Programme : Bachelor of Science (Second Year)
Subject : Computer Science
Semester : IV
Paper Code : CSC104
Paper Title : Computer Organization and Operating Systems
Title of the Unit : Input – Output Organization
Module Name : Programmed I/O, Interrupt-Driven I/O, Direct Memory Access
Module Number : 17
Presenter : Milan Gaonkar
Assistant Professor Govt. College of Arts,Commerce & Science
Khandola , Marcela-Goa.

Programmed I/O Mode

In programmed I/O mode of data transfer the operations are the results of I/O instructions which is a part of computer program. Each data transfer is initiated by an instruction in the program.Normally the transfer is from a CPU register to peripheral device or vice-versa.

Once a data transfer is initiated, the CPU is required to monitor the interface to see when a transfer can be made.It is up to the programmed instructions executed in the CPU to keep close tabs on everything that is taking place in the interface unit and the I/O device.

In this technique CPU is responsible for executing data from the memory for output and storing data in memory for executing of Programmed I/O as shown in Flowchart-:

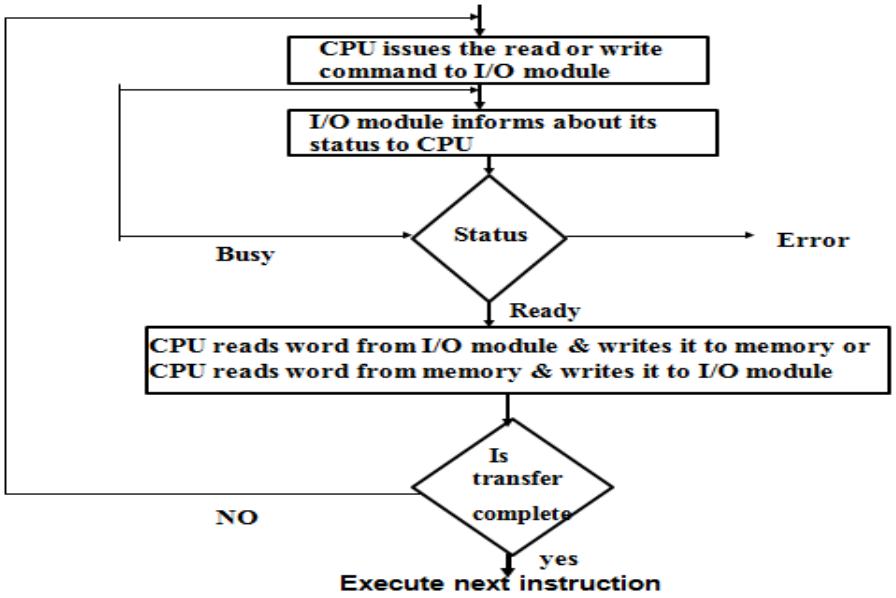


Fig :Programmed I/O

Drawback ofthe Programmed I/O

The main drawback of the Program Initiated I/O was that the CPU has to monitor the units all the times when the program is executing. Thus the CPU stays in a program loop until the I/O unit indicates that it is ready for data transfer. This is a time consuming process as a lot of CPU time is wasted.

To remove this problem an Interrupt facility and special commands are used.

Interrupt-Initiated I/O

In Interrupt-initiated I/O an interrupt facility is used to inform the device about the start and end of transfer. In the meantime the CPU executes other program. When the interface determines that the device is ready for data transfer it generates an **Interrupt Request** and sends it to the computer.

When the CPU receives such an signal, it temporarily stops the execution of the program and branches to a service program to process the I/O transfer and after completing it returns back to task, what it was originally performing.

The Execution process of Interrupt–Initiated I/O is represented in the flowchart:

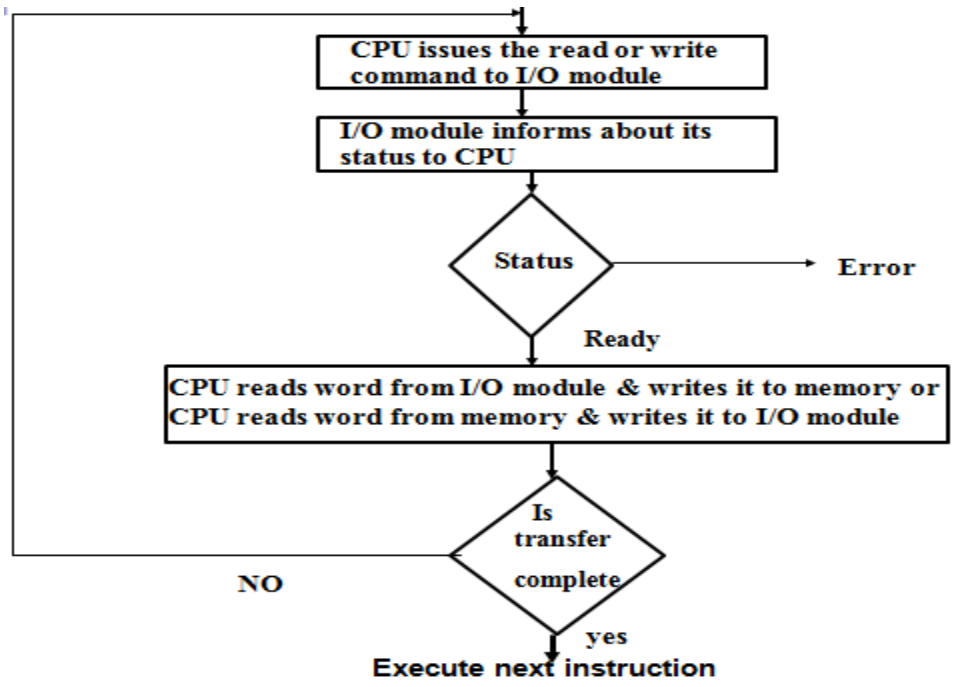


Fig : Interrupt–Initiated I/O

Direct Memory Access (DMA)

In the Direct Memory Access (DMA) the interface transfers the data into and out of the memory unit through the memory bus.The transfer of data between a fast storage device such as magnetic disk and memory is often limited by the speed of the CPU. Removing the CPU from the path and letting the peripheral device manage the memory buses directly would improve the speed of transfer. This transfer technique is called **Direct Memory Access (DMA).**

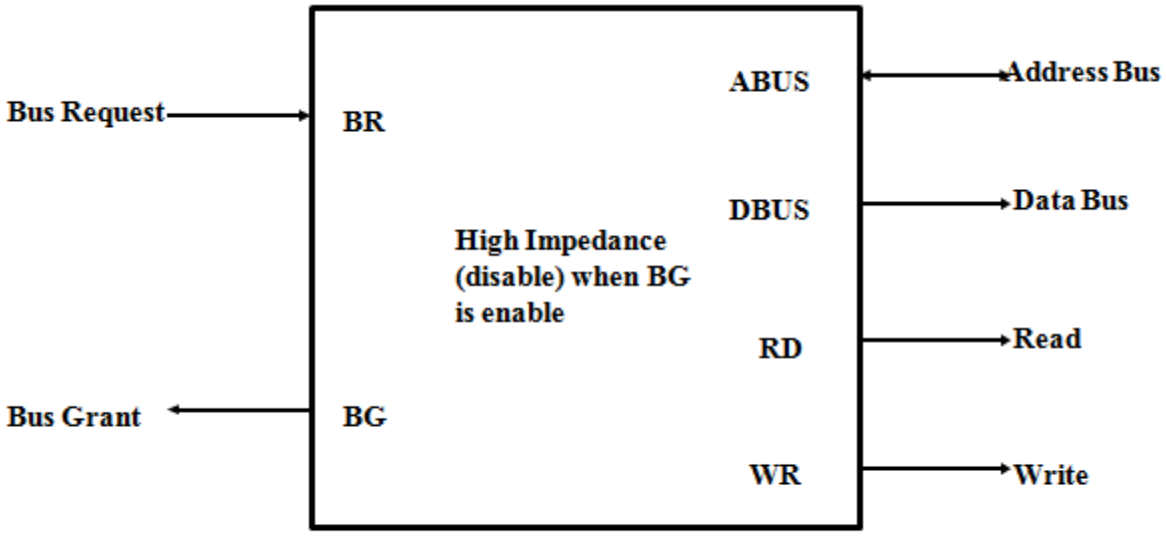
During the **DMA transfer**, the CPU is idle and has no control of the memory buses. A **DMA Controller** takes over the buses to manage the transfer directly between the I/O device and memory.

The CPU may be placed in an idle state in a variety of ways. One common method extensively used in microprocessor is to disable the buses through special control signals such as:

- **Bus Request (BR)**
- **Bus Grant (BG)**

These two control signals in the CPU that facilitates the DMA transfer.

The **Bus Request (BR)** input is used by the **DMA controller** to request the CPU. When this input is active, the CPU terminates the execution of the current instruction and places the address bus, data bus and read write lines into a **high Impedance state. High Impedance state means that the output is disconnected.**



CPU bus Signals for DMA Transfer

The CPU activates the ***Bus Grant (BG)*** output to inform the external DMA that the Bus Request (BR) can now take control of the buses to conduct memory transfer without processor.

When the DMA terminates the transfer, it disables the ***Bus Request (BR)*** line. The CPU disables the ***Bus Grant (BG)***, takes control of the buses and return to its normal operation.

