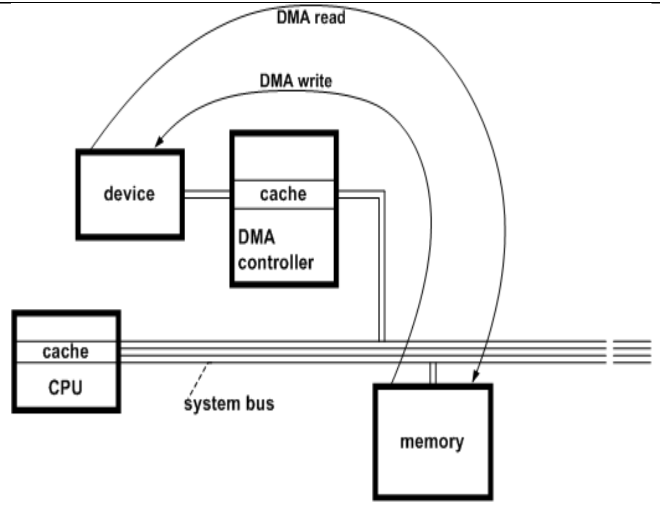
Direct Memory Access

Both the methods programmed I/O and Interrupt-driven I/O require the active intervention of the processor to transfer data between memory and the I/O module, and any data transfer must transverse a path through the processor. Thus both these forms of I/O suffer from two inherent drawbacks.

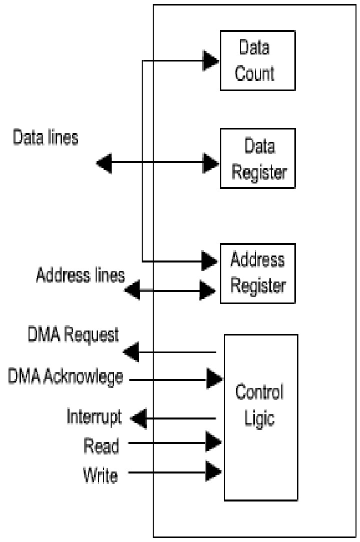
- The I/O transfer rate is limited by the speed with which the processor can test and service a device.
- The processor is tied up in managing an I/O transfer; a number of instructions must be executed for each I/O transfer.

To transfer large block of data at high speed, a special control unit may be provided to allow transfer of a block of data directly between an external device and the main memory, without continuous intervention by the processor. This approach is called direct memory access or DMA.

DMA transfers are performed by a control circuit associated with the I/O device and this circuit is referred as DMA controller. The DMA controller allows direct data transfer between the device and the main memory without involving the processor.



Functionality of DMA.



Block diagram of a DMA controller

DMA transfers are performed by a control circuit associated with the I/O device and this circuit is referred as DMA controller. The DMA controller allows direct data transfer between the device and the main memory without involving the processor.

To transfer data between memory and I/O devices, DMA controller takes over the control of the system from the processor and transfer of data take place over the system bus.

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 devise involved, communicated on the data lines.
- The starting location in the memory to read from or write to, communicated on 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.

The processor then continues with other works. It has delegated this I/O operation to the DMA module.

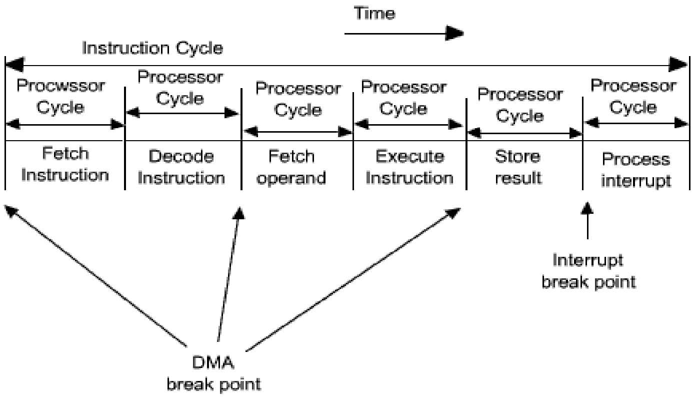
The DMA module checks the status of the I/O devise whose address is communicated to DMA controller by the processor. If the specified I/O devise is ready for data transfer, then DMA module generates the DMA request to the processor. Then the processor indicates the release of the system bus through DMA acknowledge.

