I/O Device Management

- Any device that transfers data to and from a computer and to and from a peripheral device
- All computers have physical devices for acquiring input and producing output.
- OS is responsible for managing and controlling IO operations and IO devices.

The IO units which consists mechanical components are called IO devices such as hard disk drive, printer.

There are two types of devices: block and character devices

Block devices: Stores information in fixed- sized blocks, each one with its own address. Read or write is possible independent to other blocks – Direct Access
Example: disk

Character Devices: Delivers or accepts a stream of characters without regard to any block structure. It is not addressable such as printer, mouse, and keyboard

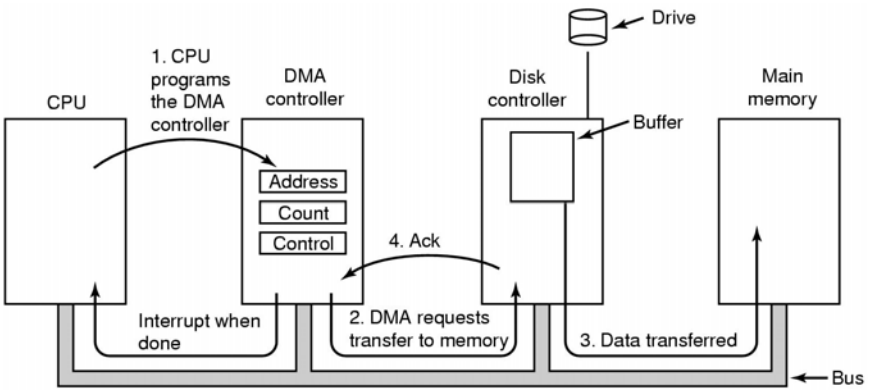
Some devices are neither block nor character such as clock.

Device Controllers

- IO units typically consist of mechanical and electronic components.
- A controller is a collection of electronics that can operate a bus or a device.
- On PC, it often takes the form of printer circuit card that can be inserted into an expansion slot.
- A single controller can handle multiple devices; some devices have their own built in controller.
- The controller has one or more registers for data and signals.
- The processor communicates with the controller by reading and writing bit patterns in these registers.
- When transferring a disk block of size 512 bytes, the block first assembled bit by bit in a buffer inside the controller. After its checksum has been verified and the blocks declared to error free, it can’t be copied to the main memory.

Direct Memory Access (DMA)

- DMA is a feature of modern computers and microcomputers that allows certain hardware subsystems within the computer to access system memory for reading/writing independently of the CPU.
- Without DMA, the CPU is fully occupied for the entire duration of the read or write operation, and is unavailable to perform other works.
- With DMA, the CPU would initiate the transfer, do other operations while the transfer is in progress and receive an interrupt form the DMA controller once the operation has been done.
- Many computers avoid burdening the CPU by offloading some of its work to special purpose processor called DMA controller.
- It consists of memory address register, a byte count register and one or more control registers.
- More commonly, a single DMA controller is available (in built in main board) for regulating transfer to multiple devices but some systems may have integrated with disk controllers.



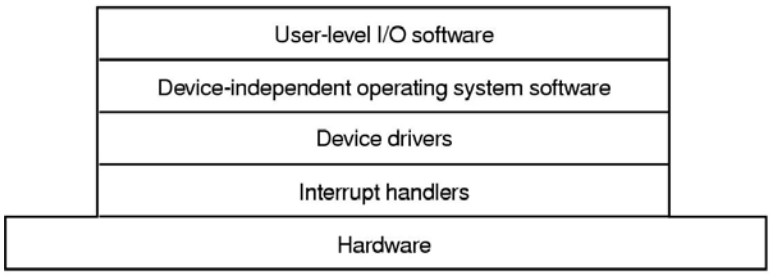
How DMA Works?

