3.2.3 DMA (Direct Memory Access)

Definition:

Direct Memory Access (DMA) is a method that allows I/O devices to transfer data to and from main memory without the direct involvement of the CPU. A specialized hardware component called a DMA controller manages these transfers, freeing up the CPU to perform other tasks.

Instead of the CPU managing every byte of data between devices and memory, a **DMA controller** handles the transfer, allowing the CPU to focus on other tasks.

Improve efficiency and reduce CPU load during I/O operations.

Example: Disk drives, graphics cards, network cards, sound cards, etc.,

Need for DMA

Normally, when a device needs to transfer data:

1. The device interrupts the CPU.
2. CPU reads/writes each byte or word from/to the device.
3. This wastes CPU time.

Buses Used in DMA:

Bus Type	Role in DMA
Address Bus	Carries the memory address where data will be read from or written to.

Bus Type	Role in DMA
Data Bus	Transfers the actual data between memory and the I/O device.
Control Bus	Sends control signals like DMA request, acknowledge, and interrupt signals.
Internal Bus	Connects components within the DMA controller for coordination.

Steps involved in bus request:

- The **DMA controller** sends a **Bus Request** to the CPU.
- The CPU responds with a **Bus Grant**, temporarily relinquishing control.
- The DMA controller becomes the **bus master**, handling the transfer independently.
- Once complete, it sends an **interrupt** to notify the CPU.

Components:

- **DMA Controller**: Manages the transfer process.
- **Registers**:
  - **Address Register**: Specifies memory location.
  - **Word Count Register**: Indicates how much data to transfer.
  - **Control Register**: Defines transfer mode and direction.

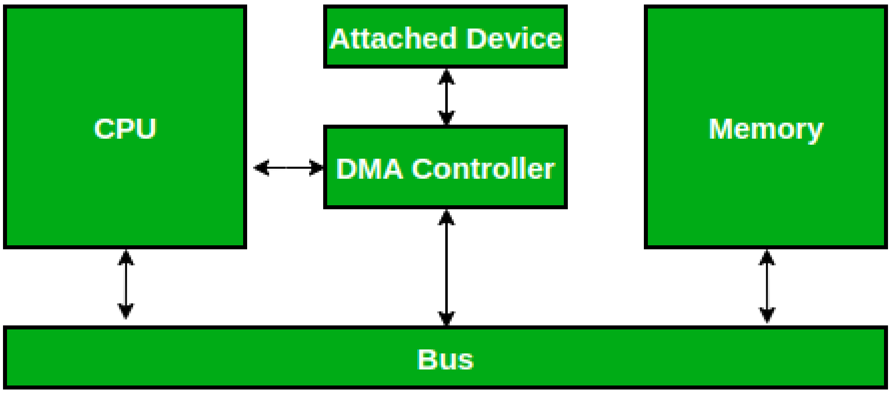
☞ **With DMA**, the CPU sets up the transfer once and then is **free to do other tasks**.

Step-by-Step DMA Transfer:

1. **CPU initializes DMA Controller** with:
   - Source address (e.g., device).
   - Destination address (e.g., RAM).

  - Size of data to transfer.

2. **DMA Controller** takes control of the bus to perform the transfer. hen done, DMA **interrupts the CPU** to indicate the transfer is complete.



Direct Memory Access

Explanation of the above steps:

Initialization: The CPU sets up the DMA controller with the memory address, data size, and direction of transfer.

Transfer: The DMA controller takes control of the system bus and moves data between the device and memory.