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 completed, the DMA module sends an interrupt signal to the processor. After receiving the interrupt signal, processor takes over the system bus.
Thus the processor is involved only at the beginning and end of the transfer. During that time the processor is suspended.

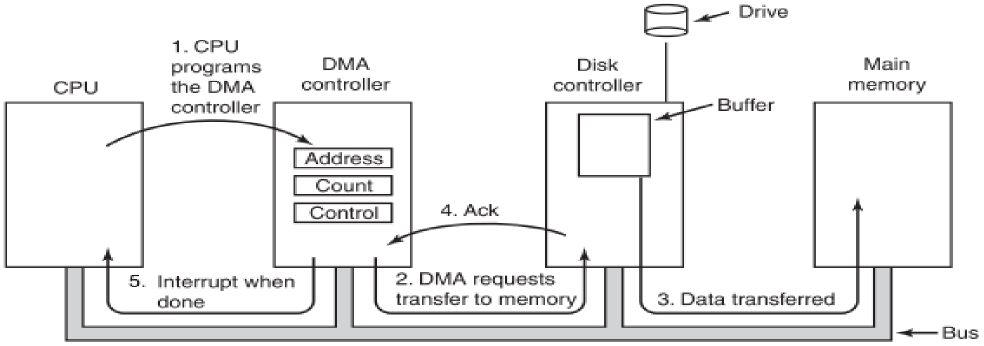
It is not required to complete the current instruction to suspend the processor. The processor may be suspended just after the completion of the current bus cycle. On the other hand, the processor can be suspended just before the need of the system bus by the processor, because DMA controller is going to use the system bus, it will not use the processor.

The point where in the instruction cycle the processor may be suspended shown in the Figure:-



When the processor is suspended, then the DMA module transfer one word and return control to the processor.

Summary:-



Types of Data transfers: The DMA Controller has several options available for the transfer of data. They are: -

1) Interleaved mode:-
An alternate half cycle i.e. half cycle DMA+ half cycle processor.

2) Block mode: DMA returns the bus after block transfer.

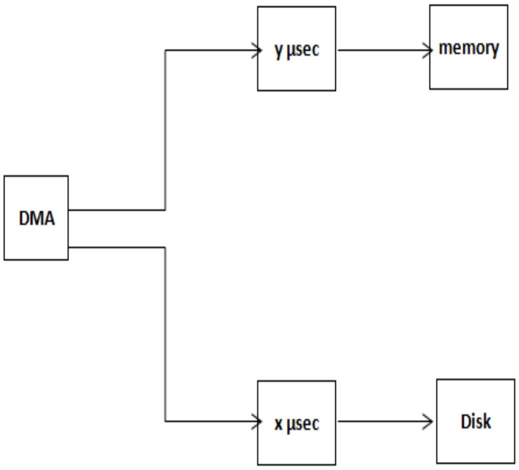
3) Cycle Stealing mode: DMA returns the bus after a word is transfer. The DMA effectively steals cycles from the processor in order to transfer the byte, so single byte transfer is also known as cycle stealing.
The concept of cycle stealing is like this:
1. Buffer the byte into the buffer
2. Inform the CPU that the device has 1 byte to transfer (i.e. bus grant request)
3. Transfer the byte (at system bus speed)
4. Release the control of the bus back to CPU.
Before moving on transfer next byte of data, device performs step 1 again so that bus isn't tied up and the transfer won't depend upon the transfer rate of device.
So, for 1 byte of transfer of data, time taken by using cycle stealing mode (T).
= time required for bus grant + 1 bus cycle to transfer data + time required to release the bus, it will be N x T.

