Input/output systems, I/O module-need & functions and Types of data transfer techniques: Programmed I/O, Interrupt driven I/O and DMA

MODULE-6

I/O Organization

INTRODUCTION –I/O SYSTEM

- In addition to the processor and a set of memory modules, the third key element of a computer system is a set of I/O modules.
- Each module interfaces to the system bus or central switch and controls one or more peripheral devices.
- An I/O module is not simply a set of mechanical connectors that wire a device into the system bus.
- Rather, the I/O module contains logic for performing a communication function between the peripheral and the bus.

INTRODUCTION –I/O SYSTEM

We may wonder why one does not connect peripherals directly to the system bus.

The reasons are as follows:

- There are a wide variety of peripherals with various methods of operation. It would be impractical to incorporate the necessary logic within the processor to control a range of devices.
- The data transfer rate of peripherals is often much slower than that of the memory or processor. Thus, it is impractical to use the high-speed system bus to communicate directly with a peripheral.
- On the other hand, the data transfer rate of some peripherals is faster than that of the memory or processor. Again, the mismatch would lead to inefficiencies if not managed properly.
- Peripherals often use different data formats and word lengths than the computer to which they are attached. Thus, an I/O module is required.

This module has two major functions :

- Interface to the processor and memory via the system bus or central switch
- Interface to one or more peripheral devices by tailored data link

