Parul® University

Operating System

Prof. Manish Kumar, Assistant Professor
Computer Science & Engineering

Parul®
University

DIGITAL LEARNING CONTENT

CHAPTER-6.1

I/O SYSTEMS

I/O DEVICE

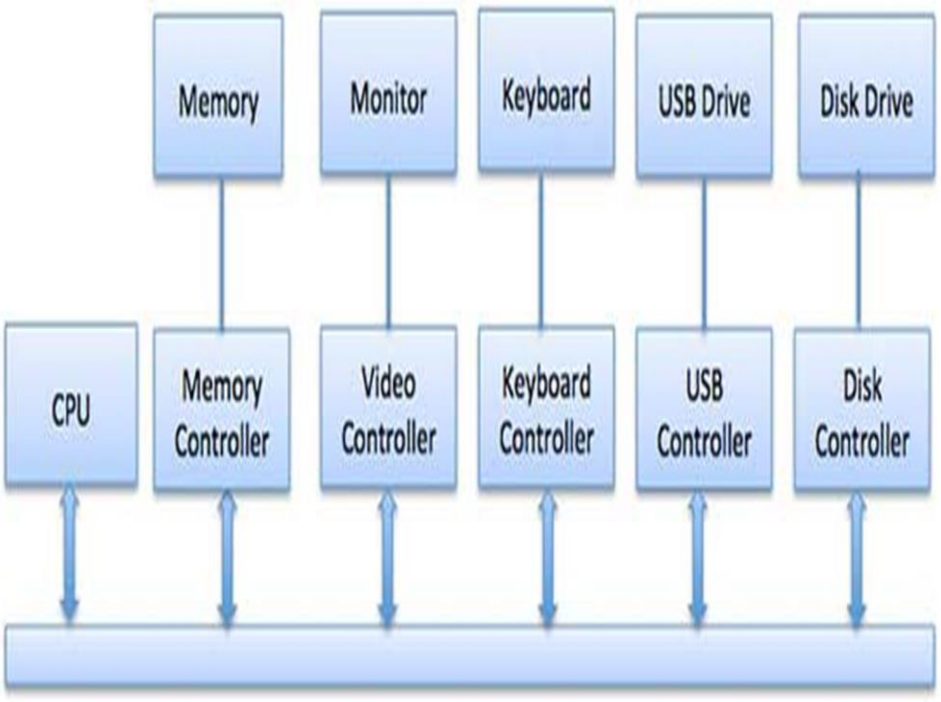
I/O Device are of two Types:

1. **Block Device :** A block device is one with which the driver communicates by sending entire blocks of data. For example, Hard disks, USB cameras, Disk-On-Key etc.
2. **Character Device :** A character device is one with which the driver communicates by sending and receiving single characters (bytes, octets). For example, serial ports, parallel ports, sounds cards etc

Image source : Google

DEVICE CONTROLLER

- **Device controller** is a collection of electronics that can operate a port, a bus, or a device. A serial-port controller is an example of a simple device controller.
- Interaction with a device controller is managed through a device driver.



Device Controller[1]

DEVICE CONTROLLER

- It is a single chip in the computer that controls the signals on the wires of a serial port.
- Any device connected to the computer is connected by a plug and socket, and the socket is connected to a device controller
- Each device controller has a local buffer and a command register.
- It communicates with the CPU by interrupts. A device's controller plays an important role in the operation of that device.
- It functions as a bridge between the device and the operating system.