In cycle stealing mode we always follow pipelining concept that when one byte is getting transferred then Device is parallel preparing the next byte. "The fraction of CPU time to the data transfer time" if asked then cycle stealing mode is used.
Where,
X µsec =data transfer time or preparation time (words/block)
Y µsec =memory cycle time or cycle time or transfer time (words/block)

% CPU idle (Blocked) =$\frac{Y}{X}$ x 100

% CPU busy = $\frac{X}{Y}$ x 100

Or,
% CPU busy = (100-% CPU Busy)

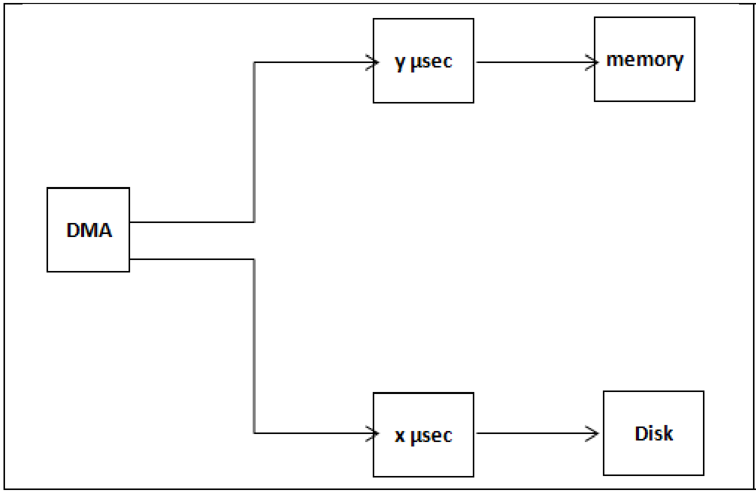


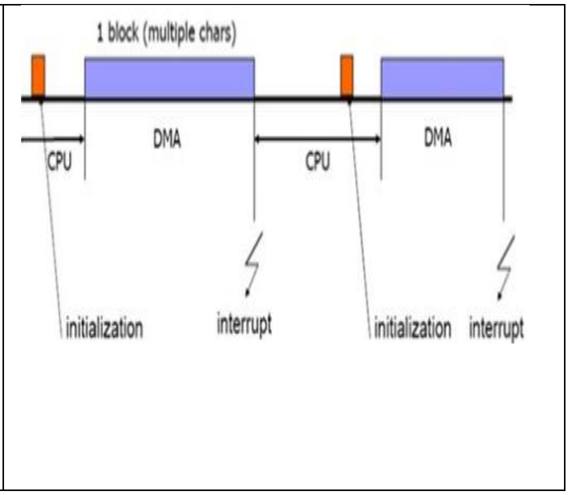
4) Burst mode: DMA returns the bus after complete data transfer. A register is used as a byte count, being decremented for each byte transfer, and upon the byte count reaching zero, the DMAC will release the bus. When the DMAC operates in burst mode, the CPU is halted for the duration of the data transfer.

The concept of burst mode in DMA is like this:

1. Bus grant request time
2. Transfer the entire block of data at transfer rate of device because the device is usually slow than the speed at which the data can be transferred to CPU
3. Release the control of the bus back to CPU

So, total time taken to transfer the N bytes
= Bus grant request time + (N) * (memory transfer rate) + Bus release control time.





Where,
X µsec =data transfer time or preparation time (words/block)
Y µsec =memory cycle time or cycle time or transfer time (words/block)

% CPU idle (Blocked) $=\frac{Y}{X+Y}$ x 100

% CPU busy = $\frac{X}{X+Y}$ x 100

- It should be used only when independent data quantum transfers take place. E.g. Cycle of prepare and transfer.

- Burst mode can be used if this line "**constantly transferring data to memory using DMA"** or **"the disk is actively transferring 100% of the time"** appears in the question.