I/O MODULE

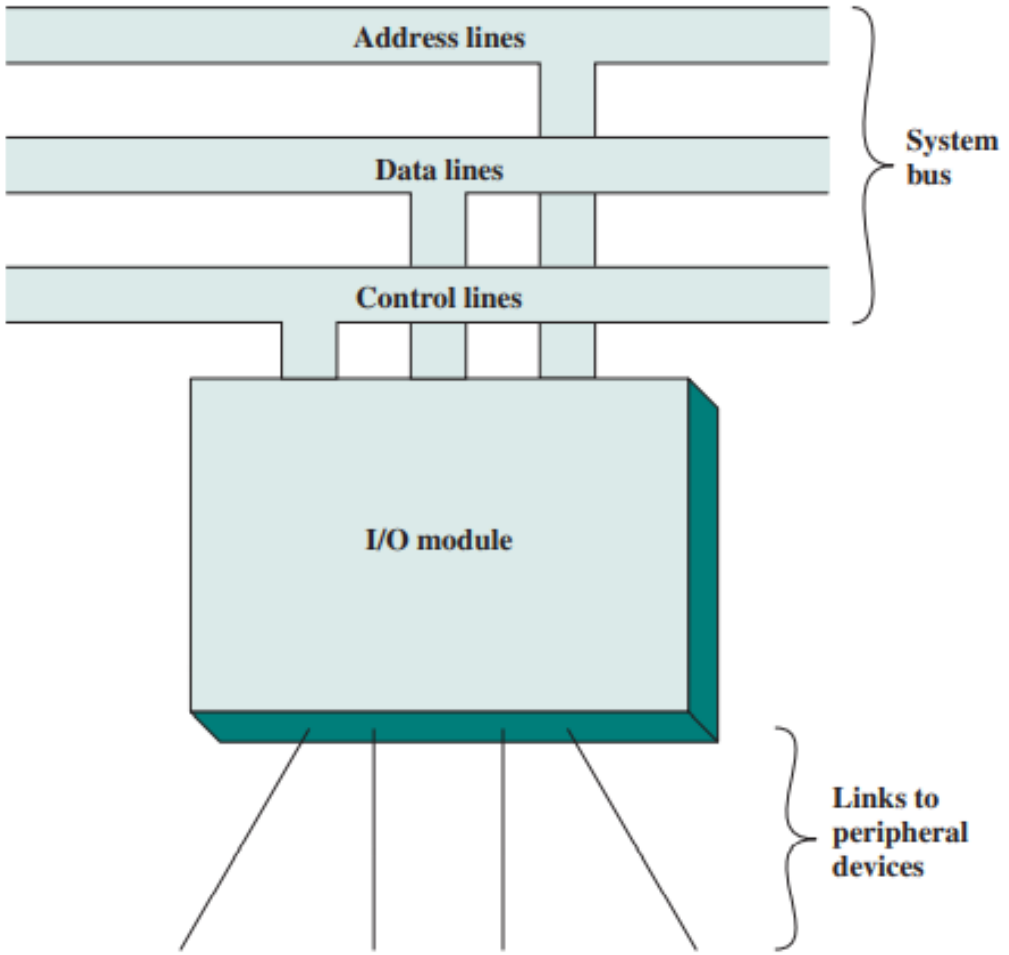


Figure 7.1 Generic Model of an I/O Module

I/O MODULE

- I/O operations are accomplished through a wide assortment of external devices that provide a means of exchanging data between the external environment and the computer. An external device attaches to the computer by a link to an I/O module.
