status register
 - read state of device
 - input / output register

Memory Mapped I/O

- registers part of regular memory address space
 - e.g. 680x0 family
 - directly mapped
 - preserve part of memory address space for I/O locations
 - disable virtual memory management
 - not frequently used
 - software mapped
 - virtual memory management available

I/O Ports

- use a special address space for I/O
 - controllers have I/O addresses and interrupt vectors
 - separate read/write lines for I/O ports
 - special instructions

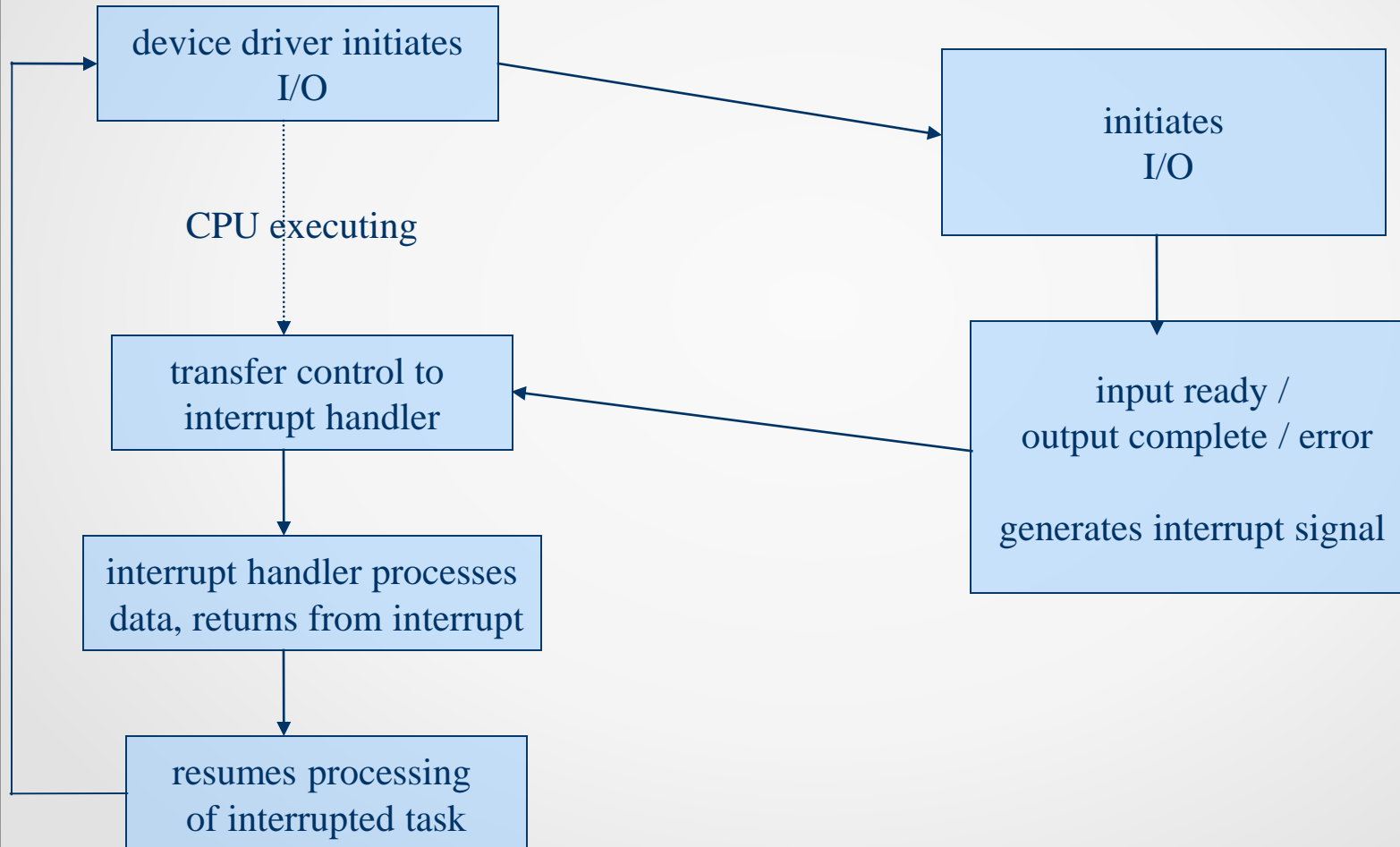
I/O Devices

- protocol for interaction between host and controller
 - polling (via handshaking)
 - uses controller status register
 - interrupt driven

