CHAPTER FOUR

MEMORY ORGANIZATION

4.1 Introduction

The memory unit is an essential component in any digital computer since it is needed for storing programs and data. The memory unit that communicates directly with the CPU is called the *main memory*. Devices that provide backup storage are called *auxiliary memory*. They are used for storing system programs, large data files, and other backup information. Only programs and data currently needed by the processor reside in main memory. All other information is stored in auxiliary memory and transferred to main memory when needed.

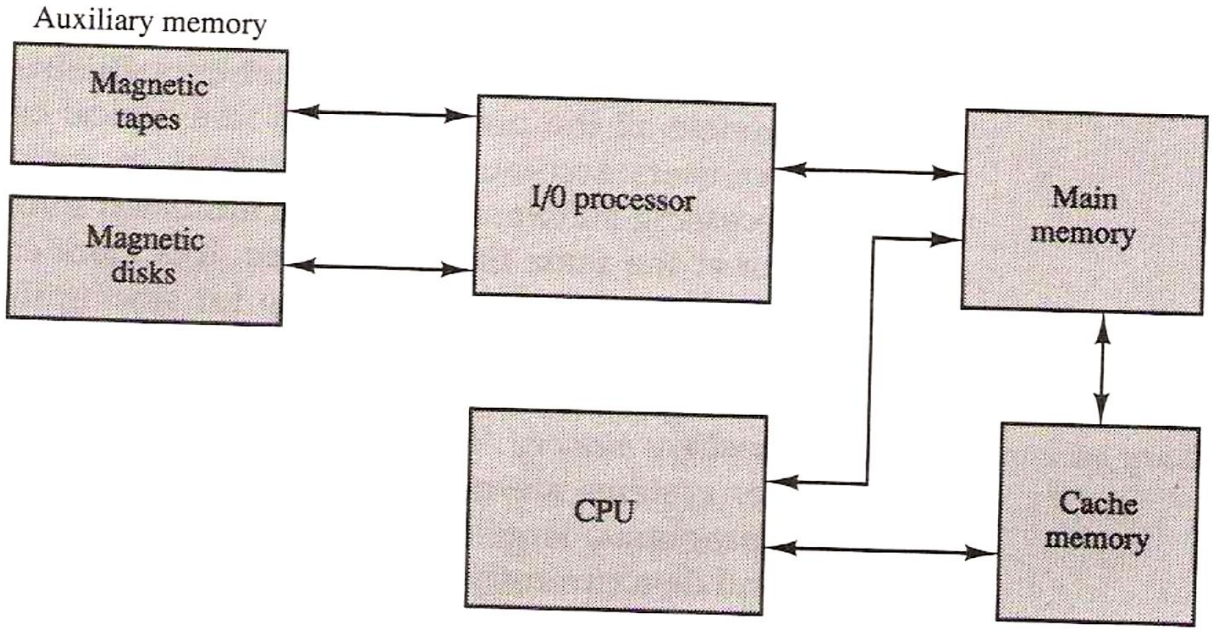
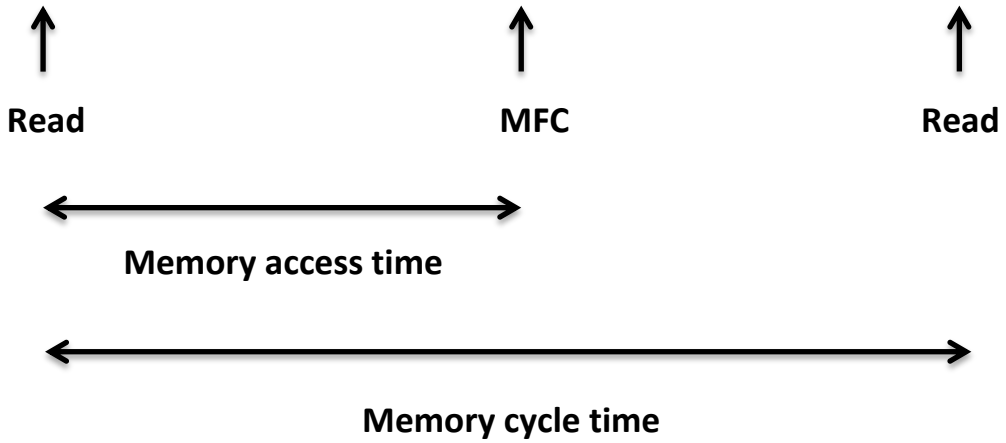


Figure 7.1: Memory hierarchy in a computer system.

