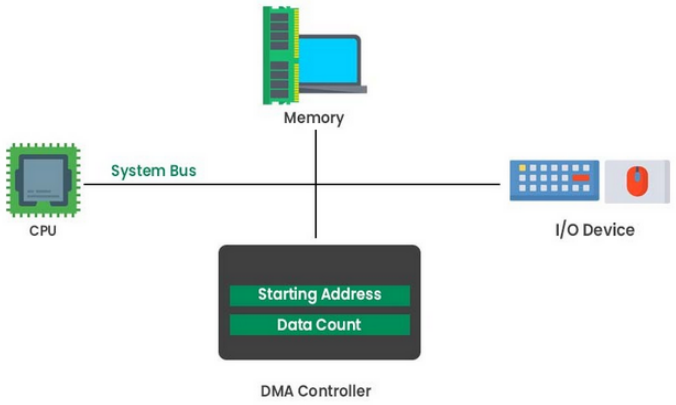
Direct Memory Access (DMA)

- Direct memory access (DMA) is a method that allows an input/output (I/O) device to send or receive data directly to or from the main memory, bypassing the CPU to speed up memory operations. The process is managed by a chip known as a DMA controller (DMAC).



Why it is needed?

A computer's system resource tools are used for communication between hardware and software. The four types of system resources are:

- I/O addresses
- Memory addresses
- Interrupt request numbers (IRQ)
- Direct memory access (DMA) channels

DMA channels are used to communicate data between the peripheral device and the system memory.

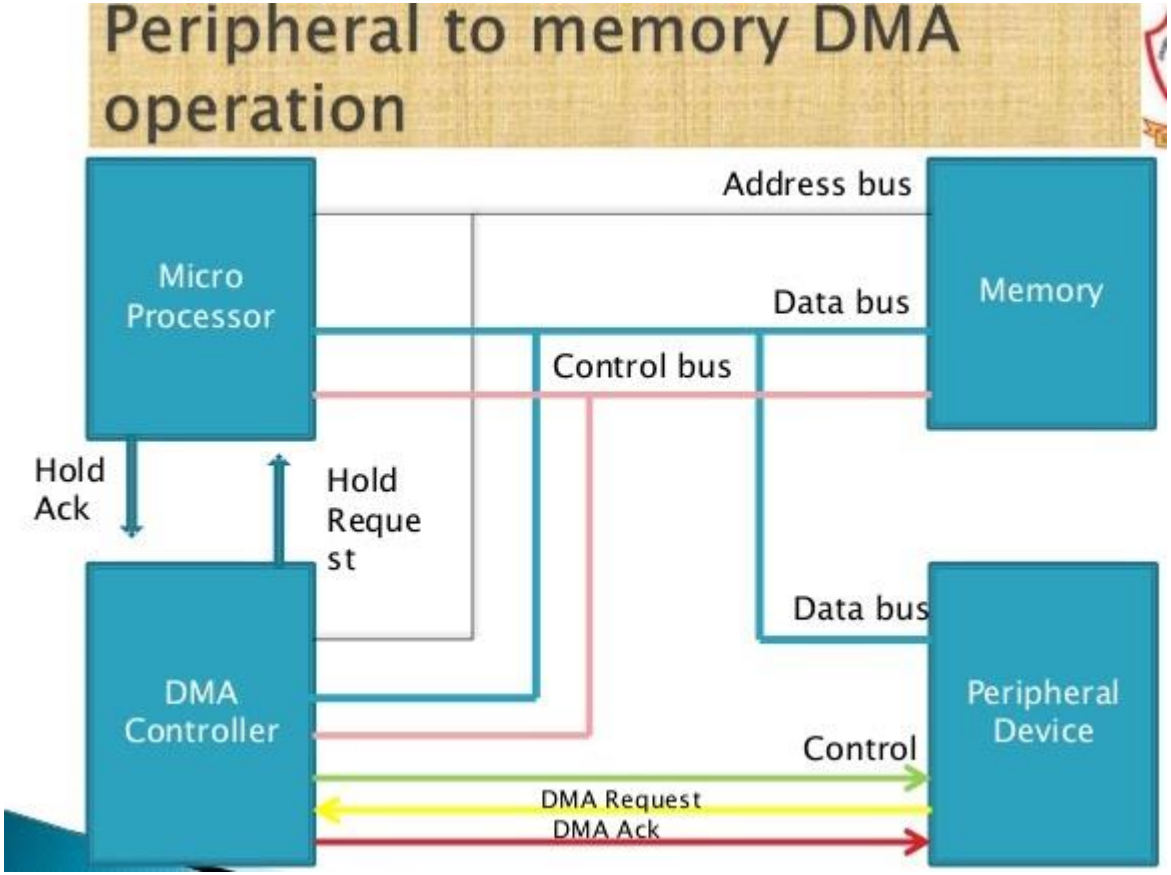
Why it is needed?

