# Operating Systems W14L1 - I/O Part 1

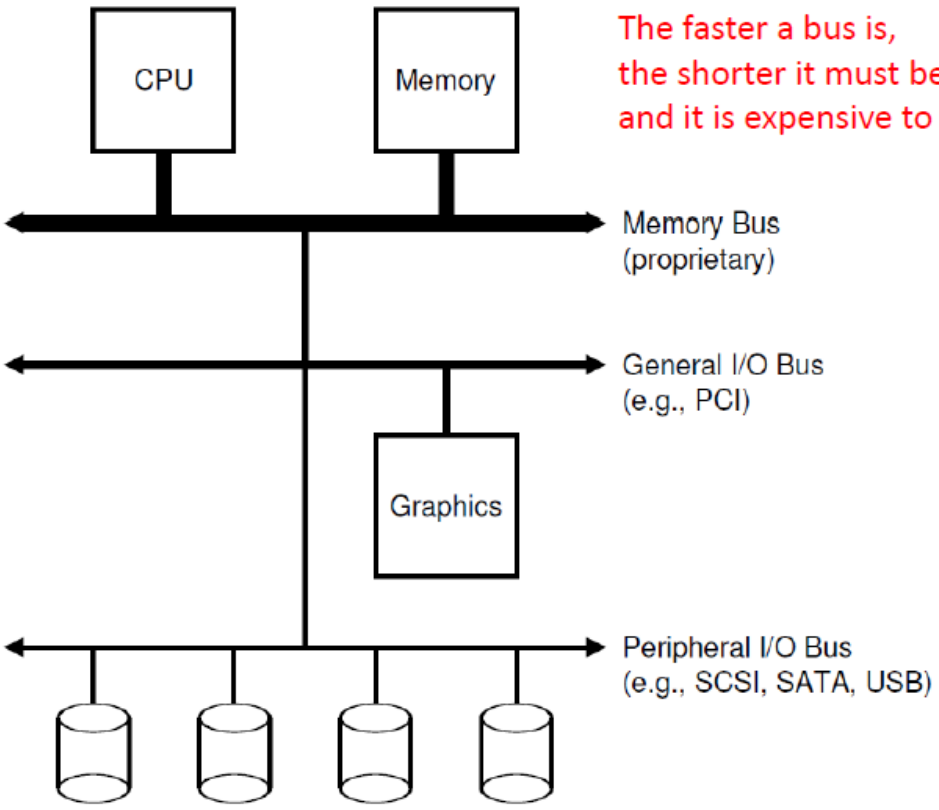
| | |
|---|---|
| Class | Operating Systems |
| Created | @Dec 4, 2020 8:56 PM |
| Materials | 11 - IO I.pdf |
| Reviewed | ☐ |
| Type | Lecture |

> ! I/O part 1 is the last part included in final exam

## Initial Discussion

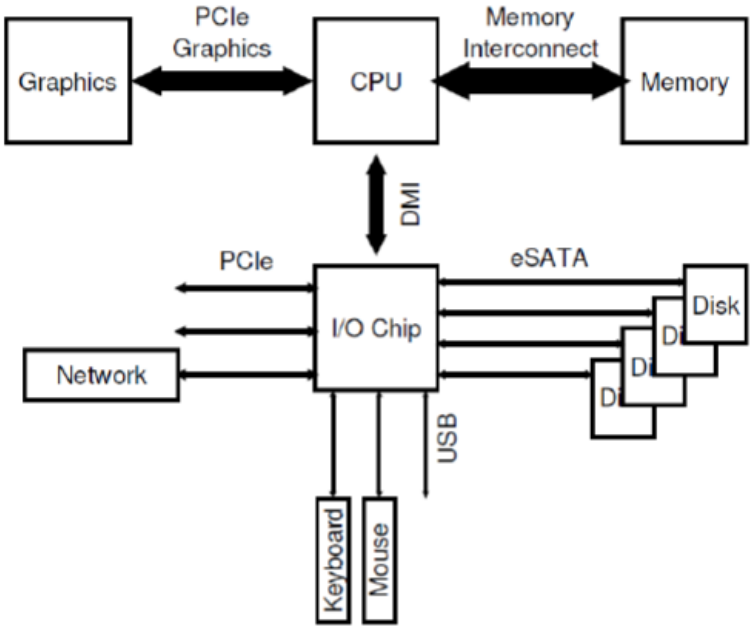
- OS must deal with different ways of controlling among the large amount of possible I/O devices