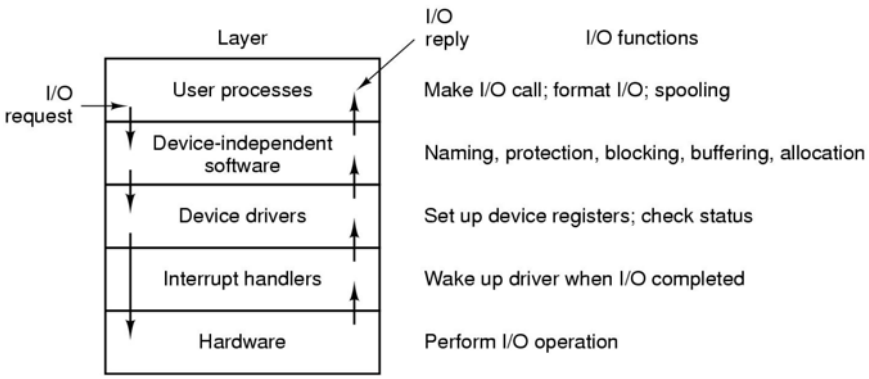
1. The CPU programs the DMA controller by its registers so it knows what to transfer where. It also issue a command to disk controller telling it to read data from disk to its internal buffer and verify the checksum. When valid data are in disk's controller buffer, DMA can begin.
2. The DMA controller initiates the transfer by issuing a read request over the bus to disk controller.
3. Data transferred from disk controller to memory.
4. When transfer completed, the disk controller sends an acknowledgement signal to DMA controller. The DMA controller then increments the memory address to use and decrement the byte count. This continues till the byte count is greater than 0.
5. When transfer is complete, the DMA controller interrupts the CPU.

I/O software Goals

1. **Device Independence:** A program that reads a file as input should be able to read a file on a floppy, hard disk, CD or USB without having to modify for each type of file.
2. **Uniform Naming:** Name of file or device should simply be a string or integer and not dependent upon the device anyway. All files and devices should be addressed in same way: by pathname
3. **Error Handling:** In general error should be handled by hardware. If the device controller finds an error, it should try to correct itself. If it can't, then the device drivers should handle it, perhaps just by trying to read the block again.
4. **Synchronous vs. Asynchronous Transfer :** Most I/O is asynchronous i.e. the CPU starts the transfer and does something else until the interrupt arrives. In synchronous, after a read system call the program is automatically suspended until the data are available in the buffer.
5. **Buffering:** Often data come off a device can' be stored directly in its final destination. E.g. when a packet comes in off a network, the OS doesn't know where to put it until it has stored the packet somewhere and examined it.

Software Layer Structure





Interrupt Handlers

Block the driver until the I/O has completed and the interrupt occur.

The interrupt handler determines the cause of the interrupt and performs the necessary processing.

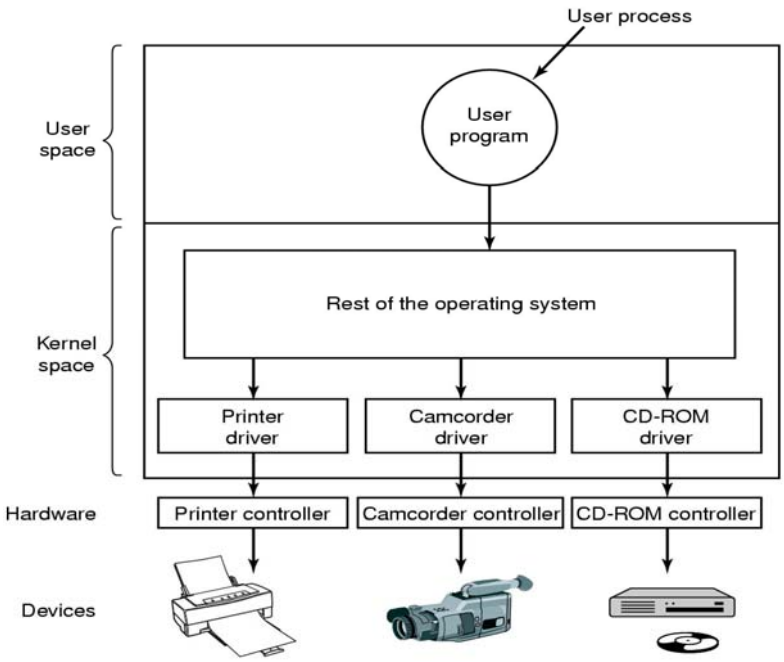
1. Save any register (including the Process Status Word, PSW) that have not already been saved by the interrupt hardware.
2. Set up a stack for interrupt service procedure
3. Ack interrupt controller
4. Copy registers from where they were saved(stack) to the process table
5. Run the interrupt service procedure.
6. Set up the MMU context for the process to run next.
7. Load new process registers, including PSW.
8. Start running the new process

Device Drivers

- Encapsulates detail of device
- Each I/O device attached to a computer needs some device specific code for controlling it, called device driver, is generally written by the device's manufacturer and delivered along with the device.
- Each device driver normally handles one device type, or at most one class of closely related devices.
- In some systems, the OS is a single binary program that contains all of the drivers that it will need compiled into it.