Image source : Google

EXAMPLES OF DEVICE CONTROLLER

Network card
Disk controller
DVD-ROM controller
USB
Serial port
Sound card

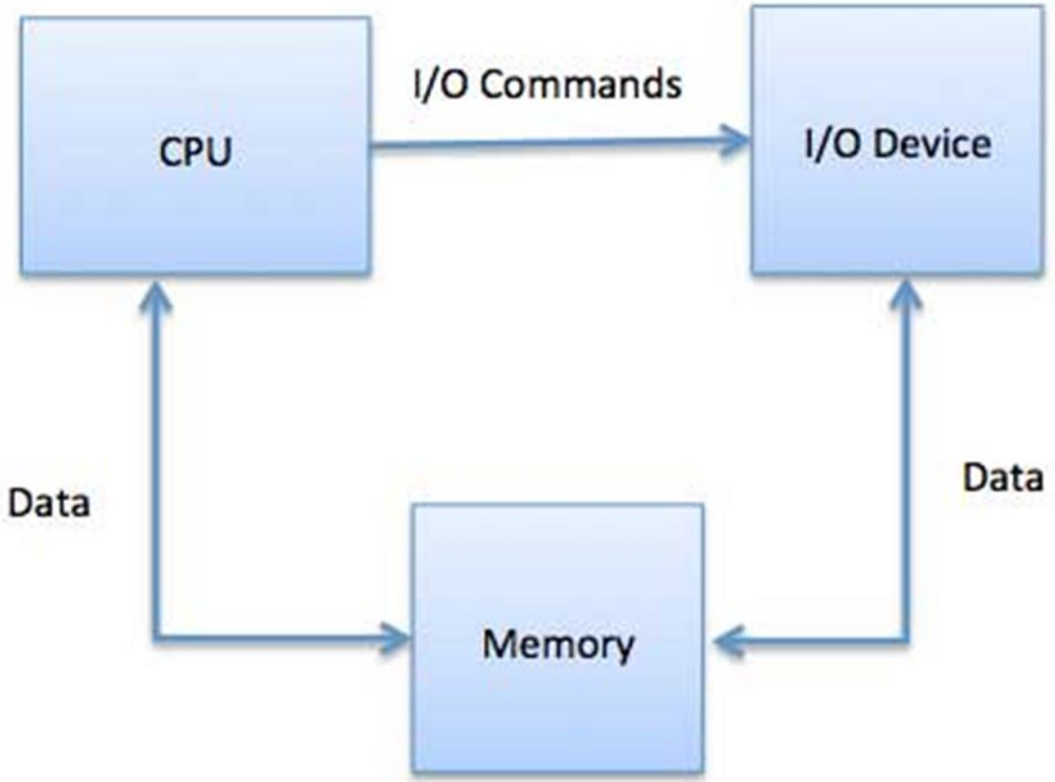
MEMORY MAPPED I/O

- It is the communication method among CPU, Memory and I/O
- This configuration uses same bus and same control signals but unique address space between memory and I/O.
- Same address space is shared by memory and I/O devices. But some addresses represent memory cells, while others represents registers in I/O devices
- In this mechanism, whenever the CPU generates the memory request. Even the control signals are matching but address space is not matching therefore only one node can perform the operation

Image source : Google

MEMORY MAPPED I/O

- Same address space is shared by memory and I/O devices. But some addresses represent memory cells, while others represents registers in I/O devices
- In this mechanism, whenever the CPU generates the memory request. Even the control signals are matching but address space is not matching therefore only one node can perform the operation.



Imemory-Mapped[2]

