東南大學
SOUTHEAST UNIVERSITY

OPERATING SYSTEM CONCEPTS

Chapter 13. I/O Systems

A/Prof. Kai Dong

Contents

A Canonical Device

- A device has two important components.
  - The first is the hardware interface it presents to the rest of the system
  - The second part of any device is its internal structure, which is implementation specific.

Registers	Status	Command	Data	**Interface**
Micro-controller (CPU) Memory (DRAM or SRAM or both) Other Hardware-specific Chips				**Internals**

- Three registers:
  - Status — to be read to see the current status of the device;
  - Command — to tell the device to perform a certain task;
  - Data — to pass data to , or get data from the device.

A Canonical Protocol

- Polling is inefficient

```
While (STATUS == BUSY)
        ;       // wait until device is not busy
Write data to DATA register
Write command to COMMAND register
        (Doing so starts the device and executes the command)
While (STATUS == BUSY)
        ;       // wait until device is done with your request
```

Interrupts

- Interrupt-driven
- Instead of polling the device repeatedly, the OS can issue a request, put the calling process to sleep, and context switch to another task. When the device is finally finished with the operation, it will raise a hardware interrupt, causing the CPU to jump into the OS at a pre-determined interrupt service routine (ISR) or more simply an interrupt handler.
  - Without interrupts

CPU	1	1	1	1	1	p	p	p	p	p	1	1	1	1	1
Disk						1	1	1	1	1					

  - Interrupt-driven

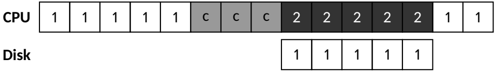
CPU	1	1	1	1	1	2	2	2	2	2	1	1	1	1	1
Disk						1	1	1	1	1					

Interrupts

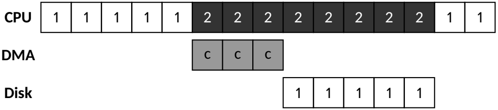
- Take the device speed into account
  - If a device is fast, it may be best to poll; if it is slow, interrupts are best.
  - If the speed of the device is not known, or sometimes fast and sometimes slow, it may be best to use a hybrid that polls for a little while and then, if the device is not yet finished, uses interrupts.
- Livelocks
  - In networks, when a huge stream of incoming packets each generate an interrupt, it is possible for the OS to livelock, that is, find itself only processing interrupts and never allowing a user-level process to run and actually service the requests.
  - Slashdot effect
  - Coalescing: multiple interrupts can be coalesced into a single interrupt delivery

Direct Memory Access

- One remaining problem:
- Suppose a transfer of a large chunk of data to a device
  - The CPU is overburdened



- Direct Memory Access (DMA).
  - A DMA engine is essentially a very specific device within a system that can orchestrate transfers between devices and main memory without much CPU intervention.



Direct Memory Access

- How to communicate with devices?
  - I/O instructions.
    - E.g., the in and out instructions in x86.
    - Such instructions are usually privileged.
  - Memory mapped I/O.
    - The hardware makes device registers available as if they were memory locations.

Device Driver

- Device driver
  - A piece of software at the lowest level in the OS which know in detail how a device works.

Contents

Application I/O Interface

- *Character-stream or block*. A character-stream device transfers bytes one by one, whereas a block device transfers a block of bytes as a unit.
- *Sequential or random access*. A sequential device transfers data in a fixed order determined by the device, whereas the user of a random-access device can instruct the device to seek to any of the available data storage locations.
- *Synchronous or asynchronous*. A synchronous device performs data transfers with predictable response times, in coordination with other aspects of the system. An asynchronous device exhibits irregular or unpredictable response times not coordinated with other computer events.
- *Sharable or dedicated*. A sharable device can be used concurrently by several processes or threads; a dedicated device cannot.
- *Speed of operation*. Device speeds range from a few bytes per second to a few gigabytes per second.
- *Read–write, read only, or write only*. Some devices perform both input and output, but others support only one data transfer direction.

Application I/O Interface

Contd.

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

Application I/O Interface

Clocks and Timers

- Most computers have hardware clocks and timers that provide three basic functions:
  - Give the current time.
  - Give the elapsed time.
  - Set a timer to trigger operation X at time T
- On many computers, the interrupt rate generated by the hardware clock is 18 $\sim$ 60 ticks per second.

Application I/O Interface

Nonblocking and Asynchronous I/O

- Asynchronous I/O
  - interfaces enable an application to issue an I/O request and return control immediately to the caller, be-fore the I/O has completed;
  - additional interfaces (IO interrupts) enable an application to determine whether various I/Os have completed.
- AIO control block

```
struct aiocb {
    int aio_fildes; // File descriptor
    off_t aio_offset;   // File offset
    volatile void*aio_buf;  // Location of buffer
    size_t aio_nbytes;  // Length of transfer
};
```

  - To issue an asynchronous read to a file, an application should first fill in this structure with the relevant information: the file descriptor of the file to be read (*aio_fildes*), the offset within the file (*aio_offset*)as well as the length of the request (*aio_nbytes*), and finally the tar-get memory location into which the results of the read should be copied (*aio_buf*).

Contents

Kernel I/O Subsystem

- I/O Scheduling
  - to schedule a set of I/O requests means to determine a good order in which to execute them.
- Buffering
  - to cope with a speed mismatch between the producer and consumer of a data stream.
  - to provide adaptations for devices that have different data-transfer sizes.
  - to support copy semantics for application I/O.
- Caching
  - a region of fast memory that holds copies of data.
- Spooling and Device Reservation
  - *Spooling*: OS intercepts all output to a device (such as a printer, that cannot accept interleaved data streams), and each application's output is spooled (buffered) to a separate disk file.
  - *Device reservation*: explicit facilities for coordination among concurrent applications and dedicated devices.