- The link is used to exchange control, status, and data between the I/O module and the external device.
- An external device connected to an I/O module is often referred to as a peripheral device or, simply, a peripheral.

I/O DEVICES

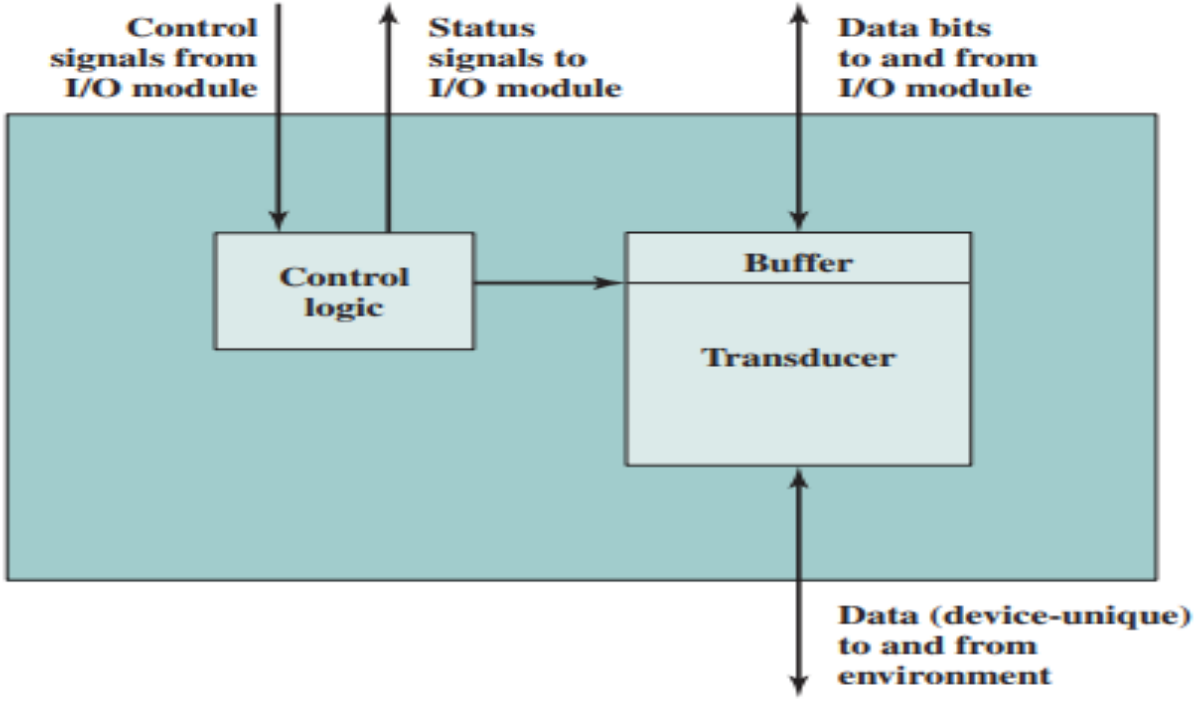


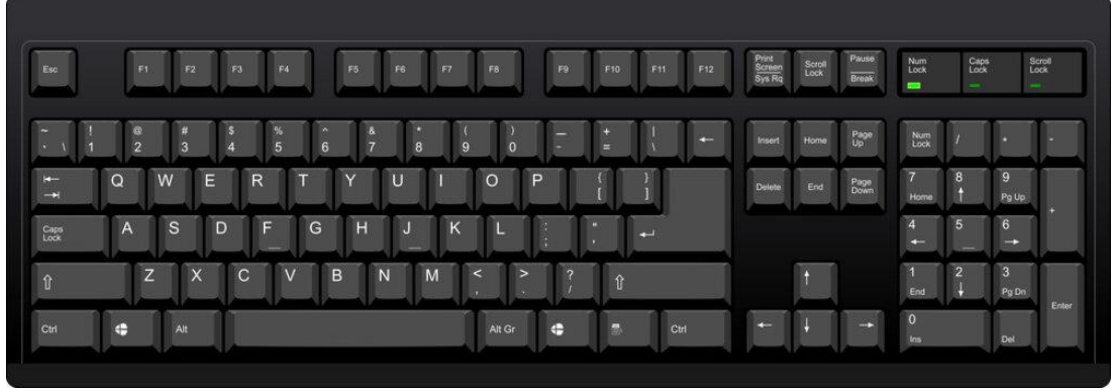
Figure 7.2 Block Diagram of an External Device