4.2 Main Memory

Memory access time it's a time required between the initiation of an operation and the completion of that operation (e.g. the time required between READ and MFC)

Memory cycle time this is the minimum time delay required between the initiations of two independent memory operations (e.g. two successive READ operations). The cycle time is usually larger than the access time.



Solution for the speed problem (which arises between a CPU and a main memory) are:

1- Using cache memory
2- Dividing the memory into a number of memory modules

4.2.1 Memory System Consideration

The choice of a RAM chip for a given application depends on several factors. Foremost among these are the speed, power dissipation, and size of the chip.

Bipolar memories are generally used whenever very fact operation is the primary requirement. For example, this is usually the case in microprogram memories and cache memories. High power dissipation in bipolar circuit makes it difficult to achieve high densities. Hence, a

memory implemented with bipolar memory chips requires a relatively large number of chips.

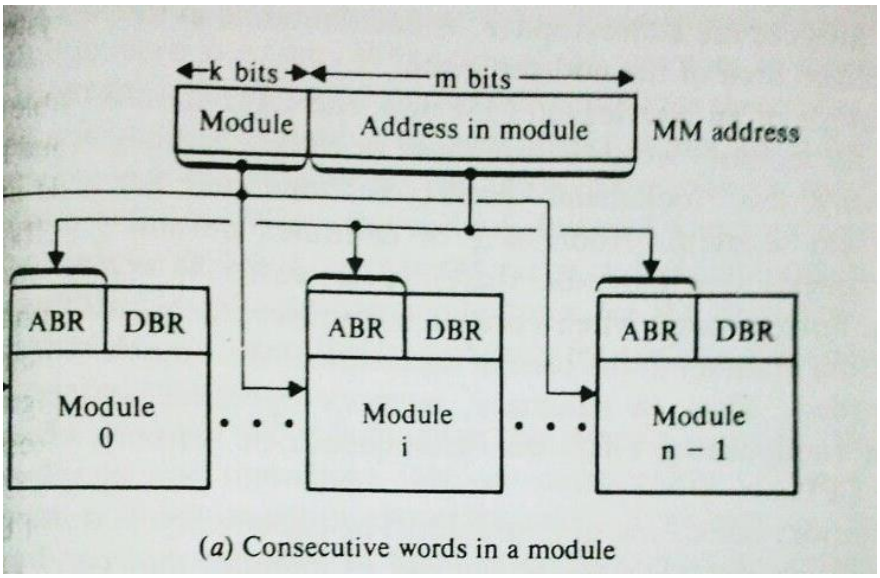
Dynamic MOS memory is the predominant technology used in computer main memories. The high densities achievable in this technology make the implementation of large memories economically attractive.

Static MOS memories chips have higher densities and slightly longer access time than bipolar chips. They have lower densities than dynamic memories. They are also easier to use because they do not require refreshing. A static memory is particularly well suited to applications in which the occasional increase in access delay caused by refresh cycles cannot be tolerated.

4.2.2 Multiple-Model Memory and Interleaving

If main memory is structured as a collection of physically separates, each with its own address buffer register (ABR) and data buffer register (DBR), it is possible for more than one module to be performing Read or Write operations at any given time. The average rate of transmission of word to and from the total main memory system can thus be increased.

The way in which individual addresses are distributed over the modules is a critical factor in determining the average number of modules that can be kept busy as computations proceed. Two methods of address layout are indicated in figure 7.2.(A &B) In first case, the MM address generated by the CPU is decoded as shown in figure 7.2a. The high-order *k* bits of the address name one of *n* modules, and the low-order *m* bits name a particular word in the module. If the CPU issues Read requests to consecutive locations, then only one module is kept busy by the CPU. However, devices with direct memory access (DMA) ability may be operating in other memory modules.