I/O DEVICE

- **Advantages of memory mapped I/O:**

It makes programming simpler

It does not have special commands to access I/O devices

- **Disdvantages of memory mapped I/O:**

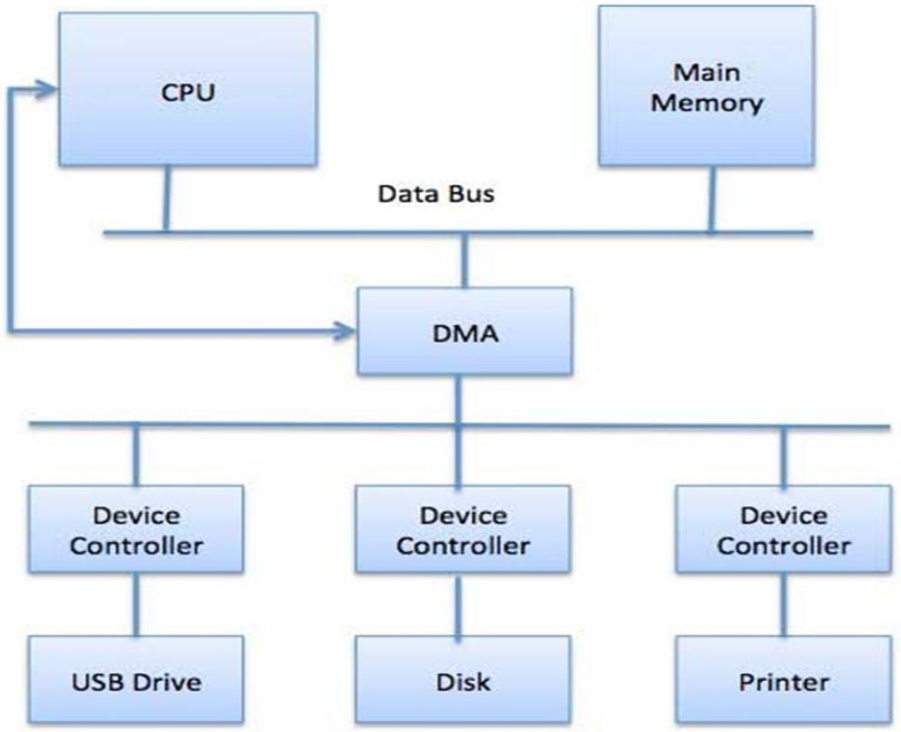
Memory utilization is poor

It is suitable only when the processor consist the limited numbers of I/O ports.

Image source : Google

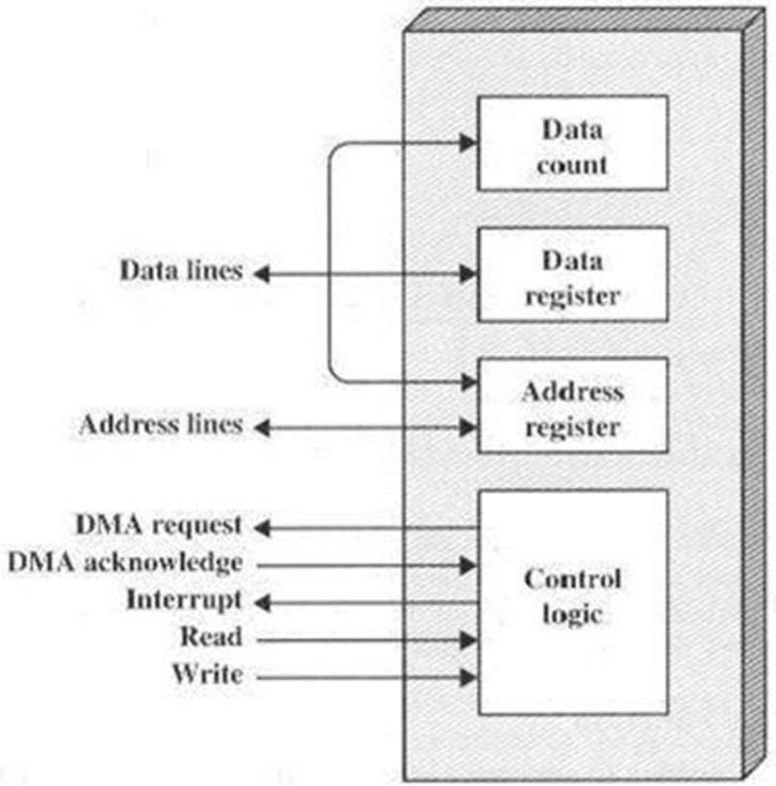
DMA(DIRECT MEMORY ACCESS)

- A special unit may be provided to allow transfer of block of data directly between external device and the main memory, without continuous intervention by the processor. This approach is called Direct Memory Access (DMA)
- This means that once reading and writing has begun, the remainder of the data can be transferred to and from memory without processor (CPU) intervention