Functions

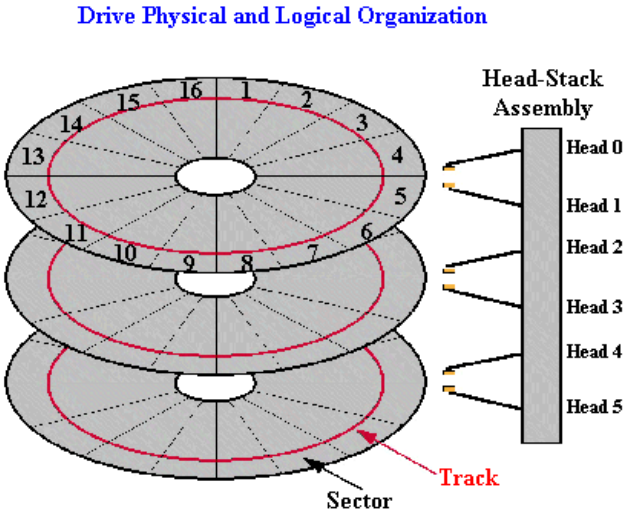
- Accept read and write requests from the device independent software above it.
- Initialize the devices if necessary
- Manage power requirement and log events
- It checks the status of devices-in use or free
- Decides which command to issue if there is command queue



Disk Structure

- Disks come in many sizes and speeds, and information may be stored optically or magnetically; however, all disks share a number of important features.
- E.g. floppy disks, hard disks, CDs and DVDs.
- Disk surface is divided into number of logical block called sectors and tracks

Hard Disk Structure



Disk Operations

- Latency Time: The time taken to rotate from its current position to a position adjacent to the read write head.
- Seek: The process of moving the arm assembly to new cylinder

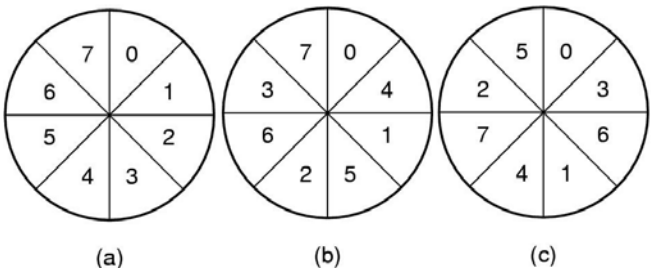
- To access a particular record; first the arm assembly must be moved to the appropriate cylinder and then rotate the disk until it is immediately under the read-write head.
- The time taken to access the whole record is called transmission time.

Disk Formatting

- Before a disk can store data, it must be divided into sectors that the disk controller can read and write, called low-level formatting
- The sector typically consists of preamble, data and ECC.
- The preamble contains the cylinder and sector number and the ECC contains redundant information that can be used to recover from read error.
- The size depends upon the manufacturer; depending on reliability.

Preamble	Data	ECC

- If the disk I/O operations are limited to transferring a single sector at a time, it reads sector from the disk and doing the ECC calculation, and transfers to main memory, during this time the next sector will fly by the head.
- When transferring completes the controller will have to wait almost an entire rotation for the second sector to come around again.
- This problem can be eliminated by numbering the sectors in an interleaved fashion when formatting the disk.
- According to the copying rate interleaving may be of single or double.



Error Handling

- Most frequently, one or more sectors become defective or most disk even come from factory with bad blocks.
- Depending on the disk and controller in use, these blocks handled in variety of ways
    - Bad blocks are handled manually- for example run chkdsk command in DOS to find bad block or a format command to create new block – data resides on bad blocks usually are lost.
    - Using bad block recovery: the controller maintains the list of bad blocks on the disk and for each bad block one of the spares is substituted.

Input Software

Keyboard

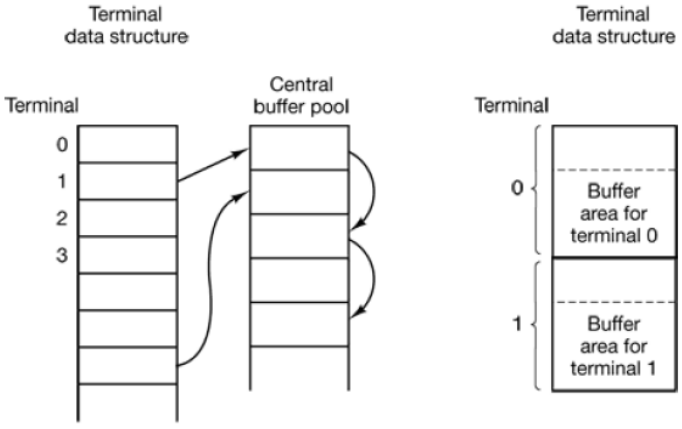
- Basic job of keyboard driver is to collect input from keyboard and pass to user program when they are read from the terminal
- 2 possible philosophies can be adopted from the driver
- 1st Philosophy
    - Driver's job is just to accept input and pass it unmodified
    - A program reading from the terminal gets a raw sequence of ASCII codes
    - E.g. dste ← ← ← date CR
    - Wanted to type *date* but typed *dste* so 3 back spaces and a carriage return.
    - Hence, 11 ASCII codes generated