- Without DMA, during I/O operation, the whole CPU is occupied for read and write operation, and thus is unavailable for other works.
- With DMA, CPU can do other works while the transfer is in progress.
- DMA is used in disk drives, graphics cards, network cards and sound cards.

How DMA operations are performed?

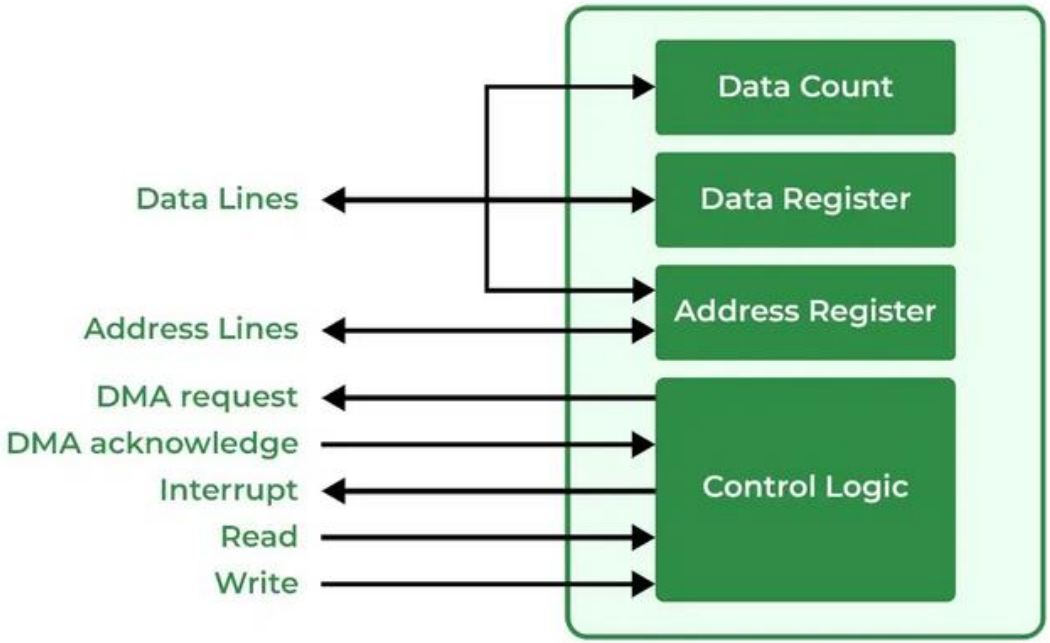
- Initially, when any device has to send data between the device and the memory, the device has to send DMA request (DRQ) to DMA controller.
- The DMA controller sends Hold request (HRQ) to the CPU and waits for the CPU to assert the HLDA.
- Then the microprocessor tri-states all the data bus, address bus, and control bus. The CPU leaves the control over bus and acknowledges the HOLD request through HLDA signal.
- Now the CPU is in HOLD state and the DMA controller has to manage the operations over buses between the CPU, memory, and I/O devices.

How DMA operations are performed?



DMA Controller Diagram in Computer Architecture

DMA Controller is a type of control unit that works as an interface for the data bus and the I/O Devices. As mentioned, DMA Controller has the work of transferring the data without the intervention of the processors, processors can control the data transfer. DMA Controller also contains an address unit, which generates the address and selects an I/O device for the transfer of data. Here we are showing the block diagram of the DMA Controller.