▼ Various buses for different components




- The faster a bus, the shorter it must be → Also expensive to design
- Speed will slow to match a bus

▼ Modern system architecture

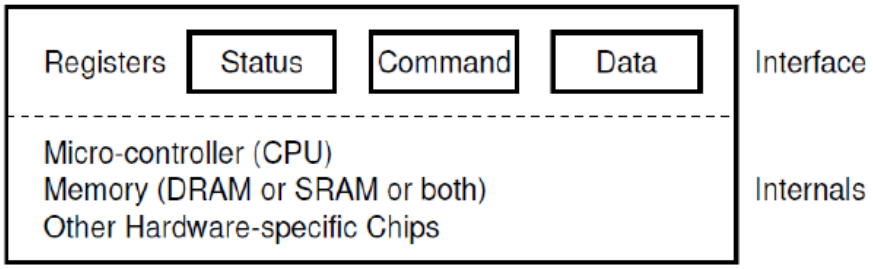



Example of one of Intel Chipsets.

DMI: Intel's proprietary Direct Media Interface.

- An I/O device has a *controller* which is what the OS talks to

▼ Generic I/O Device




The OS can deal with the above device as follows:

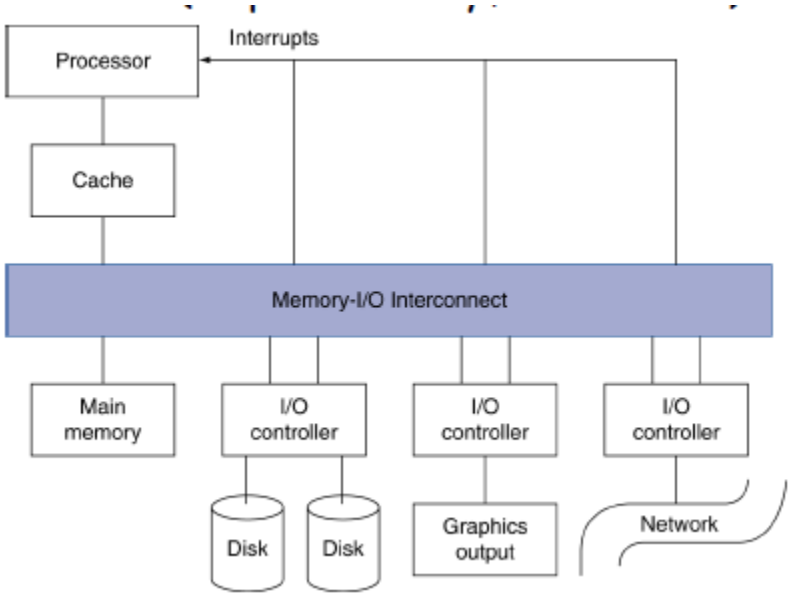
```
While (STATUS == BUSY)
    ; // wait until device is not busy
Write data to DATA register
Write command to COMMAND register
    (starts the device and executes the command)
While (STATUS == BUSY)
    ; // wait until device is done with your request
```

Polling is not efficient.

- Using while loops, or *polling*, is rather inefficient

- Different categories of I/O devices
    1. Human Readable — Suitable for communicating with the computer user (i.e. printers, keyboards)
    2. Machine Readable — Suitable for communicating with electronic equipment (i.e. USBs, controllers)
    3. Communication — Suitable for communicating with remote devices (i.e. ethernet cards, wifi adapters)
- Device driver allows OS to "learn" about new I/O devices
    - Simple definition — Move data from I/O to processor via modules and buffer
- The OS provies the interface between devices and the rest of the system
- Various challenges are presented, such as...
    - Diverse devices
    - Expandability and resilience
    - Performance
    - Verity of diffusers

