“Module 5: Assignment 1 - Input/Output”

Harsh Siddhapura

“Dr. Dinesh Sthapit”

“October 10, 2023”

1. **Types of I/O devices:** Computer input and output devices are available in a range of sizes and designs. The devices that may be accessible include keyboards, mice, trackpads, touch screens, scanners, audio jacks, and USB devices. Examples of output devices include printers, graphics cards, audio devices, and USB devices. Tapes, solid state drives, magnetic discs, optical drives, and pen drives can all be used as input and output devices.

   Simple vs complex devices: Given that they rely on user input to operate, keyboards and mouse are regarded as essential devices. High-speed network adapters are fundamental components that need specialized performance to handle large amounts of data input and output.

   Factors affecting choice of I/O interface: The selection of I/O interface devices is influenced by several factors. Cost is an important consideration because it costs more to use some junctions than others. Speed is another factor to take into account, since some devices may need rapid data transmission rates to function properly. The type of data to be transmitted (such as character data or block data) has an impact on the interface choice as well. Compatibility with existing hardware and software should also be taken into account when choosing an I/O interface for a device.

2. **Impact of I/O on program speed:** The total performance of a computer programme might suffer from input and output. This is caused by the fact that I/O operations often take up a large amount of computer time and that they frequently go more slowly than the CPU can handle data processing. In comparison to the time it takes a printer to print one character, a CPU, for instance, may process millions of instructions. As a result, software that depends on I/O operations won't be able to fully employ the CPU's processing power, which will cause the application to run slowly.

   Necessary features of an I/O system for good performance: In order for the I/O system to work well and be more efficient, it must have some simple functions. These tasks include:

   - Device Identification: The I/O system has to be able to identify and classify the numerous devices that are connected to the computer. This is important since “different devices will

transmit messages differently, operate at varied speeds, and have distinct controls" (Miley, 2020, p. 294).

- Data integrity: The I/O system has to give I/O devices and the CPU or memory a trustworthy way to exchange data. This includes techniques for managing speedier devices using block adjustments and particular messages for slower devices. One of the technologies, direct memory access (DMA), enables data flow between I/O and memory without the need for the CPU, freeing the CPU for other tasks.
- The I/O system must be able to manage devices with different operating speeds. It ought to make it straightforward to synchronize data with I/O operations and services, reducing the impact on system performance as a whole. Additionally, the system must be able to handle devices with a variety of controls, enabling the CPU to manage I/O for each device in an easy-to-understand and reliable way.

3. I/O devices require special connection methods or interfaces due to several reasons:
    - Different profiles: Each device ought to have its own profile. While some devices might need a block of data, some could just need a single data point. Devices may also need a variety of files, including 8, 16, 32, or 64 bit files. These conflicts need each device to have its own hardware and software.
    - I/O devices and the CPU will work at different speeds, which is a speed mismatch. Synchronizing data flow between devices and CPUs can be difficult, particularly when several devices are attempting to conduct I/O at once. Positive data, like the receiver, facilitates updating the data and helps maintain the speed disparity.
    - Data transfer: Some gadgets, including multimedia devices (which play music and video), need to constantly transmit data. Never surrender. On the other hand, "some devices could work in an unexpected way. Both types of data transport must be supported by I/O systems for them to be effective" (Miley, 2020, p. 298).

- Particular guidelines: Different products are subject to different sets of rules. For instance, in order to operate the print head or motor head on disc drives, electromechanical control is necessary. Using the CPU, obtaining this control takes a lengthy time. Because of this, each device needs a specific interface to meet its needs.

I/O controllers are used to connect I/O devices to the CPU in order to solve these problems. To meet the demands of numerous I/O variables, such as buffering, addressing, synchronization, status, and external control, I/O controllers are outfitted with specialist electronics. This method makes sure that all devices can connect to the CPU in an easy and consistent way.

4. **CPU Management of I/O Data through Registers:** I/O data is managed by the CPU using registers, particularly I/O address registers and I/O data registers. The Memory Address Register (MAR) and Memory Data Register (MDR) and these registers both perform the same functions. Whether the transfer is to memory or I/O is specified by the CPU in a control signal sent to the bus. The transmission address is kept in the I/O address register, whereas the data transferred to or from the I/O device is kept in the I/O data register.

   Single-Word Transfers in Programmed I/O: In programmed I/O/O, data is conveyed in single words. This suggests that each I/O command causes the I/O device to deliver one word of data to the CPU. For instance, the CPU executes the INPUT instruction to transfer the address of the keyboard I/O controller into the I/O address register while reading keyboard input. “The controller then writes a word of data (for instance, a keystroke) to the I/O data register. The information can subsequently be made public or published in a database” (Miley, 2020, p. 301).

   Examples of Programmed I/O processes:

   - Programmed I/O is frequently used for character-based data transfers like reading data from keyboards. For instance, when a user types a letter on a keyboard, the keyboard I/O controller

transmits the letter's ASCII code to the I/O data register. The CPU can then copy the data to a register for further processing.