DIRECT MEMORY ACCESS

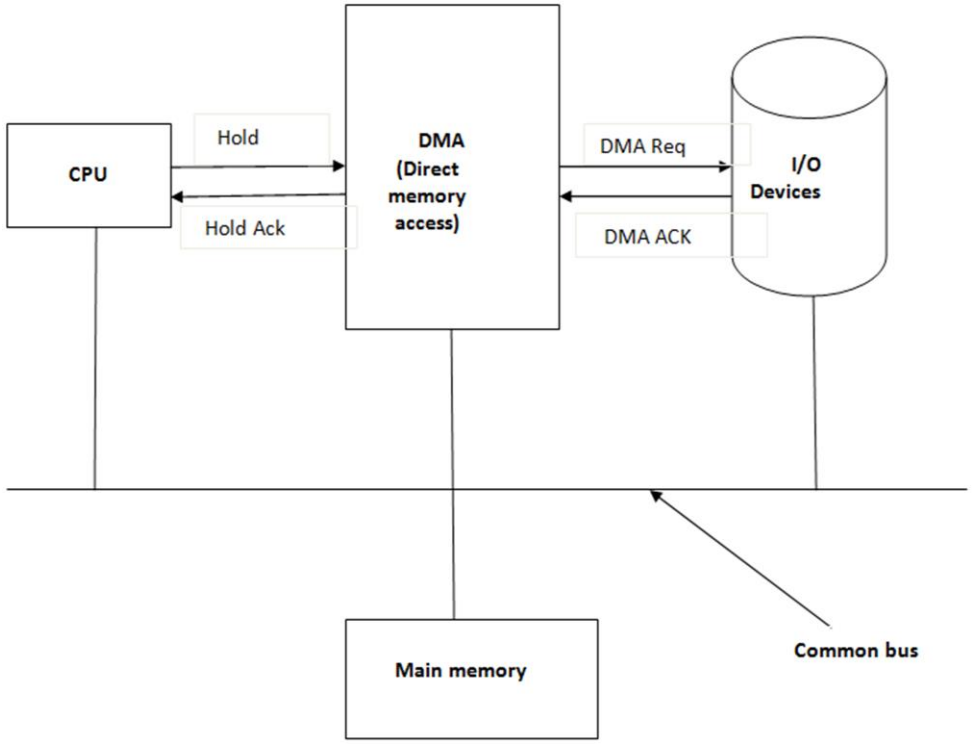
- DMA is particularly useful on devices like disks, where many bytes of information can be transferred in single I/O operations
- When used in conjunction with an interrupt, CPU is notified only after the entire block of data has been transferred
- Direct Memory Access is an I/O technique that allows a control unit to directly access main memory.
- Without DMA, the processor is responsible for the physical movement of data between main memory and device.



Block Diagram of DMA[4]

WORKING OF DIRECT MEMORY ACCESS

- During execution of first part of the user program, the CPU initializes the DMA with control signals, IO address, Memory starting address and the count value
- Then the CPU is busy with execution of the part1(user program part1). Simultaneously the DMA initializes the operation to the corresponding IO device by sending the DMA request signal
- After Preparing the data, the I/O device sends the DMA ACK signal to the DMA



DMA Working[5]

WORKING OF DIRECT MEMORY ACCESS

- Whenever the DMA module receives the ACK signal from the IO, it enables the HOLD signal for the purpose of gain the control of the bus and it is waiting for the HOLD ACK signal.
- When the processor receives the HOLD signal it enables the HOLD ACK (HLDA) signal.
- Whenever DMA receives HLDA signal, it transfer the data from the IO device to memory until the count becomes zero
- Until transfer the data, the CPU is in the blocked state. After completion of data transfer operation, the DMA reestablish the bus connection to the CPU.

Parul®
University

DIGITAL LEARNING CONTENT

MODES OF DIRECT MEMORY ACCESS

- Burst mode
- Cycle stealing mode
- Blocked mode

Image source : Google

MODES OF DIRECT MEMORY ACCESS

- **Burst mode:** In the burst mode of DMA, it transfer a bulk amount of data from the IO to M/Y, after receiving the HLDA signal.
- **Cycle stealing mode:** The DMA forcefully suspends the CPU operation and take the control of bus that means without receiving the hold acknowledgement(HLDA) signal, the DMA gains the control of bus. In this mode, DMA transfers the very important data to the memory and reestablish the connection to the CPU.
- **Blocked mode :** DMA transfers the data in the form of blocks.

Image source : Google

DMA

- **Advantages of DMA:**

High transfer rates

Fewer CPU cycles for each transfer

- **Disdvantages of DMA:**

Expensive system

Synchronization mechanism must be provided in order to avoid accessing non-updated information from RAM

Image source : Google

POLLING VS INTERRUPT

- A computer must have a way of detecting the arrival of any type of input.
- There are two ways that this can happen, known as **polling** and **interrupts.**
- **Polling I/O :** The process of periodically checking status of the device to see if it is time for the next I/O operation, is called polling.
- The I/O device simply puts the information in a Status register, and the processor must come and get the information**.**

Image source : Google

POLLING VS INTERRUPT

- **Interrupt I/O :**An alternative scheme for dealing with I/O is the interrupt-driven method.
- An interrupt is a signal to the microprocessor from a device that requires attention.
- A device controller puts an interrupt signal on the bus when it needs CPU's attention when CPU receives an interrupt, It saves its current state and invokes the appropriate interrupt handler using the interrupt vector (addresses of OS routines to handle various events).
- When the interrupting device has been dealt with, the CPU continues with its original task as if it had never been interrupted.