Block Diagram of DMA Controller

Types of Direct Memory Access

Single-Ended DMA: Single-Ended DMA Controllers operate by reading and writing from a single memory address. They are the simplest DMA.

Dual-Ended DMA: Dual-Ended DMA controllers can read and write from two memory addresses. Dual-ended DMA is more advanced than single-ended DMA.

Arbitrated-Ended DMA: Arbitrated-Ended DMA works by reading and writing to several memory addresses. It is more advanced than Dual-Ended DMA.

Interleaved DMA: Interleaved DMA are those DMA that read from one memory address and write from another memory address.

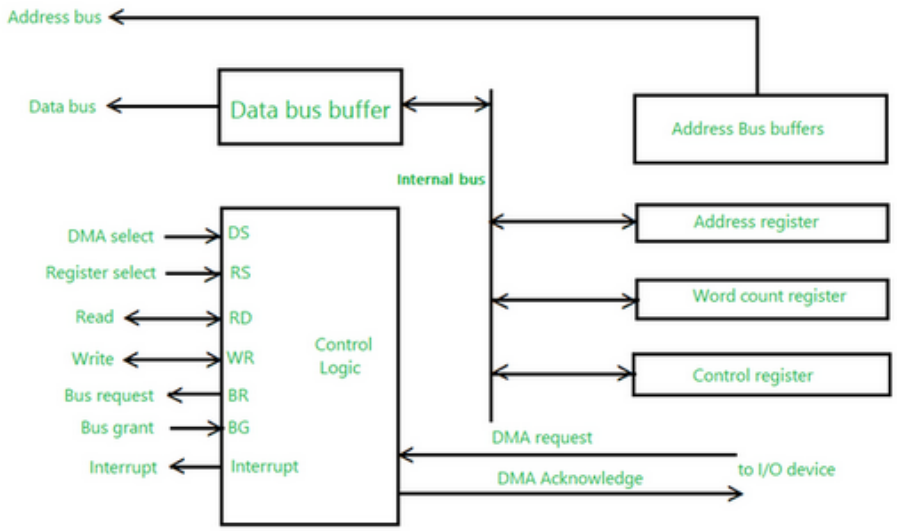
Working of DMA Controller

The DMA controller registers have three registers as follows.

- **Address register** – It contains the address to specify the desired location in memory.
- **Word count register** – It contains the number of words to be transferred.
- **Control register** – It specifies the transfer mode.

Note: All registers in the DMA appear to the CPU as I/O interface registers. Therefore, the CPU can both read and write into the DMA registers under program control via the data bus.

The figure below shows the block diagram of the DMA controller. The unit communicates with the CPU through the data bus and control lines. Through the use of the address bus and allowing the DMA and RS register to select inputs, the register within the DMA is chosen by the CPU. RD and WR are two-way inputs. When BG (bus grant) input is 0, the CPU can communicate with DMA registers. When BG (bus grant) input is 1, the CPU has relinquished the buses and DMA can communicate directly with the memory.



Working Diagram of DMA Controller

Explanation: The CPU initializes the DMA by sending the given information through the data bus.

- The starting address of the memory block where the data is available (to read) or where data are to be stored (to write).
- It also sends word count which is the number of words in the memory block to be read or written.
- Control to define the mode of transfer such as read or write.
- A control to begin the DMA transfer

DMA Transfer Modes

1. Burst or block transfer DMA
2. Cycle steal or single byte transfer DMA.
3. Transparent or hidden DMA.

DMA Transfer Modes

1) Burst or block transfer DMA

- It is the fastest DMA mode. In this two or more data bytes are transferred continuously.
- Processor is disconnected from system bus during DMA transfer. N number of machine cycles are adopted into the machine cycles of the processor where N is the number of bytes to be transferred.
- DMA sends HOLD signal to processor to request for system bus and waits for HLDA signal.
- After receiving HLDA signal, DMA gains control of system bus and transfers one byte. After transferring one byte, it increments memory address, decrements counter and transfers next byte.
- In this way, it transfer all data bytes between memory and I/O devices. After transferring all data bytes, the DMA controller disables HOLD signal & enters into slave mode.

Burst or block transfer DMA

- **Pros:**
  - Most Efficient way for DMA Transfer.
- **Cons:**
  - Rate of DMA Transfer will be less.
  - CPU won’t be blocked entire time.

