Operating System Assignment

Name : Rofi Abdillah

Class/Smt : X/7

Student ID : L200144016

Input / Output

In addition to providing abstractions such as processes, address spaces, and files, an operating system also controls all the computer’s I/O (Input/Output) devices. It must issue commands to the devices, catch interrupts, and handle errors. It should also provide an interface between the devices and the rest of the system that is simple and easy to use. To the extent possible, the interface should be the same for all devices (device independence).

* I/O Devices

I/O devices can be roughly divided into two categories: **block devices** and **character devices**. A block device is one that stores information in fixed-size blocks, each one with its own address. Common block sizes range from 512 to 65,536 bytes. All transfers are in units of one or more entire (consecutive) blocks. The essential property of a block device is that it is possible to read or write each block independently of all the other ones. Hard disks, Blu-ray discs, and USB sticks are common block devices

**A character device** delivers or accepts a stream of characters, without regard to any block structure. It is not addressable and does not have any seek operation. Printers, network interfaces,

mice (for pointing), rats (for psychology lab experiments), and most other devices that are not disk-like can be seen as character devices.

Some kinds of I/O devices are :

1. Disks

Disks come in a variety of types. The most common ones are the magnetic hard disks. They are characterized by the fact that reads and writes are equally fast, which makes them suitable as secondary memory (paging, file systems, etc.). Arrays of these disks are sometimes used to provide highly reliable storage. For distribution of programs, data, and movies, optical disks (DVDs and Blu-ray) are also important. Finally, solid-state disks are increasingly popular as they are fast and do not contain moving parts. In the following sections we will discuss magnetic disks as an example of the hardware and then describe the software for disk devices in general.

1. Clocks

In general, the clock refers to a microchip that regulates the timing and speed of all computer functions. Within this chip is a crystal that vibrates at a specific frequency when electricity is applied. The shortest time any computer is capable of performing is one clock, or one vibration of the clock chip. The speed of a computer processor is measured in [clock speed](https://www.computerhope.com/jargon/c/clockspe.htm), for example, 1 MHz is one million cycles, or vibrations, a second. 2 GHz is two billion cycles, or vibrations, a second.

1. User Interface (Keyboard, Mouse, Printer)
2. Input Software

User input comes primarily from the keyboard and mouse (or somtimes touch screens), so let us look at those. On a personal computer, the keyboard contains an embedded microprocessor which usually communicates through a specialized serial port with a controller chip on the parentboard (although increasingly keyboards are connected to a USB port). An interrupt is generated whenever a key is struck and a second one is generated whenever a key is released. At each of these keyboard interrupts, the keyboard driver extracts the information about what happens from the I/O port associated with the keyboard. Everything else happens in software and is pretty much independent of the hardware.

1. Output Software

Basically, output is the result of the input hardware and software operation. For example, when keyboard pressed (input software) and the I/O register of the keyboard give a response, so it will give us the result displayed on the screen such as font, number, etc.

* Device Controllers and Memory Mapped I/O

I/O units often consist of a mechanical component and an electronic component. The electronic component is called the **device controller** or **adapter**. Each controller has a few registers that are used for communicating with the CPU. By writing into these registers, the operating system can command the device to deliver data, accept data, switch itself on or off, or otherwise perform some

action. By reading from these registers, the operating system can learn what the device’s state is, whether it is prepared to accept a new command, and so on.

In addition to the control registers, many devices have a data buffer that the operating system can read and write. For example, a common way for computers to display pixels on the screen is to have a video RAM, which is basically just a data buffer, available for programs or the operating system to write into.

I/O instruction example :

IN REG,PORT,

the CPU can read in control register PORT and store the result in CPU register REG.

* Direct Memory Access

No matter whether a CPU does or does not have memory-mapped I/O, it needs to address the device controllers to exchange data with them. The CPU can request data from an I/O controller one byte at a time, but doing so wastes the CPU’s time, so a different scheme, called **DMA** (**Direct Memory Access**) is often used. The operating system can use only DMA if the hardware has a DMA controller, which most systems do.

* I/O Software Layers

1. Interrupt Handlers

While programmed I/O is occasionally useful, for most I/O, interrupts are an unpleasant fact of life and cannot be avoided. They should be hidden away, deep in the bowels of the operating system, so that as little of the operating system as possible knows about them. The best way to hide them is to have the driver starting an I/O operation block until the I/O has completed and the interrupt occurs. The driver can block itself, for example, by doing a down on a semaphore, a wait on a condition variable, a receive on a message, or something similar.

1. Device Driver

A device driver is a small piece of software that tells the [operating system](https://www.lifewire.com/operating-systems-2625912) and other software how to communicate with a piece of [hardware](https://www.lifewire.com/computer-hardware-2625895). For example, printer drivers tell the operating system, and by extension whatever program you have the thing you want to print open in, exactly how to print information on the page.

1. Device Independent I/O Software

The basic function of the device-independent software is to perform the I/O functions that are common to all devices and to provide a uniform interface to the user-level software.

1. User Space I/O Software

These are the libraries which provide richer and simplified interface to access the functionality of the kernel or ultimately interactive with the device drivers. I/O Libraries (e.g., stdio) are in user-space to provide an interface to the OS resident device-independent I/O SW. For example putchar(), getchar(), printf() and scanf() are example of user level I/O library stdio available in C programming.

NB : I am sorry, sir. I am in individual, I do not have a group