

Computer Labs: I/O Devices

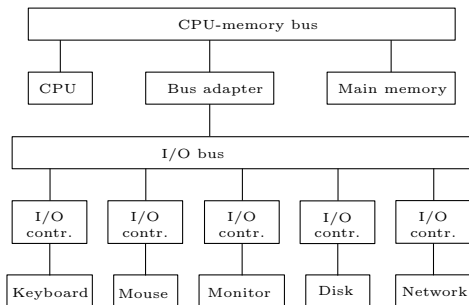
2º MIEIC

Pedro F. Souto (pfs@fe.up.pt)

September 30, 2014

I/O Devices

- ▶ I/O devices provide the interface between the CPU and the outside world.



I/O Controllers

- ▶ Each I/O device is controlled by an electronic component, usually called **controller** or **adapter**.
- ▶ I/O controllers typically include three kinds of registers:
 - Control:** used to request I/O operations
 - Status:** used to get the state of the device or pending I/O operations
 - Data:** used to transfer data to/from the I/O devices
- ▶ Programming at the register level may require a detailed knowledge of the device's operation

How does the CPU access an I/O controller?

Via memory-mapped I/O

- ▶ Portions of the address-space are assigned to I/O devices
- ▶ Access to an I/O device is done using the CPU's memory access instructions

Special I/O instructions

- ▶ I/O uses different address-space and each I/O device is assigned a portion of that address space
- ▶ Access to an I/O device is done using special CPU instructions
 - ▶ Not all CPU's provide special instructions for I/O, the Intel CPU's used in the PC have always provided them
- ▶ I/O instructions are legal only when executing at a high privilege level, typically that of the kernel/supervisor mode

Intel's I/O Instructions

Port Is an abstraction of a device's controller register

- ▶ In the Intel documentation, a port is the name of an address in the I/O address space
- ▶ The I/O address space uses 16-bit addresses
- ▶ Two/four-consecutive 8-bit ports can be treated as 16/32-bit ports – should align them for performance

Instruction IN Input from port, i.e. read from an I/O register

- ▶ The source operand, i.e. the I/O port, is either a “byte immediate” or the DX register
- ▶ The destination operand is one of the AL, AX and EAX registers, depending on the size of the port being accessed

```
in    al, 80h           ; read byte from port 80h
```

Instruction OUT Output to Port, i.e. write to an I/O register

```
mov   dx, 3F8h
out   dx, al           ; write byte to port 3F8h
```

PC's I/O address map