Interrupt-Driven I/O Cycle

CPU

I/O Controller



Device Controllers

- for I/O
 - CPU
 - writes command into controller registers (with parameters)
 - continues with other work
 - controller
 - accepts and executes command
 - causes an interrupt on conclusion
 - CPU
 - gets results and device status from controller registers

Direct Memory Access (DMA)

- many controllers support DMA
 - especially for block devices
- a DMA controller is used
- handshaking between DMA controller and device controller

Disk Read Operation without DMA (Programmed I/O - PIO)

- device controller
 - reads from disk serially until block completed
 - into controller's internal buffer
 - verifies no errors
 - causes interrupt

operating system

reads byte / word from
controller's buffer

stores into memory

repeats until completed

Wastes CPU time !

Disk Read Operation with DMA

- CPU

- passes extra information to controller
 - disk block address
 - memory address to store block
 - number of bytes to transfer

DMA controller

device controller reads from disk serially

DMA controller copies data from buffer to memory

no CPU intervention

I/O Software

- concepts
 - abstraction: standardized interface
 - encapsulation: device drivers
 - layering
- organized as a series of layers
 - lower layers hide the hardware specific operations
 - higher layers provide easy-to-use, regular interface to users

Aspects of I/O Software Design

- device independence
- uniform naming
 - name of a file or device
- error handling
 - generally should be done closer to hardware if possible

Aspects of I/O Software Design

- blocking x interrupt driven transfers
 - better for CPU to do interrupt driven transfers
 - easier for user programs to use blocking I/O operations
- ⇒ operating system makes interrupt-driven operations look blocking to users

Aspects of I/O Software Design

- shared x dedicated devices
 - e.g. disks x printers
- ⇒ operating system handles the devices accordingly

Kernel I/O Subsystem

- services provided
 - I/O scheduling
 - order in which they are issued may not be the best order to execute them
 - requests are queued
 - scheduling re-arranges order in queue
 - improves efficiency

Kernel I/O Subsystem

(services provided cntd.)

- buffering
 - to cope with speed mismatch
 - e.g. receive file through modem to store on disk
 - to adapt between devices that have different data-transfer sizes
 - e.g. network packets
 - to support copy-semantics for application I/O

Kernel I/O Subsystem

(services provided cntd.)

- caching
 - provides faster access
- error handling
- spooling and device reservation system