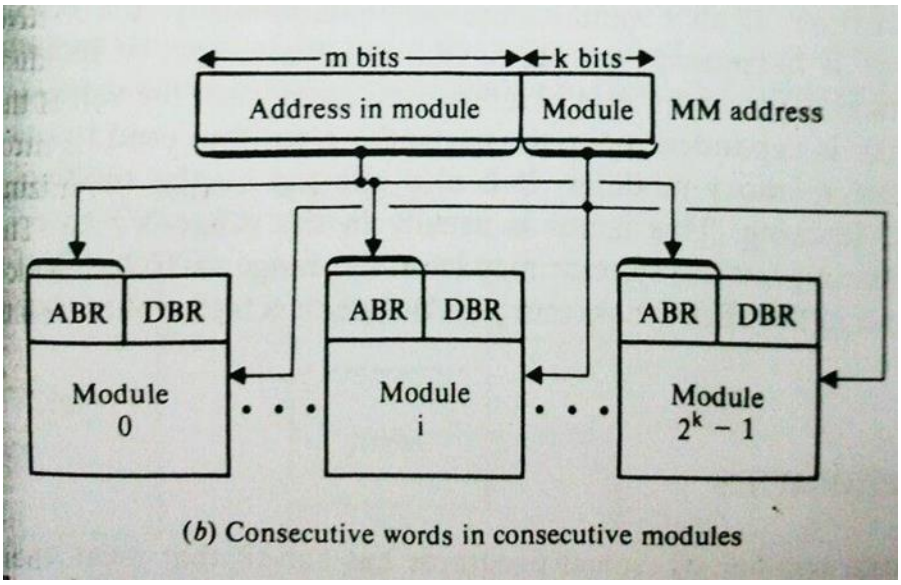


0000	10
0001	20
0010	30
0011	40
0100	50
0101	60
0110	70
0111	80
1000	90
1001	100
1010	110
1011	120

	Module - 00		Module - 01		Module - 10
00	10	00	50	00	90
01	20	01	60	01	100
10	30	10	70	10	110
11	40	11	80	11	120

Figure 7.2 A: Addressing multiple-module memory systems(Consecutive word per module)

The second and more effective way of addressing the modules is shown in figure 7.2b. It is called memory interleaving. A module is selected by the low-order k bits of the MM address, and the high-order m bits name a location within that module. Therefore, consecutive addresses are located in successive modules, thus, any component of the system that generates requests for access to consecutive MM locations can keep a number of modules busy at any one time.



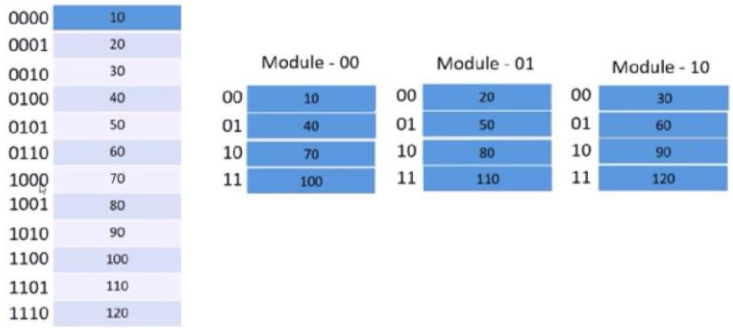
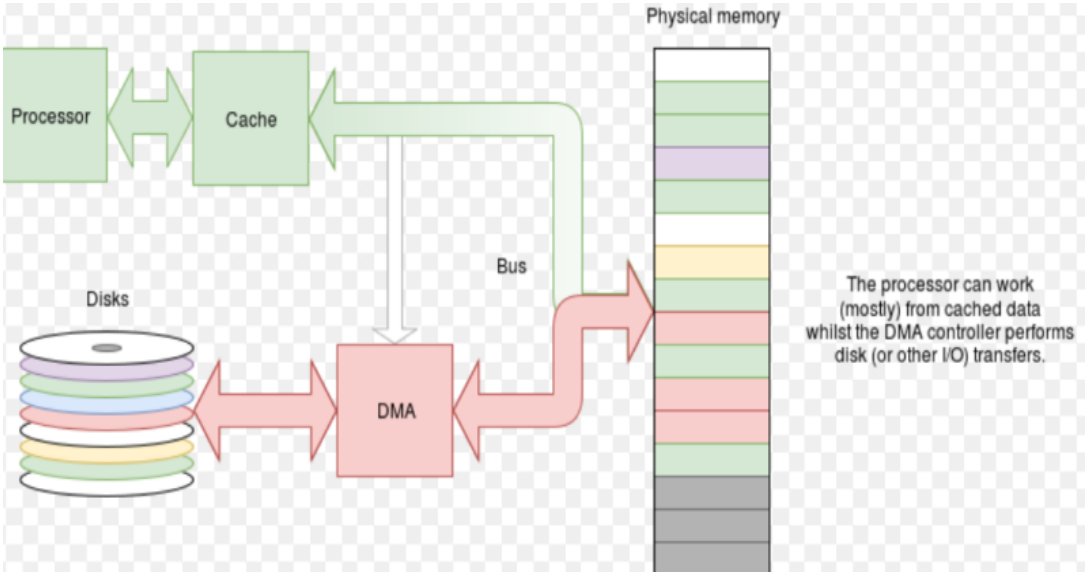