- Not all programs want this much detail
- Just want the corrected input
- 2nd Philosophy
  - Driver handles all intraline editing and just delivers corrected lines to user programs
- 1st one: character oriented – raw mode – non canonical
- 2nd one: line oriented – cooked mode – canonical

- 2 approaches to character buffering are common

  - **1st one:**
  - Driver contains a central pool of buffers, each buffer holding perhaps 10 characters.
  - Associated with each terminal is a data structure, which contains a pointer to the chain of buffers.
  - Where characters are passed to a user program, the buffers are removed and put back in the central pool

  - **2nd one:**
  - Do the buffering directly in the terminal data structure itself, with no central pool of buffer.



Echoing:

- whether to display all the things typed from keyboard on screen or not
- e.g. password to be displayed or not

Tab Handling:

- depends upon driver to compute the proper number of spaces from current cursor position
- Characters handled especially in canonical mode

Character	**POSIX Name**	**Comment**
CTRL – H	ERASE	Backspace one character
CTRL – U	KILL	Erase entire line being typed
CTRL – V	LNEXT	Interpret Next character literally
CTRL – S	STOP	Stop output
CTRL – Q	START	Start output

Output Software

- Simpler than input
- Computer sends characters to the terminal and they are displayed there
- Generally a block of characters – a line is written to the terminal in one system call
- The method commonly used for RS232 terminal is to have output buffers associated with each terminal
- The buffers may come from same pool or be dedicated
- When the program writes to the terminal, the output is first copied to the buffer
- After all the outputs have been copied to the buffer, the first character is output and the driver goes to sleep. When the interrupt comes in, the next character is output and so on.

Escape Sequence

- Screen editors must be able to update the screen in complex ways such as replacing one line in the middle of the screen
- To accommodate this need, most terminals support a series of commands to move the cursor, insert or delete characters or lines at the cursor.
- These commands are often called escape sequence.

Escape Sequence	Meaning
ESC [n A	Move up n lines
ESC [n B	Move down n lines
ESC [n C	Move right n spaces
ESC [n D	Move left n spaces
ESC [n M	Delete n lines at cursor

Terminal Hardware

RS – 232 Terminal Hardware

- Are the hardware device containing both a keyboard and a display and communicate using a serial interface, one bit at a time
- Use 9 pin or 25 pin connector
- 1 pin for transmitting data, 1 pin for receiving, 1 pin ground and other pins for control functions
- Lines in which characters are sent 1 – bit at a time are called serial lines
- To send a character over a serial line to an RS-232 terminal, the computer must transmit 1-bit at a time, prefixed by a start bit and followed by 1 or 2 stop bits.
- A parity bit for error detection may be inserted preceding the stop bits.
- Start bit – data bits – parity bit – stop bit
- RS 232 are still commonly used in mainframe and mini computers.

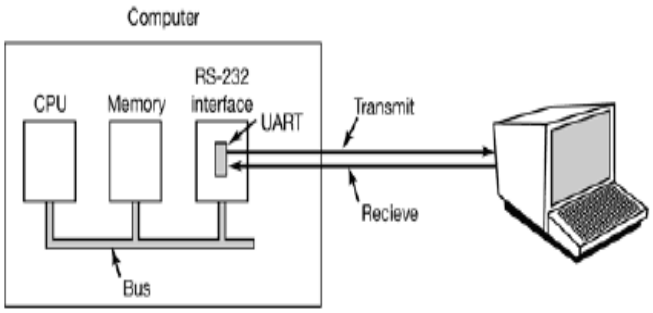


Figure 5-34. An RS-232 terminal communicates with a computer over a communication line, one bit at a time.

- RS 232 terminals are character oriented i.e. the window displays a certain number of lines of text, each of maximum size. A typical size is 25 lines of 80 each.
- UART chips (Universal Asynchronous Receiver Transmitters) have been developed to do the character – to – serial and serial – to – character conversion.
- UARTs are attached to the computer by plugging RS-232 interface cards into the bus.
- The UART can send and receive character simultaneously.