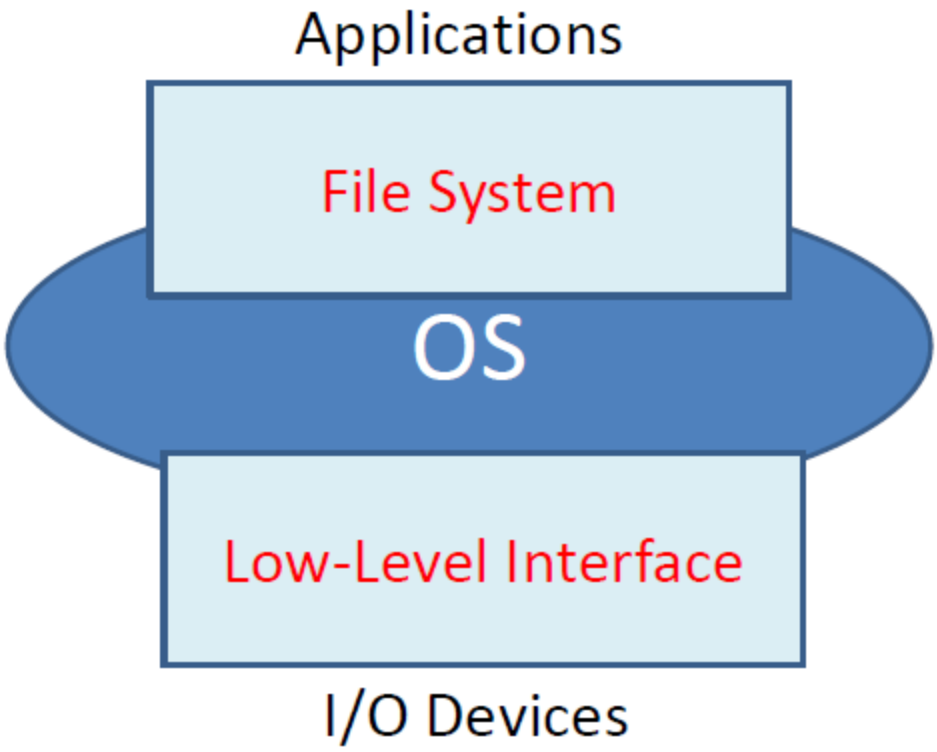
    ▼ Diagram



## Devices

- Block device
  - Stores info in fixed-size blocks
  - Transfers in blocks (each block has own address)
  - This includes things like USB sticks and hard disks
- Character device
  - Delivers (or accepts) stream of characters
  - Not addressable
  - This includes things like mice and printers

