I/O Systems

Dr. Jun Zheng

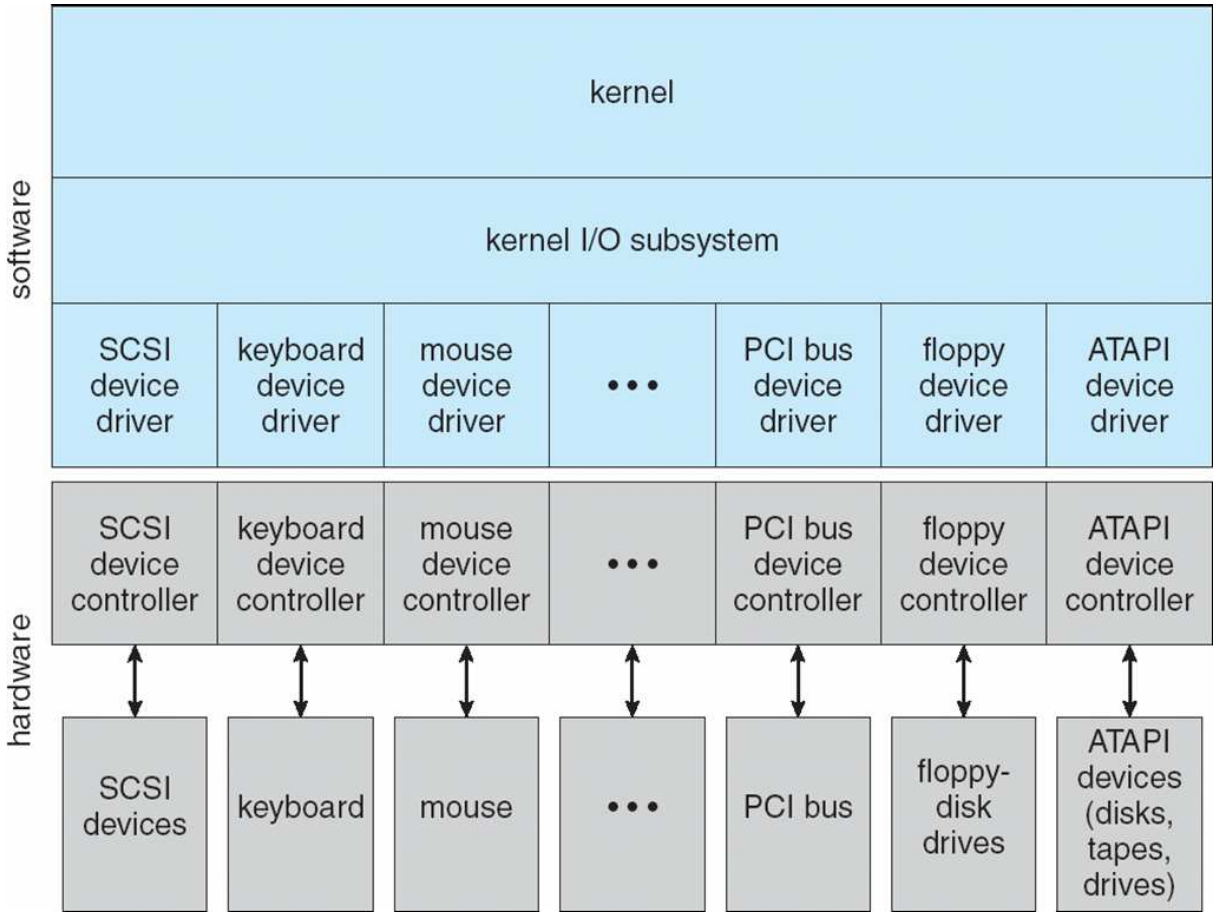
CSE325 Principles of Operating Systems

12/4/2019

Architecture of I/O Systems

- Key components
  - **System bus:** allows the device to communicate with the CPU, typically shared by multiple devices.
  - A device **port** typically consisting of 4 registers:
    - **Status** indicates a device busy, data ready, or error condition
    - **Control:** command to perform
    - **Data-in:** data being sent from the device to the CPU
    - **Data-out:** data being sent from the CPU to the device
  - **Controller:** receives commands from the system bus, translates them into device actions, and reads/writes data onto the system bus.
  - The device itself
- Traditional devices: disk drive, printer, keyboard, modem, mouse, display
- Non-traditional devices: joystick, robot actuators, flying surfaces of an airplane, fuel injection system of a car, …

Kernel I/O Subsystem



Memory Mapped I/O

- Memory mapped I/O is a popular way for the device controller to do the I/O transfer.
- The CPU executes I/O requests using the standard data transfer instructions to read and write the device-control registers at their mapped locations in physical memory.