Image source : Google

Parul® University

DIGITAL LEARNING CONTENT

SHARED VS DEDICATED DEVICE

•**Dedicated device:** are those device, which can be shared by only one user at a time for ex. Printer and CD-ROM drives.

•**Shared Devices**: Shared devices are those devices which can be shared by concurrent processes without any deadlock. For ex- Disks, Ethernet port

Image source : Google

PRINCIPLE OF I/O SOFTWARE

•**GOALS OF I/O SOFTWARE:**

•**Device Independence:** Device independence enables I/O devices to be treated in a standard, uniform way. For instance, how an application can open a file on a disk without knowing what kind of disk it is, and how new disks and other devices can be added to a computer without the operating system being disrupted.

•**Error Handling:** An operating system that uses protected memory can guard against many kinds of hardware and application errors. There are several aspects to error handling including: Detection, correction and reporting

Image source : Google

Parul®
University

DIGITAL LEARNING CONTENT

PRINCIPLE OF I/O SOFTWARE

•**GOALS OF I/O SOFTWARE:**

•**Buffering:** A buffer is a memory area that stores data while they are transferred between two devices or between a device and an application.

Buffering is done for the following reasons :

- One reason is to cope with a speed mismatch between the
- producer and consumer of a data stream.

Image source : Google

I/O DEVICE

Interrupt: Interrupt is unusual event of disturb.

- The CPU hardware has a wire called the interrupt request line that the CPU senses after executing every instruction.
- The CPU responds to the interrupt only after completion of the current instruction execution

Image source : Google

I/O DEVICE

Interrupt Handler:

- The interrupt handler determines the cause of the interrupt and performs the necessary processing, then executes a return from interrupt instruction to return the CPU to the execution state prior to the interrupt.
- We say that the device controller raises an interrupt by asserting a signal on the interrupt request line, then, CPU catches the interrupt and dispatches to the interrupt handler, and the handler clears the interrupt by servicing the device.

Image source : Google

- Instruction cycle with interrupt consist of three sub cycles:

1. **Fetch cycle**
2. **Execution cycle**
3. **Interrupt cycle**

Image source : Google

I/O DEVICE

Interrupt flag/pin:

- In the processor environment, interrupt flag is present to enable or disable the interrupt. When the INTR (interrupt) flag is in enable state, there is a need of interrupt cycle to service the interrupt
- Otherwise, after completion of current instruction, the CPU fetch the next current instruction from the memory based on the program counter. In the processor, INTR pin is used. It holds the status of interrupt
- The CPU responds to the interrupt only after completion of t current instruction execution