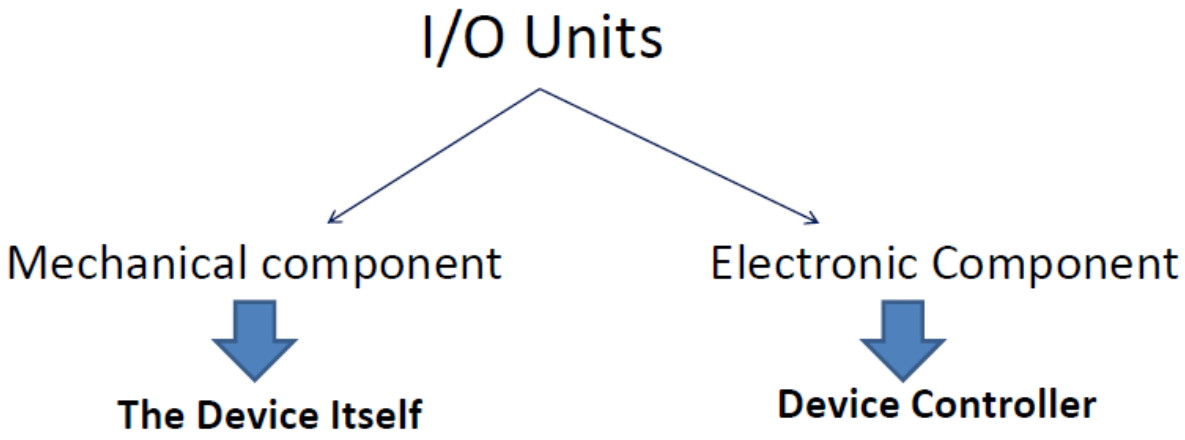
▼ Big Picture Idea

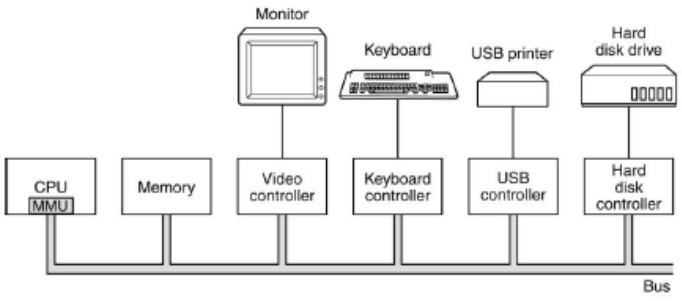



▼ Differing data rates across devices

| Device | Data rate |
|---|---|
| Keyboard | 10 bytes/sec |
| Mouse | 100 bytes/sec |
| 56K modem | 7 KB/sec |
| Scanner | 400 KB/sec |
| Digital camcorder | 3.5 MB/sec |
| 802.11g Wireless | 6.75 MB/sec |
| 52x CD-ROM | 7.8 MB/sec |
| Fast Ethernet | 12.5 MB/sec |
| Compact flash card | 40 MB/sec |
| FireWire (IEEE 1394) | 50 MB/sec |
| USB 2.0 | 60 MB/sec |
| SONET OC-12 network | 78 MB/sec |
| SCSI Ultra 2 disk | 80 MB/sec |
| Gigabit Ethernet | 125 MB/sec |
| SATA disk drive | 300 MB/sec |
| Ultrium tape | 320 MB/sec |
| PCI bus | 528 MB/sec |

▼ Mechanical component and electronic component



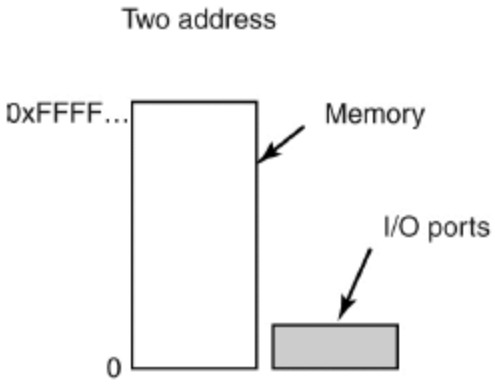

# Device Controllers

- Each controller has registers that communicate with CPU
  - Written/read into/from
- There are alos data buffers that can be read/written by OS
- Two main approaches for communication
  1. I/O port space
  2. Memory-mapped I/O
  - Both are seen in all operating systems today

## Port Space

- Each register → I/O port number
- Sets of I/O ports is I/O port space
- Port space is protected