DMA Transfer Modes

2) Cycle steal or single byte transfer DMA.

- In this mode only one byte is transferred at a time. This is slower than burst DMA.
- DMA sends HOLD signal to processor and waits for HLDA signal on receiving HLDA signal, it gains control of system bus and executes only one DMA cycle.
- After transfer one byte, it disables HOLD signal and enters into slave mode.
- Processor gains control of system bus and executes next machine cycle. If count is not zero and data is available then the DMA controller sends HOLD signal to the processor and transfer next byte of data block.

DMA Transfer Modes

3) Transparent or Hidden DMA transfer

- Processor executes some states during which is floats the address and data buses. During this process, processor is isolated from the system bus.
- DMA transfers data between memory and I/O devices during these states. This operation is transparent to the processor.
- This is slowest DMA transfer. In this mode, the instruction execution speed of processor is not reduced. But, the transparent DMA requires logic to detect the states when the processor is floating the buses.

Transparent or Hidden DMA transfer

- **Pros:**
  - CPU will not be blocked at all.
- **Cons:**
  - Slowest DMA transfer rate.

- **Advantages of DMA Controller**
  - Data Memory Access speeds up memory operations and data transfer.
  - CPU is not involved while transferring data.
  - DMA requires very few clock cycles while transferring data.
  - DMA distributes workload very appropriately.
  - DMA helps the CPU in decreasing its load.
- **Disadvantages of DMA Controller**
  - Direct Memory Access is a costly operation because of additional operations.
  - DMA suffers from Cache-Coherence Problems.
  - DMA Controller increases the overall cost of the system.
  - DMA Controller increases the complexity of the software.

Purpose of an Interrupt in Computer Organization

Read | Courses

Interrupt is the mechanism by which modules like I/O or memory may interrupt the normal processing by CPU. It may be either clicking a mouse, dragging a cursor, printing a document etc the case where interrupt is getting generated. **Why we require Interrupt?** External devices are comparatively slower than CPU. So if there is no interrupt CPU would waste a lot of time waiting for external devices to match its speed with that of CPU. This decreases the efficiency of CPU. Hence, interrupt is required to eliminate these limitations. **With Interrupt:**

1. Suppose CPU instructs printer to print a certain document.
2. While printer does its task, CPU engaged in executing other tasks.
3. When printer is done with its given work, it tells CPU that it has done with its work. (The word 'tells' here is interrupt which sends one message that printer has done its work successfully.).

Advantages:

- It increases the efficiency of CPU.
- It decreases the waiting time of CPU.
- Stops the wastage of instruction cycle.
- Enables multitasking by allowing the CPU to quickly switch between different processes.
- Simplifies input/output (I/O) operations by allowing devices to communicate directly with the CPU.

Disadvantages:

- CPU has to do a lot of work to handle interrupts, resume its previous execution of programs (in short, overhead required to handle the interrupt request.).
- Overhead required to handle the interrupt request can reduce the efficiency of the system.
- Interrupt storms can occur when there is high levels of interrupt activity.
- Priority inversion can occur when a low-priority task holds a resource needed by a higher-priority task.

Difference between Interrupt and Exception :

Interrupt	Exception
These are Hardware interrupts.	These are Software Interrupts.
Occurrences of hardware interrupts usually disable other hardware interrupts.	This is not a true case in terms of Exception.
These are asynchronous external requests for service (like keyboard or printer needs service).	These are synchronous internal requests for service based upon abnormal events (think of illegal instructions, illegal address, overflow etc).
Being asynchronous, interrupts can occur at any place in the program.	Being synchronous, exceptions occur when there is abnormal event in your program like, divide by zero or illegal memory location.
These are normal events and shouldn't interfere with the normal running of a computer.	These are abnormal events and often result in the termination of a program

Difference between Hardware Interrupt and Software Interrupt :