The transfer can be made in several ways that are:

- **DMA Burst**
- **Cycle Stealing**

DMA Burst :- In DMA Burst transfer, a block sequence consisting of a number of memory words is transferred in continuous burst while the DMA controller is master of the memory buses.

Cycle Stealing :- Cycle stealing allows the DMA controller to transfer one data word at a time, after which it must returns control of the buses to the CPU.

DMA controller

The DMA controller needs the usual circuits of an interface to communicate with the CPU and I/O device. The DMA controller has three registers:

i. Address Register

ii. Word Count Register

iii. Control Register

Address Register :- Address Register contains an address to specify the desired location in memory.

Word Count Register :- WC holds the number of words to be transferred. The register is incremented/decremented by one after each word transfer and internally tested for zero.

Control Register :- Control Register specifies the mode of transfer.

The unit communicates with the CPU via the data bus and control lines. The registers in the DMA are selected by the CPU through the address bus by enabling the **DS (DMA select)** and **RS (Register select)** inputs. The **RD (read)** and **WR (write)** inputs are bidirectional. When the **BG (Bus Grant) input is 0,** the CPU can communicate with the DMA registers through the data bus to read from or write to the DMA registers.

When **BG =1**, the DMA can communicate directly with the memory by specifying an address in the address bus and activating the **RD or WR** control.

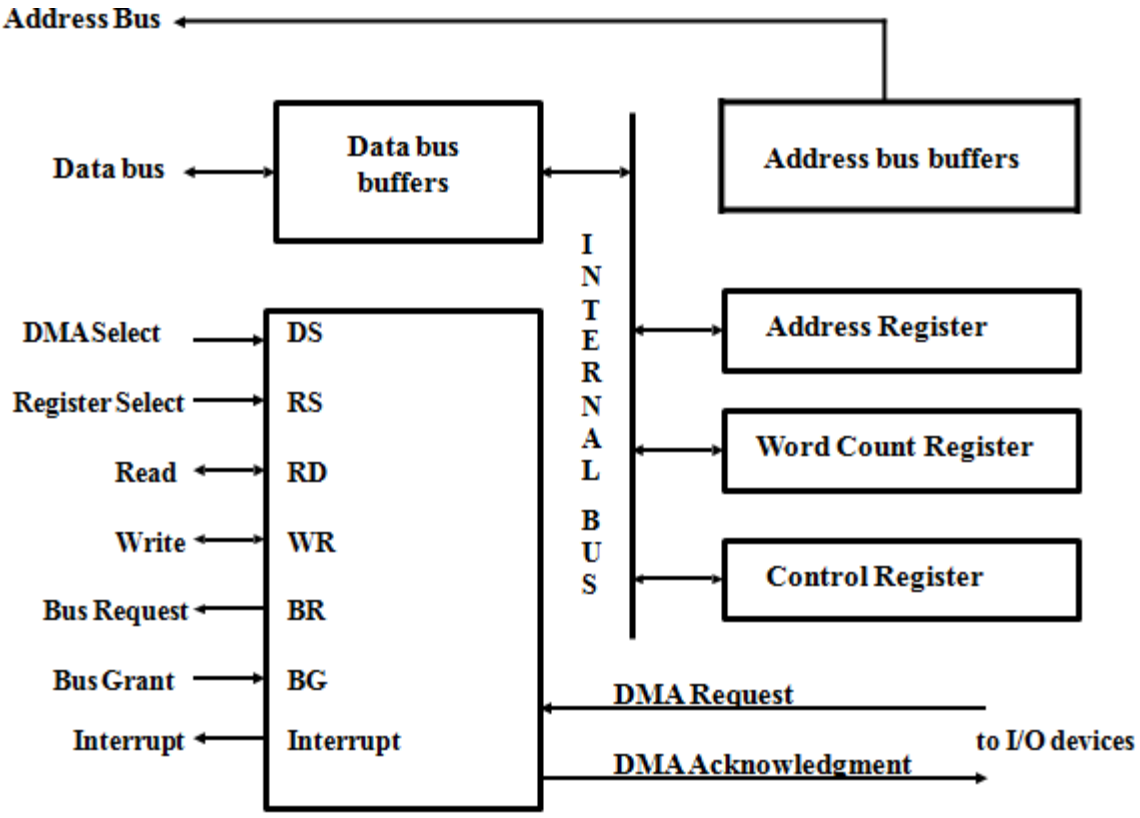


Fig :Block diagram of DMA Controller

The CPU communicates with the DMA through the address and data buses as with any interface unit. The DMA has its own address, which activates the DS and RS lines. The CPU initializes the DMA through the data bus. Once the DMA receives the start control command, it can transfer between the peripheral and the memory.When **BG = 0** the **RD and WR are input lines allowing the CPU to communicate** with the internal DMA registers. When **BG=1**, the **RD and WR are output lines from the DMA controller** to the random access memory to specify the read or write operation of data.