I/O address range (hexadecimal)	device
000–00F	DMA controller
020–021	interrupt controller
040–043	timer
200–20F	game controller
2F8–2FF	serial port (secondary)
320–32F	hard-disk controller
378–37F	parallel port
3D0–3DF	graphics controller
3F0–3F7	diskette-drive controller
3F8–3FF	serial port (primary)

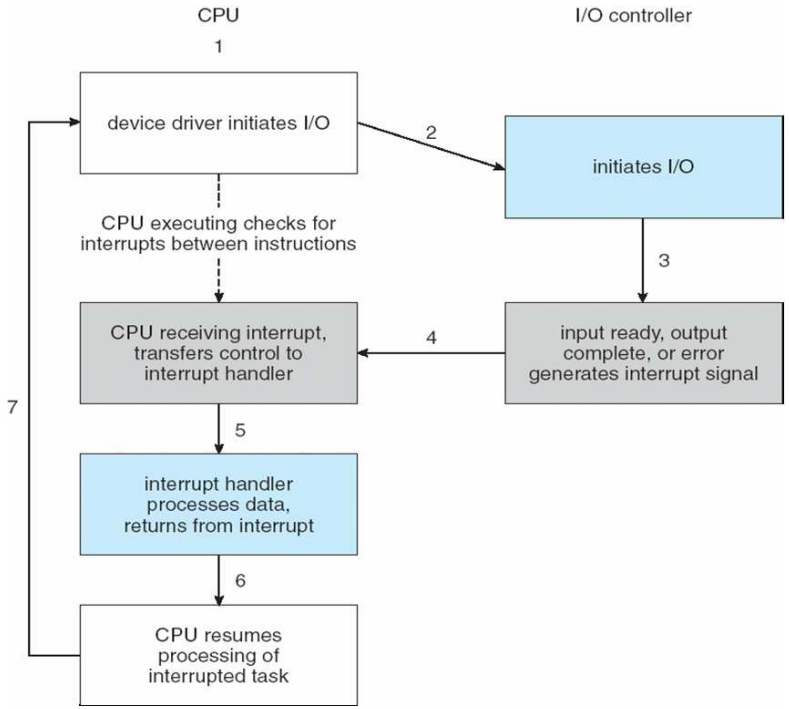
Device I/O port locations on PCs (partial)

Polling

- CPU busy-waits until the status of the controller is idle.
- CPU sets the command register and data-out if it is an output operation.
- CPU sets status to command-ready => controller sets status to busy.
- Controller reads the command register and performs the command, placing a value in data-in if it is an input command.
- If the operation succeeds, the controller changes the status to idle.
- CPU observes the change to idle and reads the data if it was an input operation.
- Good choice if data must be handled promptly, like for a modem or keyboard.
- What happens if the device is slow compared to the CPU?

Interrupts

- Rather than using busy waiting, the device can interrupt the CPU when it completes an I/O operation.
- On an I/O interrupt:
  - Determine which device caused the interrupt.
  - If the last command was an input operation, retrieve the data from the device register.
  - Start the next operation for that device.