SR.NO.	Hardware Interrupt	Software Interrupt
1	Hardware interrupt is an interrupt generated from an external device or hardware.	Software interrupt is the interrupt that is generated by any internal system of the computer.
2	It do not increment the program counter.	It increment the program counter.
3	Hardware interrupt can be invoked with some external device such as request to start an I/O or occurrence of a hardware failure.	Software interrupt can be invoked with the help of INT instruction.
4	It has lowest priority than software interrupts	It has highest priority among all interrupts.
5	Hardware interrupt is triggered by external hardware and is considered one of the ways to communicate with the outside peripherals, hardware.	Software interrupt is triggered by software and considered one of the ways to communicate with kernel or to trigger system calls, especially during error or exception handling.
6	It is an asynchronous event.	It is synchronous event.
7	Hardware interrupts can be classified into two types they are: 1. Maskable Interrupt. 2. Non Maskable Interrupt.	Software interrupts can be classified into two types they are: 1. Normal Interrupts. 2. Exception
8	Keystroke depressions and mouse movements are examples of hardware interrupt.	All system calls are examples of software interrupts

Difference between maskable and nonmaskable interrupt :

SR.NO.	Maskable Interrupt	Non Maskable Interrupt
1	Maskable interrupt is a hardware Interrupt that can be disabled or ignored by the instructions of CPU.	A non-maskable interrupt is a hardware interrupt that cannot be disabled or ignored by the instructions of CPU.
2	When maskable interrupt occur, it can be handled after executing the current instruction.	When non-maskable interrupts occur, the current instructions and status are stored in stack for the CPU to handle the interrupt.
3	Maskable interrupts help to handle lower priority tasks.	Non-maskable interrupt help to handle higher priority tasks such as watchdog timer.
4	Maskable interrupts used to interface with peripheral device.	Non maskable interrupt used for emergency purpose e.g power failure, smoke detector etc .
5	In maskable interrupts, response time is high.	In non maskable interrupts, response time is low.
6	It may be vectored or non-vectored.	All are vectored interrupts.
7	Operation can be masked or made pending.	Operation Cannot be masked or made pending.
8	RST6.5, RST7.5, and RST5.5 of 8085 are some common examples of maskable Interrupts.	Trap of 8085 microprocessor is an example for non-maskable interrupt.

Difference between Programmed and Interrupt Initiated I/O :

Programmed I/O	Interrupt Initiated I/O
Data transfer is initiated by the means of instructions stored in the computer program. Whenever there is a request for I/O transfer the instructions are executed from the program.	The I/O transfer is initiated by the interrupt command issued to the CPU.
The CPU stays in the loop to know if the device is ready for transfer and has to continuously monitor the peripheral device.	There is no need for the CPU to stay in the loop as the interrupt command interrupts the CPU when the device is ready for data transfer.
This leads to the wastage of CPU cycles as CPU remains busy needlessly and thus the efficiency of system gets reduced.	The CPU cycles are not wasted as CPU continues with other work during this time and hence this method is more efficient.
CPU cannot do any work until the transfer is complete as it has to stay in the loop to continuously monitor the peripheral device.	CPU can do any other work until it is interrupted by the command indicating the readiness of device for data transfer
Its module is treated as a slow module.	Its module is faster than programmed I/O module.
It is quite easy to program and understand.	It can be tricky and complicated to understand if one uses low level language.
The performance of the system is severely degraded.	The performance of the system is enhanced to some extent.

Differences between System Bus and Address Bus

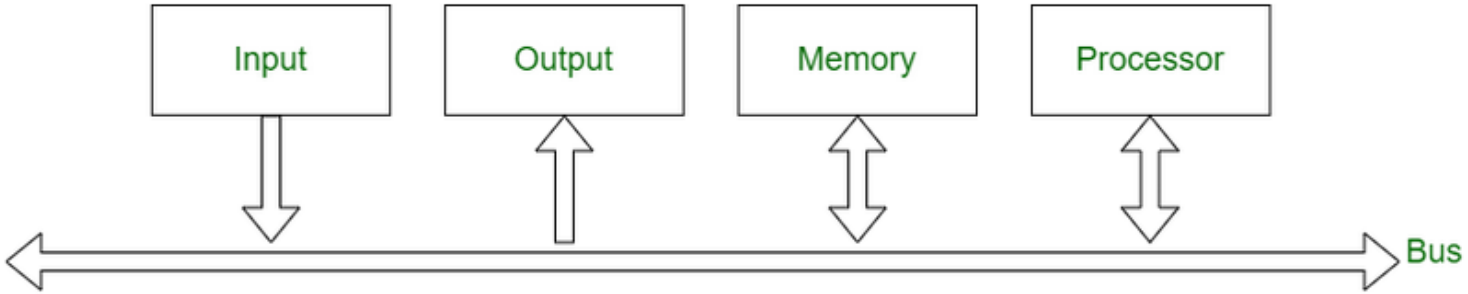
This tabular form summarizes the primary distinctions between the system bus and the address bus, emphasizing their respective functions, directionality, and roles in computer architecture.