- We can broadly classify external devices into three categories:
- **Human readable**: Suitable for communicating with the computer user
- **Machine readable**: Suitable for communicating with equipment
- **Communication**: Suitable for communicating with remote devices

I/O DEVICES

- Examples of human-readable devices are video display terminals (VDTs) and printers.
- Examples of machine-readable devices are magnetic disk and tape systems, and sensors and actuators, such as are used in a robotics application
- Communication devices allow a computer to exchange data with a remote device, which may be a human-readable device, such as a terminal, a machine-readable device, or even another compute

Keyboard/Monitor



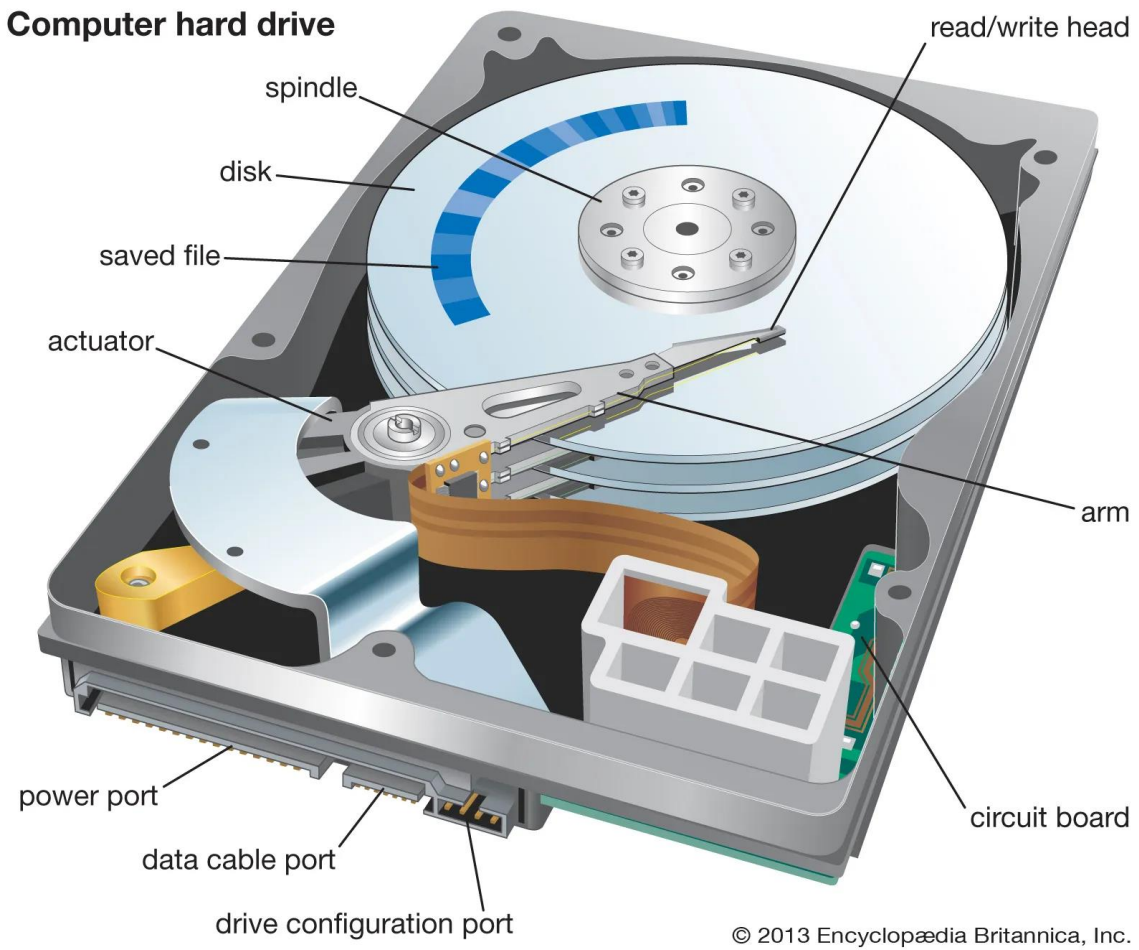
- The most common means of computer/user interaction is a keyboard/monitor arrangement.
- The user provides input through the keyboard. This input is then transmitted to the computer and may also be displayed on the monitor.
- The basic unit of exchange is the character. Associated with each character is a code, typically 7 or 8 bits in length.
- The most commonly used text code is the International Reference Alphabet (IRA).
- Each character in this code is represented by a unique 7-bit binary code; thus, 128 different characters can be represented.
- Characters are of two types: **printable and control**.
- **Printable** characters are the alphabetic, numeric, and special characters that can be printed on paper or displayed on a screen.
- Some of the **control characters** have to do with controlling the printing or displaying of characters; an example is carriage return.
- Other **control characters** are concerned with communications .