Figure 7.2: Addressing multiple-module memory systems interleave (consecutive word in consecutive modules).

A failure in any module in second method affects all areas of the address space. From the other hand, interleaving memory structure faster than first structure, as it allows parallel access , provide higher average utilization of the memory hence it is faster. A failed module in the first system affects only a localized area of the address space.

4.3 Direct Memory Access Controller (DMAC)

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 (processor) known as a DMA controller (DMAC).



In older computers, four DMA channels were numbered 0, 1, 2 and 3. When the 16-bit industry standard architecture (ISA) expansion bus was introduced, channels 5, 6 and 7 were added. Each DMA transfers approximately 2 MB of data per second.

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***. All four system resources rely on certain lines on a bus. Some lines on the bus are used for IRQs, some for addresses (the I/O addresses and the memory address) and some for DMA channels.

With DMA, the CPU can process other tasks while data transfer is being performed. The transfer of data is first initiated by the CPU. During the transfer of data between the DMA channel and I/O device, the CPU performs other tasks. When the data transfer is complete, the CPU receives an interrupt request from the DMA controller.

4.3.1 DMA Function

DMA involves an additional module on the system bus. The DMA module (Figure 1) is capable of mimicking the processor and, indeed, of taking over control of the system from the processor.

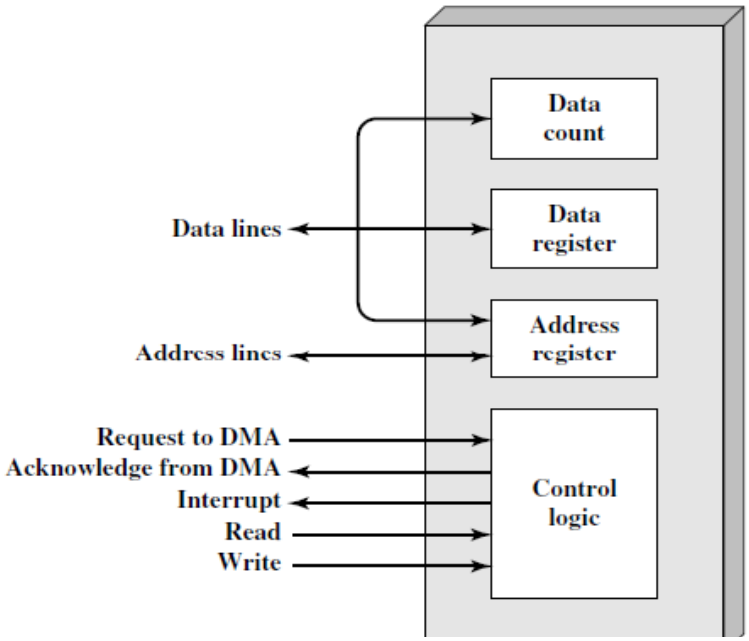


Figure 1: Typical DMA Block Diagram.

It needs to do this to transfer data to and from memory over the system bus. For this purpose, the DMA module must use the **bus** only when the processor does not need it. This technique is more common and is referred to as ***cycle stealing***, because the DMA module in effect steals a bus cycle.

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 address 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.

The processor then continues with other work. It has delegated this I/O operation to the DMA module. 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 (Figure 2).