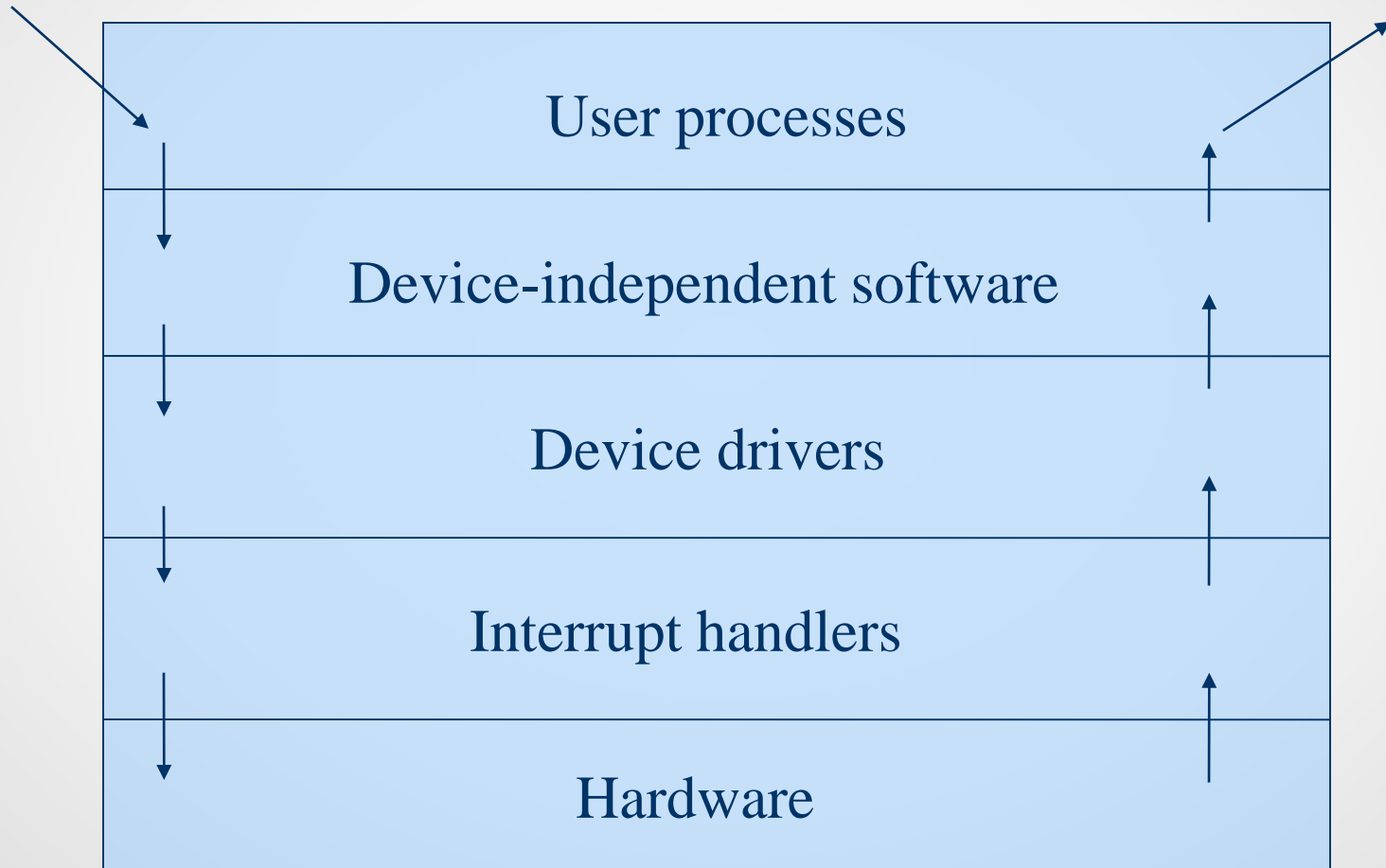
Structure of an I/O Software

- organized as 4 layers:
 - interrupt handlers
 - device drivers
 - device independent operating system software
 - user level software

I/O System

I/O Request

I/O Reply



Interrupt Handlers

- interrupts hidden from rest of system
- I/O requesting process blocks until request completed
- when I/O is completed, interrupt occurs
- process is made to unblock

Device Drivers

- device dependent code
- a driver for each device type
- e.g. for a disk, driver knows
 - controller registers
 - disk info (sectors, tracks, cylinders, ...)

Device Drivers

- accepts abstract requests from device-independent software
- translates request
 - decides on sequence of controller operations
 - e.g. for a disk driver
 - finds block on actual disk
 - checks drive's motor
 - positions disk arm...

Device Drivers

- issues commands to controller
- blocks until operation completed
- unblocks on interrupt
- checks for errors
- passes required info to device independent software
- returns status info to caller
- ready for next request

Device - Independent I/O Software

- performs I/O functions common to all devices
- provides uniform interface to user-level software

Functions of the Device - Independent I/O Software

- uniform interfacing for device drivers
- device naming
- device protection
- provide device independent block sizes

buffering

allocating and releasing
dedicated devices

error reporting

User-Space I/O Software

- a small part of I/O software provided as libraries
 - system calls are made by library procedures
e.g. `printf`
 - takes format string and parameters as input
 - builds an ASCII string
 - calls `WRITE` to output string
- library procedures run as part of user programs

Summary: Life cycle of an I/O Request