Intel X86 Event-Vectors

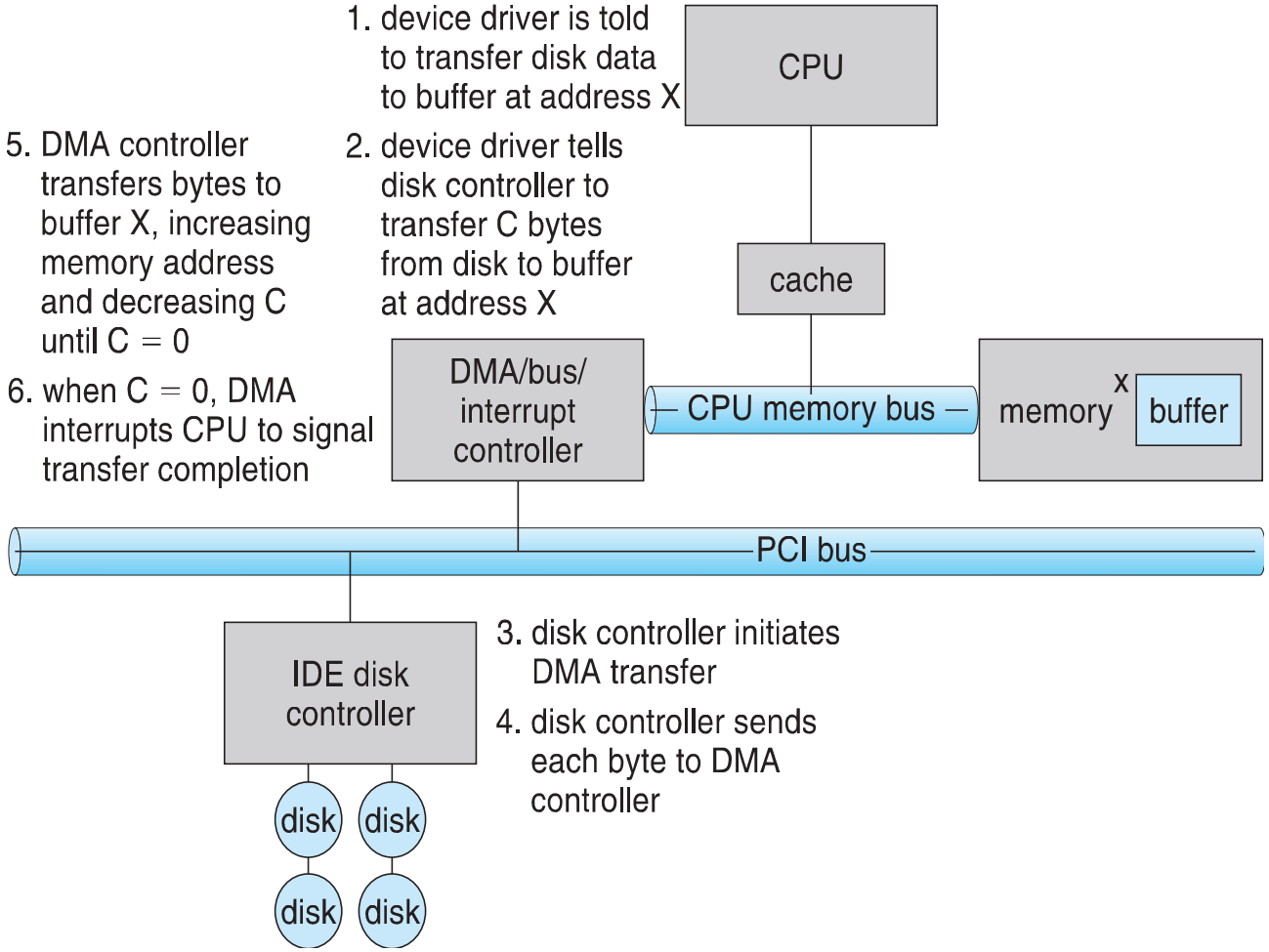
- Unmaskable interrupt: reserved for events such as unrecoverable memory errors
- Maskable: it can be turned off by the CPU before the execution of critical instruction sequences that must not be interrupted.

vector number	description
0	divide error
1	debug exception
2	null interrupt
3	breakpoint
4	INTO-detected overflow
5	bound range exception
6	invalid opcode
7	device not available
8	double fault
9	coprocessor segment overrun (reserved)
10	invalid task state segment
11	segment not present
12	stack fault
13	general protection
14	page fault
15	(Intel reserved, do not use)
16	floating-point error
17	alignment check
18	machine check
19–31	(Intel reserved, do not use)
32–255	maskable interrupts

Direct Memory Access

- For devices that transfer large volumes of data at a time (like a disk block), it is expensive to have the CPU retrieve these one byte at a time.
- **Solution:** Direct memory access (DMA)
  - Use a sophisticated DMA controller that can write directly to memory. Instead of data-in/data-out registers, it has an address register.
  - The CPU tells the DMA the locations of the source and destination of the transfer.
  - The DMA controller operates the bus and interrupts the CPU when the entire transfer is complete, instead of when each byte is ready.
  - The DMA controller and the CPU compete for the memory bus, slowing down the CPU somewhat, but still providing better performance than if the CPU had to do the transfer itself.

Six Step Process to Perform DMA Transfer



Examples of I/O Device Types

aspect	variation	example
data-transfer mode	character block	terminal disk
access method	sequential random	modem CD-ROM
transfer schedule	synchronous asynchronous	tape keyboard
sharing	dedicated sharable	tape keyboard
device speed	latency seek time transfer rate delay between operations	
I/O direction	read only write only read–write	CD-ROM graphics controller disk