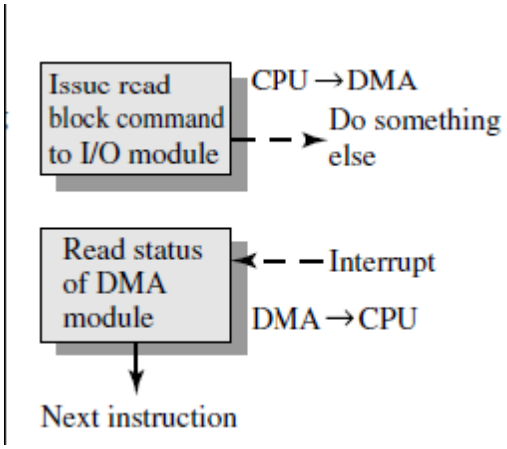


Figure 2: Direct Memory Access.

Figure 3 shows where in the instruction cycle the processor may be suspended. In each case, the processor is suspended just before it needs to use the bus. The DMA module then transfers one word and returns control to the processor. ***Note that this is not an interrupt***; the processor does not save a context and do something else. Rather, the processor pauses for one bus cycle. The overall effect is to cause the processor to execute more slowly. Nevertheless, for a multiple-word I/O transfer, DMA is far more efficient than interrupt-driven or programmed I/O.

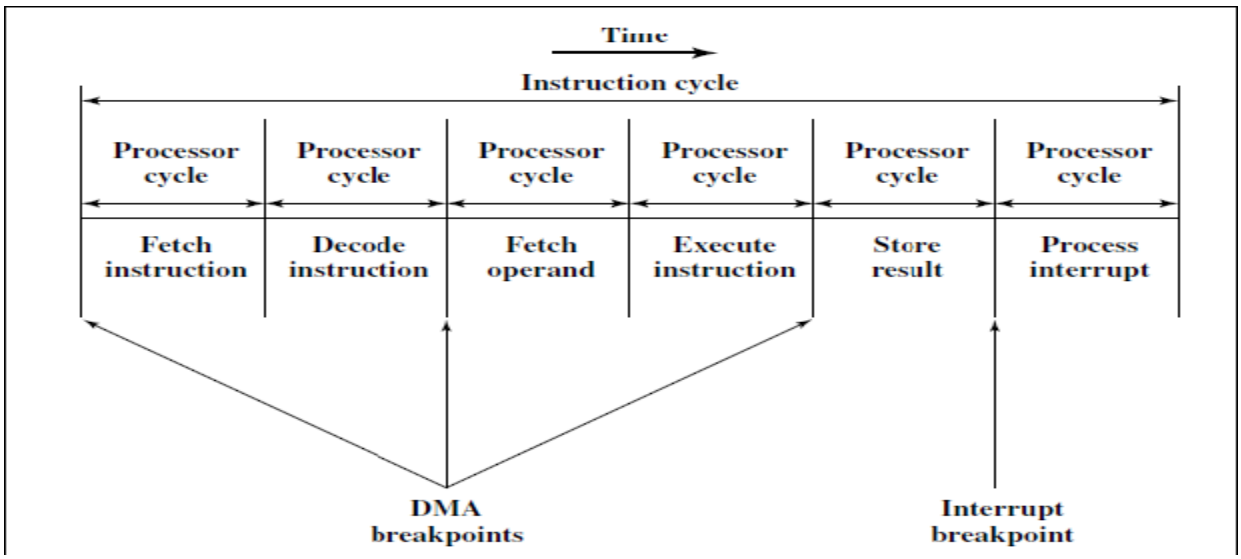


Figure 3: DMA and Interrupt Breakpoints during an Instruction Cycle.

4.3.2 Intel 8237A DMA Controller

The Intel 8237A DMA controller interfaces to the 80x86 family of processors and to DRAM memory to provide a DMA capability. Figure 4 indicates the location of the DMA module.