System Bus	Address Bus
Handles overall communication within a computer system.	Specifically handles memory addressing.
Transfers data and instructions between CPU and other components.	Carries memory addresses from CPU to memory modules.
Acts as a comprehensive communication highway.	Functions as a targeted path for memory access.
Consists of multiple lines or channels for data transfer.	Typically a unidirectional bus.
Facilitates efficient communication between hardware components.	Enables CPU to specify memory locations.
Determines the width and bandwidth of the bus.	Determines the maximum memory capacity that can be addressed.
Connects CPU to memory, I/O devices, cache, etc.	Connects CPU to memory modules for data access.

Difference between Single Bus Structure and Double Bus Structure

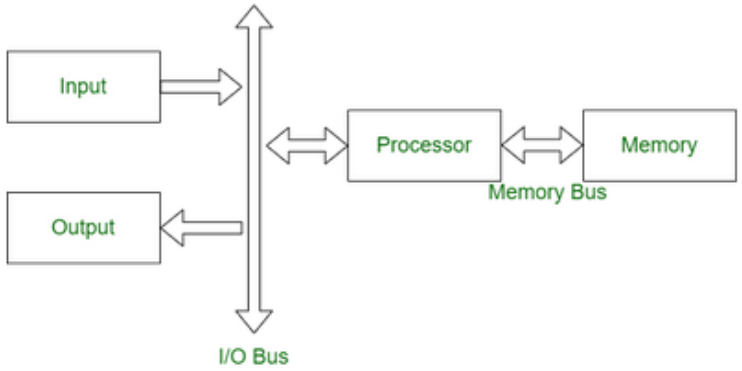
Read Courses

1. Single Bus Structure: In a single bus structure, one common bus is used to communicate between peripherals and microprocessors. It has disadvantages due to the use of one common bus.



Single Bus Structure

2. Double Bus Structure: In a double bus structure, one bus is used to fetch instructions while other is used to fetch data, required for execution. It is to overcome the bottleneck of a single bus structure.



Double Bus Structure

Differences between Data paths

Read | Courses

Prerequisite – ALU and Data Path In this section, we shall discuss the difference between data-paths. These data-paths are:

1. Single Cycle
2. Multiple Cycle
3. Pipeline

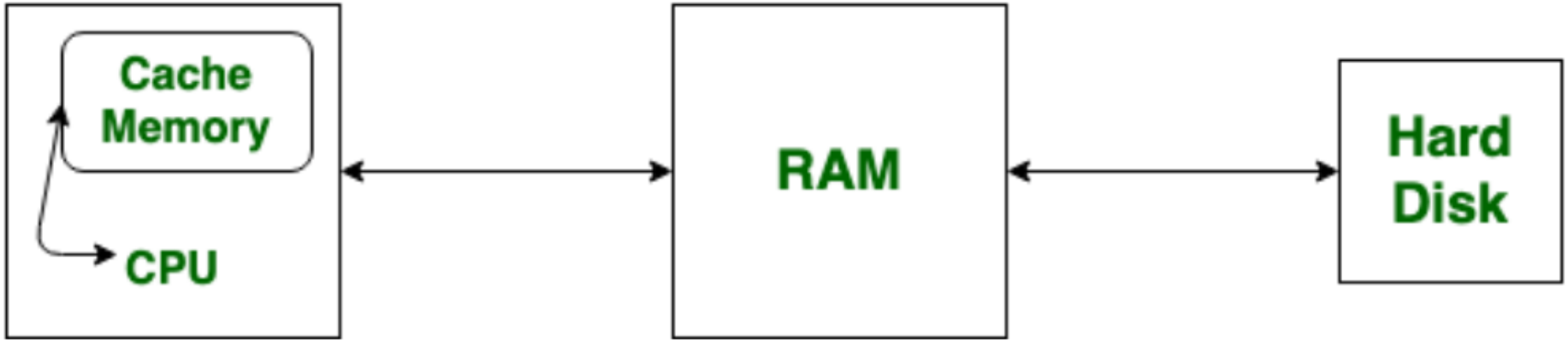
In single cycle clock cycle time is long enough for instruction while in multi cycle and pipeline clock cycle time is short enough for the instructions. Now, the difference among them are given below:

Single Cycle	Multiple Cycle	Pipeline
Single Cycle has one CPI (clock cycle per instruction).	Multiple cycle have variable number of CPI (Clock Cycle Per Instruction).	In pipeline, there is fixed number of CPI (Clock Cycle Per Instruction).
Single cycle have no instructions subdivided.	Multiple cycle have arbitrary number of instructions subdivided.	pipeline also have instructions subdivided one step per pipeline stage.
In Single cycle, there is executed one instruction at the same time.	In Multiple cycle, there is also executed one instruction at same time.	But in pipeline, many instructions are executed at the same time.
Extra registers are not used in single cycle.	But in multiple cycle, extra registers are used.	Extra registers are also used in pipeline.
In single cycle, clock cycle time is long.	In multiple cycle, clock cycle time is short.	In pipeline, clock cycle time is also short.
There is no overlapping in single cycle.	In multiple cycle, there is also no overlapping.	In pipeline, there is overlap instruction execution.

Difference between Virtual memory and Cache memory

Read | Courses

Cache Memory: Cache memory increases the accessing speed of CPU. It is not a technique but a memory unit i.e a storage device. In cache memory, recently used data is copied. Whenever the program is ready to be executed, it is fetched from main memory and then copied to the cache memory. But, if its copy is already present in the cache memory then the program is directly executed.



Virtual Memory: Virtual Memory increases the capacity of main memory. Virtual memory is not a storage unit, its a technique. In virtual memory, even such programs which have a larger size than the main memory are allowed to be executed.

Difference between Multiprogramming, multitasking, multithreading and multiprocessing

Read | Courses | Video

1. **Multiprogramming –** Multiprogramming is known as keeping multiple programs in the main memory at the same time ready for execution.
2. **Multiprocessing –** A computer using more than one CPU at a time.
3. **Multitasking –** Multitasking is nothing but multiprogramming with a Round-robin scheduling algorithm.
4. **Multithreading** is an extension of multitasking.

Feature	Multiprogramming	Multitasking	Multithreading	Multiprocessing
Definition	Running multiple programs on a single CPU	Running multiple tasks (applications) on a single CPU	Running multiple threads within a single task (application)	Running multiple processes on multiple CPUs (or cores)
Resource Sharing	Resources (CPU, memory) are shared among programs	Resources (CPU, memory) are shared among tasks	Resources (CPU, memory) are shared among threads	Each process has its own set of resources (CPU, memory)
Scheduling	Uses round-robin or priority-based scheduling to allocate CPU time to programs	Uses priority-based or time-slicing scheduling to allocate CPU time to tasks	Uses priority-based or time-slicing scheduling to allocate CPU time to threads	Each process can have its own scheduling algorithm
Memory Management	Each program has its own memory space	Each task has its own memory space	Threads share memory space within a task	Each process has its own memory space
Context Switching	Requires a context switch to switch between programs	Requires a context switch to switch between tasks	Requires a context switch to switch between threads	Requires a context switch to switch between processes
Inter-Process Communication (IPC)	Uses message passing or shared memory for IPC	Uses message passing or shared memory for IPC	Uses thread synchronization mechanisms (e.g., locks, semaphores) for IPC	Uses inter-process communication mechanisms (e.g., pipes, sockets) for IPC

Difference between IDE and SATA :