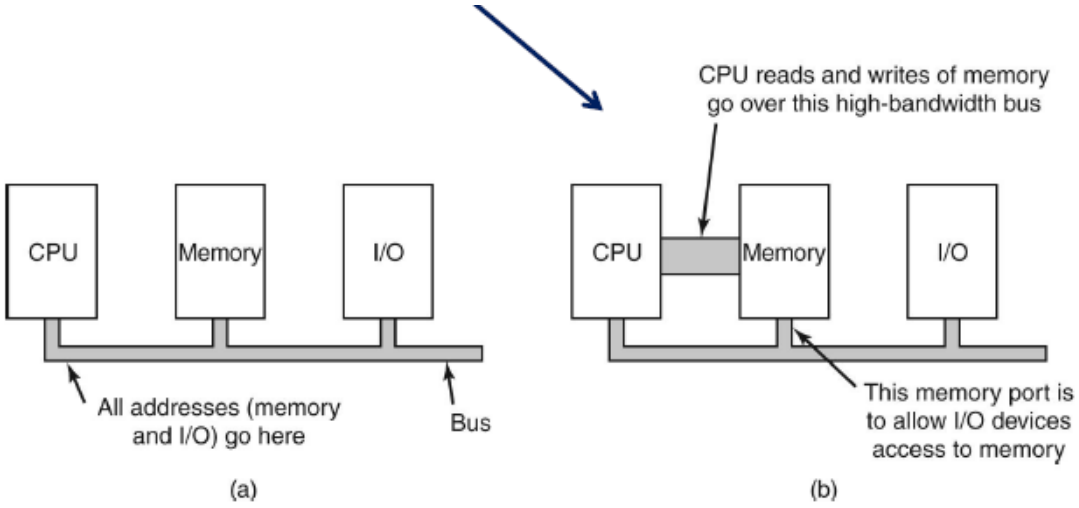
▼ Diagram



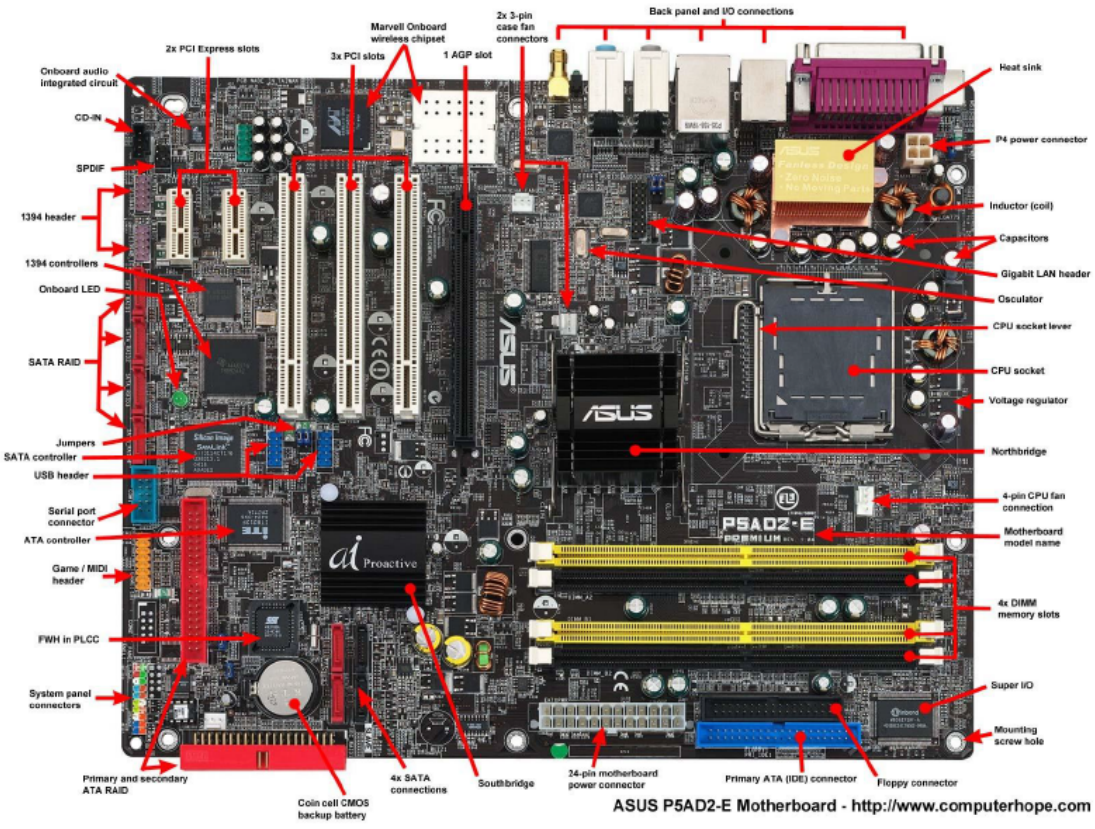

## Memory-Mapped I/O

- Control regiters mapped into memory space
- Control register given unique address
- Advantages
  - Device drivers written entirely in C
  - No special protection needed from OS
  - Every instruction that references memory can reference control registers
- Disadvantages
  - Caching a control register can be disastrous

  ▼ Hardware Complications




▼ Big boi I/O motherboard image




Source: https://www.computerhope.com/cdn/bigmb.jpg