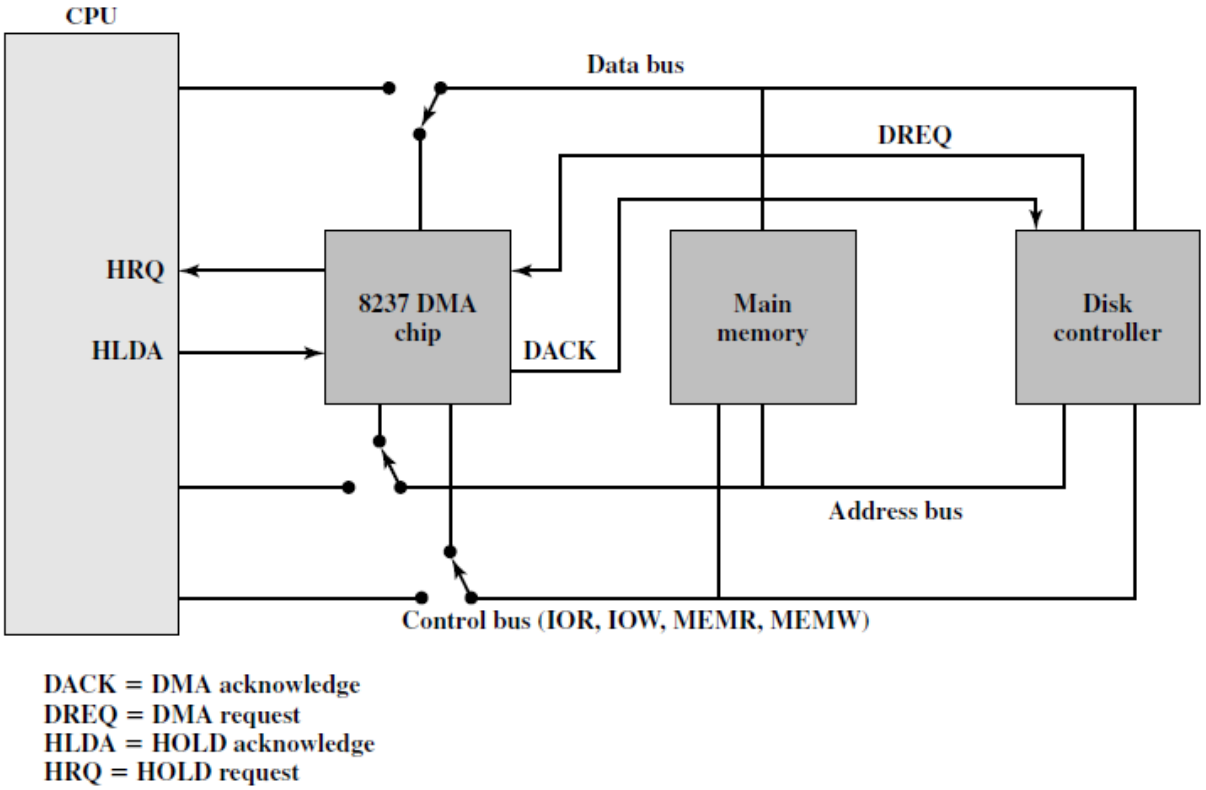


Figure 4: Intel 8237 Usage of system bus.

When the DMA module needs to use the system buses (data, address, and control) to transfer data, it sends a signal called HOLD to the processor. The processor responds with the HLDA (hold acknowledge) signal, indicating that the DMA module can use the buses. For example, if the DMA module is to transfer a block of data from memory to disk, it will do the following:

1. The peripheral device (such as the disk controller) will request the service of

DMA by pulling DREQ (DMA request) high.

2. The DMA will put a high on its HRQ (hold request), signaling the CPU through its HOLD pin that it needs to use the buses.

3. The CPU will finish the present bus cycle (not necessarily the present instruction) and respond to the DMA request by putting high on its HDLA (hold acknowledge), thus telling the 8237 DMA that it can go ahead and use the buses to perform its task. HOLD must remain active high as long as DMA is performing its task.

4. DMA will activate DACK (DMA acknowledge), which tells the peripheral device that it will start to transfer the data.

5. DMA starts to transfer the data from memory to peripheral by putting the address of the first byte of the block on the address bus and activating MEMR, thereby reading the byte from memory into the data bus; it then activates IOW to write it to the peripheral. Then DMA decrements the counter and increments the address pointer and repeats this process until the count reaches zero and the task is finished.

6. After the DMA has finished its job it will deactivate HRQ, signaling the CPU that it can regain control over its buses.

4.3.3 Basic DMA Operation

- The DMA I/O technique provides direct access to the memory while the processor is temporarily disabled.
- A DMAC temporarily borrows the address bus, data bus, and the control bus from the microprocessor and transfer the data bus directly between an I/O port and a sequential of memory locations.
- The DMA transfer is also used to do high speed memory to memory transfers.
- Two control signals are used to request and acknowledge a DMA transfer in the microprocessor based system.