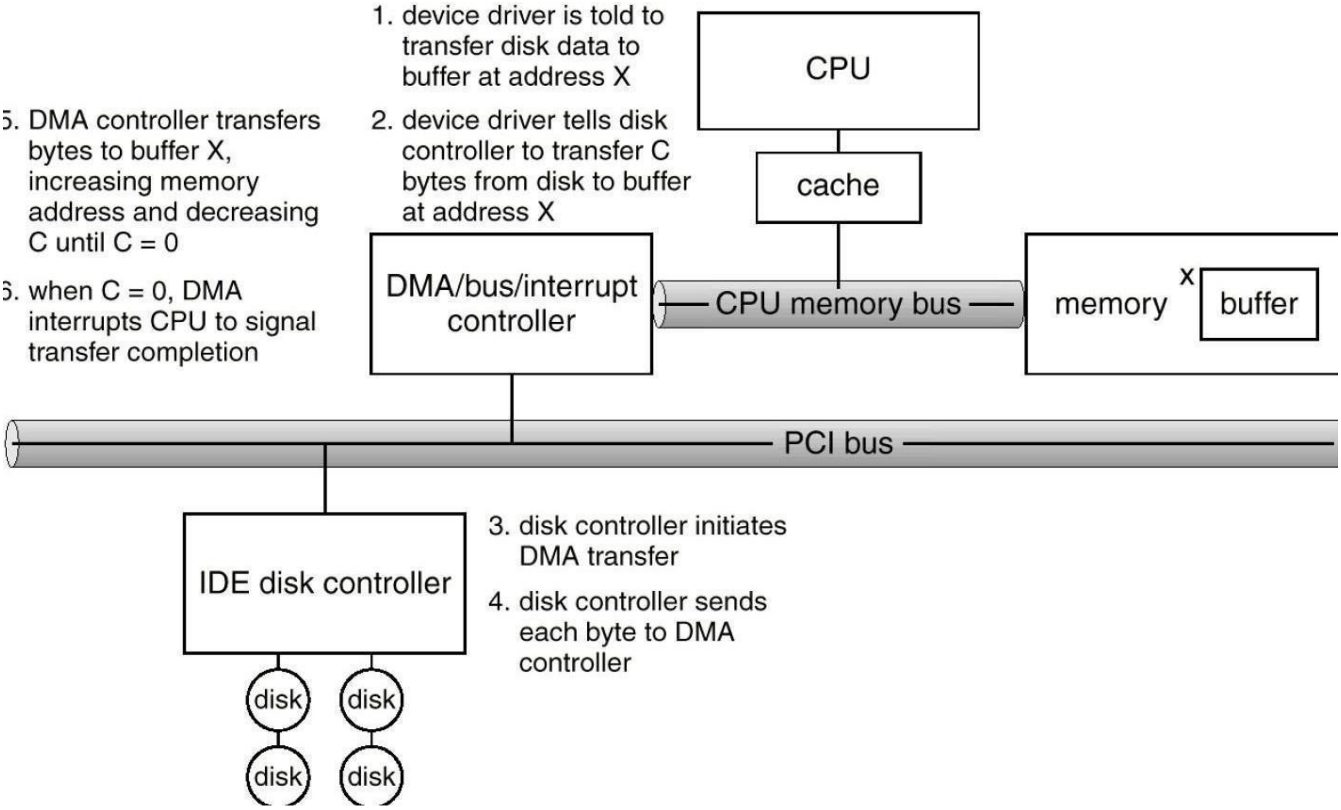
Completion: Once done, the DMA controller sends an interrupt to the CPU to signal about the completion.

Working of DMA

When the processor wishes to read or write a block of data, it issues a command to the DMA module by sending to the DMA module the following information:

- Whether a read or write is requested, using the read or write control line between the processor and the DMA module.

        - ▪ The address of the I/O device involved, communicated on the data lines.
        - ▪ The starting location in memory to read from or write to, communicated on the data lines and stored by the DMA module in its address register.
        - ▪ The number of words to be read or written, again communicated via the data lines and stored in the data count register.
    - o Then the processor continues with other work. It has delegated this I/O operation to the DMA module.
- o The DMA module transfers the entire block of data, one word at a time, directly to or from memory, without going through the processor. When the transfer is complete, the DMA module sends an interrupt signal to the processor. Thus, the processor is involved only at the beginning and end of the transfer.


1. device driver is told to transfer disk data to buffer at address X
CPU
5. DMA controller transfers bytes to buffer X, increasing memory address and decreasing C until C = 0
2. device driver tells disk controller to transfer C bytes from disk to buffer at address X
cache
6. when C = 0, DMA interrupts CPU to signal transfer completion
DMA/bus/interrupt controller
CPU memory bus
memory
X
buffer
PCI bus
IDE disk controller
3. disk controller initiates DMA transfer
4. disk controller sends each byte to DMA controller
disk
disk
disk
disk

DMA Modes

Mode	Description
Burst Mode	DMA transfers all the words/bytes of the block without releasing the bus.
Cycle Stealing	DMA takes control of the bus for one cycle at a time ie, when the requested bus is granted DMA transfer one words/bytes and release the bus. Then request the CPU for bus, this process is repeated until the whole block is transferred.
Transparent Mode	DMA transfers only when CPU is not using the bus. Here DMA monitors bus activity. When CPU needs the bus DMA release it immediately.
Block Transfer	Transfers a block of data with a single DMA request. (DMA locks the bus and moves the whole block in one shot)

Types of DMA

Type	Description
Single-Ended DMA	The DMA controller connects directly to **one peripheral device** and the **main memory**.
Dual-Ended DMA	The DMA controller connects to **both source and destination devices** (e.g., from one I/O device to another or device to memory).
Arbitrated DMA	Used when **multiple DMA devices** share the same bus.

Type	Description
	A **bus arbitration** process decides which DMA device gets control of the bus at a given time, preventing conflicts. Found in systems with many peripherals (e.g., network cards, storage devices) that all need DMA access.
Interleaved DMA	The DMA reads from one address and writes to another **in an alternating cycle**—read in one bus cycle, write in the next.

Advantage:

- **Reduced CPU overhead**: Frees CPU for other tasks.
- **Faster data transfer**: Especially for large blocks of data.
- **Improved multitasking**: CPU can handle other processes while DMA operates.
- **Lower latency**: Ideal for real-time and multimedia applications.

Disadvantage:

- **Cache Coherence Issues**: DMA can bypass CPU cache, leading to data mismatches.
- **System Complexity**: Requires extra hardware and careful setup.
- **Bus Contention**: DMA and CPU may compete for the system bus, slowing performance.
- **Higher Cost**: Adds to the cost due to additional controller hardware.
- **Limited CPU Control**: CPU has less control during DMA transfers.
- **Interrupt Overhead**: DMA still generates interrupts, which can affect system speed.