S.No.	IDE	SATA
01.	Integrated Drive Electronics in short referred as IDE.	Serial Advanced Technology Attachment in short referred as SATA.
02.	IDE is an interface standard for connection of storages devices such as Hard Disk Drives (HDD), Solid State Drives (SSD) and CD/DVD drives to the computer.	SATA is a computer bus interface or standard hardware interface which connects connecting hard drives, Solid State Drives (SSD) and CD/DVD drives to the computer.
03.	IDE has been introduced in the year 1986.	SATA has been introduced in the year 2003.
04.	It does not support hot plugging (i.e adding/removing component while the computer is running).	It supports hot plugging (i.e adding/removing component while the computer is running).
05.	In IDE data transfer speed ranges from 100 MB/s to 133 MB/s.	In SATA data transfer speed ranges from 150 MB/s for SATA I and 300 MB/s for SATA II.
06.	Here IDE cables are wide and can be up to 18 inches long.	Here SATA cable can be narrow and can be up to 39 inches long.
07.	Integrated Drive Electronics is an older standard.	Serial Advanced Technology Attachment is a newer standard.
08.	IDE drives are slower than SATA drives.	SATA drives are faster than IDE drives.
09.	It is a parallel connection.	It is a serial connection.
10.	The cables are difficult to install since there are jumpers.	The cables are easier to install since there are no jumpers.
11.	It has a single cable for data and power.	It has data and power cable on different sides.
12.	As it is wide, hence it will not allow free air flow.	As it is narrow, hence it will allow free air flow.

Understanding SAS, SATA, SCSI and ATA

For years the parallel interface has been widely used in storage systems. The need for increased bandwidth and flexibility in storage systems made the SCSI and ATA standards an inefficient option. A parallel interface is a channel capable of transferring data in parallel mode that is transmitting multiple bits simultaneously. Almost all personal computers come with at least one parallel interface. Common parallel interfaces include SCSI and ATA.

SCSI

Short for small computer system interface, a parallel interface standard used by Apple Macintosh computers, PCs and many UNIX systems for attaching peripheral devices to computers. Nearly all Apple Macintosh computers, excluding only the earliest Macs and the recent iMac, come with a SCSI port for attaching devices such as disk drives and printers. SCSI interfaces provide for data transmission rates (up to 80 megabytes per second).

ATA

Also known as IDE, ATA is a disk drive implementation that integrates the controller on the disk drive itself. ATA is used to connect hard disk drives, CD-ROM drives and similar peripherals and supports 8/16-bit interface that transfer up to 8.3MB/s for ATA-2 and up to 100MB/s (ATA-6).

So, what do parallel interfaces have to do with SAS (Serial Attached SCSI) and SATA (Serial ATA) drives? A lot, actually. It is the architectural limitations of the parallel interfaces that serial technologies like SAS and SATA address. In contrast to multiple parallel data stream, data is transmitted serially, that is in a single steam, by wrapping multiple bits into packets and it is able to move that single stream faster than parallel technology.

Serial Attached SCSI (SAS)

Abbreviated as SAS, Serial Attached SCSI, an evolution of parallel SCSI into a point-to-point serial peripheral interface in which controllers are linked directly to disk drives. SAS is a performance improvement over traditional SCSI because SAS enables multiple devices (up to 128) of different sizes and types to be connected simultaneously with thinner and longer cables; its full-duplex signal transmission supports 3.0Gb/s. In addition, SAS drives can be hot-plugged.

Serial ATA (SATA)

Often abbreviated as SATA, Serial ATA is an evolution of the Parallel ATA physical storage interface. Serial ATA is a serial link a single cable with a minimum of four wires creates a point-to-point connection between devices. Transfer rates for Serial ATA begin at 150MB/s.

Starting with SATA, it extends the capabilities of ATA and offers transfer rates starting at 150MB/s and, after years of development, has moved to the mainstream of disk interfaces. The successor the SCSI interface is SAS at speeds of up to 3Gb/s. Additionally, it also addresses parallel interface issues such as drive addressability and limitations on the number of device per port connection.

SAS devices can communicate with both SATA and SCSI devices (the backplanes of SAS devices are identical to SATA devices). A key difference between SCSI and SAS devices is the addition in SAS devices of two data ports, each of which resides in a different SAS domain. This enables complete failover redundancy. If one path fails, there is still communication along a separate and independent path.