1. Principles

❖ Third-party DMA

Standard DMA, also called ***third-party DMA***, uses a DMA controller. A DMA controller can generate ***memory addresses*** and initiate memory read or write cycles. It contains several ***hardware registers*** that can be written and read by the CPU. These include a memory address register, a byte count register, and one or more control registers. Depending on what features the DMA controller provides, these control registers might specify some combination of the source, the destination, the direction of the transfer (reading from the I/O device or writing to the I/O device), the size of the transfer unit, and/or the number of bytes to transfer in one burst.

❖ Bus mastering

In a bus mastering system, also known as a first-party DMA system, the CPU and peripherals can each be granted control of the memory bus. Where a peripheral can become bus master, it can directly write to system memory without involvement of the CPU, providing memory address and control signals as required. Some measure must be provided to put the processor into a hold condition so that bus contention does not occur.

❖ Transfer types

DMA transfers can either transfer one byte at a time or all at once in burst mode. If they transfer a byte at a time, this can allow the CPU to access memory on alternate bus cycles – this is called cycle stealing since the CPU and either the DMA controller or the bus master contend for memory access. In burst mode DMA, the CPU can be put on hold while the DMA transfer occurs and a full block of possibly hundreds or thousands of bytes can be moved. When memory cycles are much faster than processor cycles, an interleaved DMA cycle is possible, where the DMA controller uses memory while the CPU cannot.

2. Types of DMA Controllers:

a) Flow – Through (Explicit) DMAC: the data transferred between memory and I/O interface pass through the DMAC. The DMAC first reads data into an internal register and then writes it to the destination.

b) Fly-by (Implicit) DMAC: the data don't pass through the DMAC. After the DMAC gains access to the bus it puts the source (or destination) address and other control signals (R/W, VMA, etc.) out. It activates the memory and the I/O interface at the same time. So it initiates a read and a write cycle simultaneously. The data is read from the source address, and written to the destination, in one clock cycle. Therefore the fly-by technique can transfer data faster than the flow through. The fly-by DMAC can only transfer data between an I/O port and a memory address, but not between two I/O ports or two memory locations.

3. Operation Modes

- Burst mode

An entire block of data is transferred in one contiguous sequence. Once the DMA controller is granted access to the system bus by the CPU, it transfers all bytes of data in the data block before releasing control of the system buses back to the CPU, but renders the CPU inactive for relatively long periods of time. The mode is also called "***Block Transfer Mode***".

- Cycle stealing mode

The *cycle stealing mode* is used in systems in which the CPU should not be disabled for the length of time needed for burst transfer modes. In the cycle stealing mode, the DMA controller obtains access to the system bus the same way as in burst mode, using BR (***Bus Request***) and BG (***Bus***

Grant) signals, which are the two signals controlling the interface between the CPU and the DMA controller.

However, in cycle stealing mode, after one byte of data transfer, the control of the system bus is deserted to the CPU via BG. It is then continually requested again via BR, transferring one byte of data per request, until the entire block of data has been transferred. ***By continually obtaining and releasing the control of the system bus, the DMA controller essentially interleaves instruction and data transfers.*** The CPU processes an instruction, then the DMA controller transfers one data value, and so on. On the one hand, the data block is not transferred as quickly in cycle stealing mode as in burst mode, but on the other hand the CPU is not idled for as long as in burst mode. ***Cycle stealing mode is useful for controllers that monitor data in real time***.

- Transparent (Hidden) mode

Transparent mode takes the most time to transfer a block of data, yet it is also the most efficient mode in terms of overall system performance. In transparent mode, the DMA controller transfers data only when the CPU is performing operations that do not use the system buses.

The primary advantage of transparent mode is that the CPU never stops executing its programs and the DMA transfer is free in terms of time, while the disadvantage is that the hardware needs to determine when the CPU is not using the system buses, which can be complex.

4. Advantage and Disadvantage

- DMA allows a peripheral device to read / write to memory without going through the CPU.
- DMA allows for faster processing since the processor can be working on something else while the peripheral can be populating memory.

- DMA transfer requires a DMAC to carry out the operation, hence cost of the system increases.
- Cash coherence problems.