- Delivering commands using a network controller is another illustration. The CPU sends the proper instructions to the network controller via programmable I/O. The controller receives the instructions and sends them to the CPU for processing in the I/O data register.

As a whole, programmed I/O is a slow process since each I/O data word necessitates a full fetch-execute cycle of instructions. Some I/O activities still make use of it, especially those that call for little quantities of data that can be handled one character at a time.

5. Interrupts: It is a signal or processing event that causes the CPU to momentarily halt what it is doing in order to respond to the request more rapidly. They may result from user activities, input from outside devices, malicious events, or service requests from network administrators, among other things.

   Role of Interrupts in prioritizing tasks:

   - The CPU may advance thanks to interrupts and respond quickly to pressing requests. The CPU pauses its work when an interrupt happens and then attends to it. This response manager manages the outage and creates appropriate action plans to address it.
   - When using interrupts, the CPU may operate more efficiently by giving each task or thread a brief period of time. Move swiftly. The computer can use its CPU resources more effectively and carry out necessary tasks when necessary thanks to this shared time. Additionally, “interruptions provide the CPU the ability to quickly carry out certain activities in response to unforeseen computer events like hardware failure or power outages” (Miley, 2020, p. 306).

   In summary, interrupts play a crucial role in altering the CPU's normal instruction flow, allowing it to prioritize tasks and handle more urgent needs efficiently.

6. To ensure successful bypassing of the CPU for data transfers, several criteria must be fulfilled for DMA to work effectively. These requirements are as follows:
    - Method to connect I/O Interface and Memory: The I/O interface must be able to connect to memory. This can be done by building a link between the two or by joining the two to a single bus.
    - I/O Controller Capable of Reading and Writing to Memory: An associated I/O controller with a certain device must be able to read and write memory. It imitates the “CPU-memory interface by loading registers and reading/writing memory data” (Miley, 2020, p. 313).
    - Make sure that objects are not loaded into multiple positions in the memory address registers of the CPU and the I/O controller at the same time to prevent conflicts between the CPU and the I/O controller. Furthermore, it is not recommended for two different I/O controllers to transmit data simultaneously to memory and I/O on the same bus.

    If three requirements are satisfied, DMA can transmit data without using the CPU, resulting in high performance and effective input/output. Since DMA enables the CPU to perform other tasks while an I/O transfer is in progress, it is especially useful for huge data transfers, such as quick transfers.

7. The I/O controller manages operations like data storage and device interactions as a go-between for the CPU and the I/O devices. They work by attaching the CPU to an external device, such as a network interface, printer, or disc drive. One of an I/O controller's key responsibilities is buffering. They have particular buffers that temporarily store data in memory before sending it to an I/O device. This buffering helps to synchronize the speeds of the CPU and I/O devices, ensuring efficient and smooth data transfer.

    The task of interpreting messages sent to them and taking directions from the CPU is also carried out by I/O controllers. They determine what the I/O device should do based on these instructions. For instance, the disc controller can determine if DMA should be used to write data from memory to the disc. Transfers using Direct Memory Access (DMA) are also made feasible by the I/O controller. Without the

involvement of the CPU, data may be transported directly between memory and I/O devices since it supplies the registers and controls required for DMA. “This enables the CPU to manage data transfer while concentrating on other tasks. They perform tasks such as putting the disc drive head in the right place on the disc, sending information from the buffer to the disc, maintaining” (Miley, 2020, p. 317).

In a nutshell, the I/O controller regulates data storage, communications, and other controls between the CPU and the I/O devices. They manage the specific I/O devices they are connected to, perform buffering operations, accept CPU messages and commands, and allow DMA transfers.

8. Data can be transported physically between computer components through a computer bus. They act as data highways, letting information go back and forth between the CPU, memory, and gadgets. The features of data pathways include the quantity of cables or optical components, data transmission rates, data widths, and the types of devices they support.

   Role of buses in data transfer are as follows:

   - In order to enable data flow in the computer, the bus is essential. They enable the exchange of data between the CPU and the computer's processor, as well as between the CPU and memory and various CPU locations. “A dead connection's bus length can be as short as a few centimeters or as long as several meters for a computer or other piece of external hardware like a printer or monitor” (Miley, 2020, p. 236).
   - Signals for data, address, and control activities are transported through the data bus. “The data bus's proper operation is ensured by the control line, the address line designates the data's destination, and the data line, which carries the actual data to be transmitted” (Miley, 2020, p. 237).

   In a nutshell, the computer system's bus acts as a communication conduit, facilitating effective information transmission between multiple items. They differ from one another in a number of ways and are essential to the computer's operation in general as well as the execution of instructions.

References

Irv Englander. (2020). *ARCHITECTURE OF COMPUTER HARDWARE, SYSTEMS SOFTWARE, AND NETWORKING : an information... technology approach.* John Wiley.