Image source : Google

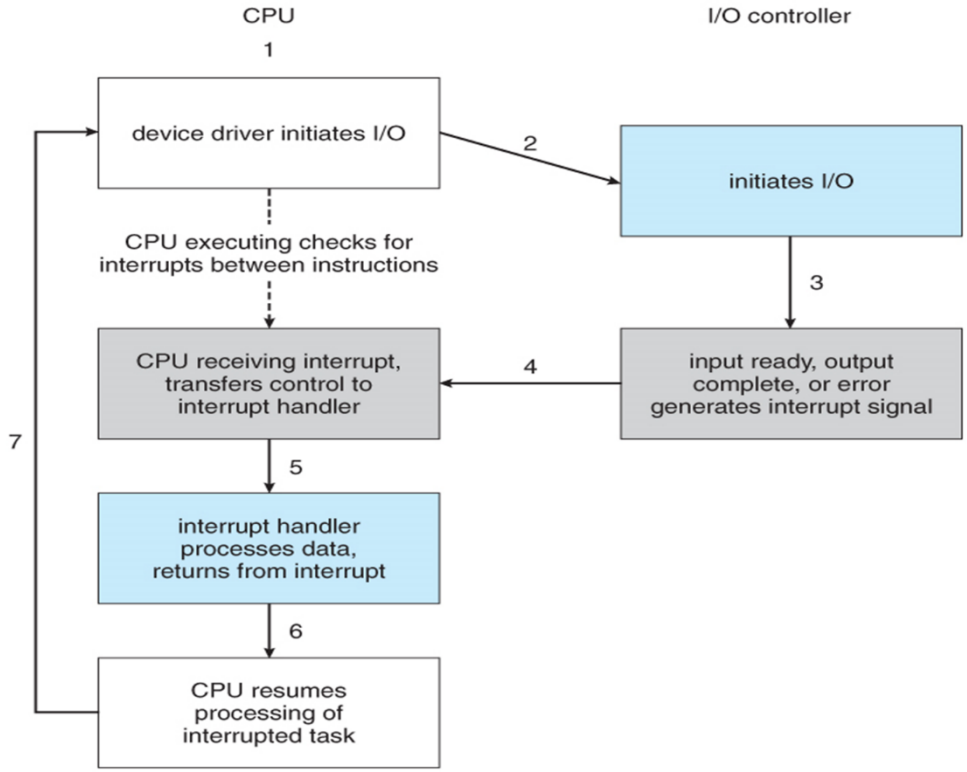
INTERRUPT HANDLING

- Whenever, the CPU recognize any interrupt after completion of current instruction, it push the program counter value onto the stack then loads the program counter with vector address
- Then the control is transferred to the interrupt subprogram. It continue the execution until “IRET” instruction

Image source : Google

INTERRUPT HANDLING

- Whenever “IRET” instruction is executed by the processor. It invokes the pop operation to restore the program counter with return address
- Now the CPU is fetching the next instruction from the main memory
- The entry point/address of interrupt subprogram is called vector address



Interrupt handling[6]

DEVICE DRIVERS

- **A device driver** is a program that controls a particular type of device that is attached to user computer. There are device drivers for printers, displays, CD-ROM readers, hard disk and so on
- A driver provides a software interface to hardware devices, enabling operating systems and other computer programs to access hardware functions without needing to know precise details of the hardware being used.

Image source : Google

DEVICE DRIVERS

- **A device driver** typically communicates with the device through the computer bus or communications subsystem to which the hardware connects.
- Drivers are hardware-dependent and operating-system-specific.
-

Image source : Google

PURPOSE OF DEVICE DRIVERS

- Device drivers simplify programming by acting as translator between a hardware device and the applications or operating systems that use it.
- Programmers can write the higher-level application code independently of whatever specific hardware the end-user is using.
- For example, a high-level application for interacting with a serial port may simply have two functions for "send data" and "receive data". At a lower level, a device driver implementing these functions would communicate to the particular serial port controller installed on a user's computer

Image source : Google

PARTS OF DEVICE DRIVERS

- **Device Specific**: This portion of a device remains the same across all operating system and is more about understanding and decoding the device data than software programming. A data sheet for a device is a document with technical details of the device, including its operation, performance, programming etc
- **OS specific:** This portion is tightly coupled with the OS mechanisms of user interfaces thus differentiates a linux device driver from windows device driver

Image source : Google

GENERAL CLASSIFICATION OF DEVICE DRIVER

- Serial port drivers
- Network device drivers
- Flash
- Bus driver

Image source : Google

DEVICE INDEPENDENT I/O SOFTWARE

- The device independent code contains most of the I/O functionality, but not most of the code since there are many drivers
- All drivers do the same thing in slightly different ways due to slightly different controllers. Functions of device independent I/O software:

- ❑ It provides buffering.
- ❑ Support for error reporting.
- ❑ Provides uniform interfacing for device drivers.
- ❑ Perform allocation and releasing dedicated devices
- ❑ Software provides a device independent block size

Image source : Google

BUFFERING

- **“Buffering”** is a technique by which the device manager can keep slower I/O devices busy during times when a process is not requiring I/O operations
- **Input buffering:** is a technique of having the input device read information into the primary memory before the process requests it
- **Output buffering:** is the technique of saving information in memory and then writing to the device while the process continues execution

Image source : Google

TYPES OF BUFFERING

- Single Buffering
- Double buffering
- Circular buffering
- No buffering

Image source : Google

Parul®
University

DIGITAL LEARNING CONTENT

REFERENCES

[1] Device Controllers. Tutorialspoint.
https://www.tutorialspoint.com/operating_system/os_io_hardware.html
[2] Memory Mapped. Tutorials point
https://www.tutorialspoint.com/operating_system/os_io_hardware.html
[3] DMA. Tutorialspoint
https://www.tutorialspoint.com/operating_system/os_io_hardware.html
[4] Block Diagram of DMA.
http://inputoutput5822.weebly.com/direct-memory-access.html
[5] Working of DMA
https://binaryterms.com/direct-memory-access-dma.html
[6] Interrupt handling
https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/13_IOSystes.html