Keyboard/Monitor

- For keyboard input, when the user depresses a key, this generates an electronic signal that is interpreted by the transducer in the keyboard and translated into the bit pattern of the corresponding IRA code.
- This bit pattern is then transmitted to the I/O module in the computer.
- On output, IRA code characters are transmitted to an external device from the I/O module.
- The transducer at the device interprets this code and sends the required electronic signals to the output device either to display the indicated character or perform the requested control function

Disk Drive

- A disk drive contains electronics for exchanging data, control, and status signals with an I/O module plus the electronics for controlling the disk read/write mechanism.
- In a fixed-head disk, the transducer can convert between the magnetic patterns on the moving disk surface and bits in the device's buffer .
- A moving-head disk must also be able to cause the disk arm to move radially in and out across the disk's surface.



I/O Module Function

- The major functions or requirements for an I/O module fall into the following categories:
- Control and timing
- Processor communication
- Device communication
- Data buffering
- Error detection

- During any period of time, the processor may communicate with one or more external devices in unpredictable patterns, depending on the program's need for I/O.
- The internal resources, such as main memory and the system bus, must be shared among a number of activities, including data I/O.
- Thus, the I/O function includes a **control and timing requirement**, to coordinate the flow of traffic between internal resources and external devices.

contd

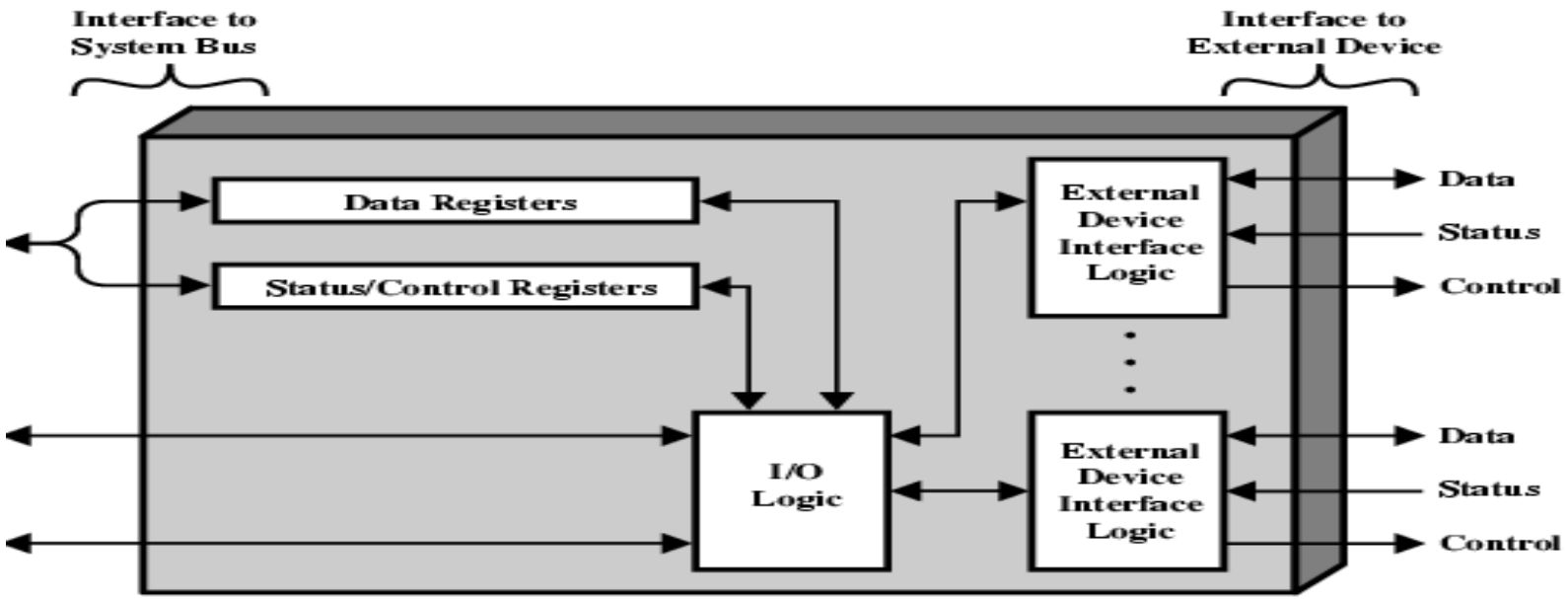


Figure 7.4 Block Diagram of an I/O Module

contd

- For example, the control of the transfer of data from an external device to the processor might involve the following sequence of steps:
- 1. The processor interrogates the I/O module to check the status of the attached device.
- 2. The I/O module returns the device status.
- 3. If the device is operational and ready to transmit, the processor requests the transfer of data, by means of a command to the I/O module.
- 4. The I/O module obtains a unit of data (e.g., 8 or 16 bits) from the external device.
- 5. The data are transferred from the I/O module to the processor

CONTD

- If the system employs a bus, then each of the interactions between the processor and the I/O module involves one or more bus arbitrations.
- **Processor communication** involves the following:
- **Command decoding**: The I/O module accepts commands from the processor, typically sent as signals on the control bus. For example, an I/O module for a disk drive might accept the following commands: READ SECTOR, WRITE SECTOR, SEEK track number, and SCAN record ID.
- **Data:** Data are exchanged between the processor and the I/O module over the data bus.
- **Status reporting**: Because peripherals are so slow, it is important to know the status of the I/O module. For example, if an I/O module is asked to send data to the processor (read), it may not be ready to do so because it is still working on the previous I/O command. This fact can be reported with a status signal. **Common status signals are BUSY and READY**.
- **Address recognition**: Just as each word of memory has an address, so does each I/O device. Thus, an I/O module must recognize one unique address for each peripheral it controls.

contd

- On the other side, the I/O module must be able to perform **device communication**. This communication involves commands, status information, and data .
- An essential task of an I/O module is **data buffering,** function is apparent whereas the transfer rate into and out of main memory or the processor is quite high, the rate is orders of magnitude lower for many peripheral devices and covers a wide range. Data coming from main memory are sent to an I/O module in a rapid burst. The data are buffered in the I/O module and then sent to the peripheral device at its data rate.
- In the opposite direction, data are buffered so as not to tie up the memory in a slow transfer operation. Thus, the I/O module must be able to operate at both device and memory speeds.
- Similarly, if the I/O device operates at a rate higher than the memory access rate, then the I/O module performs the needed buffering operation.

CONTD

- Finally, an I/O module is often responsible for **error detection** and for subsequently reporting errors to the processor. One class of errors includes mechanical and electrical malfunctions reported by the device (e.g., paper jam, bad disk track).
- Another class consists of unintentional changes to the bit pattern as it is transmitted from device to I/O module. Some form of error-detecting code is often used to detect transmission errors.
- A simple example is the use of a parity bit on each character of data. For example, the IRA character code occupies 7 bits of a byte. The eighth bit is set so that the total number of 1s in the byte is even (even parity) or odd (odd parity). When a byte is received, the I/O module checks the parity to determine whether an error has occurred.

Three techniques are possible for I/O operations

- With **programmed I/O**, data are exchanged between the **processor** and **the I/O module**. The processor executes a program that gives it **direct control** of the I/O operation, including **sensing device status**, sending a **read or write** command, and **transferring** the data.
- When the processor issues a command to the I/O module, it must wait until the I/O operation is complete. If the processor is faster than the I/O module, this is wasteful of processor time.
- **With interrupt-driven I/O**, the processor issues an I/O command, continues to execute other instructions, and is interrupted by the I/O module when the latter has completed its work.
- With both **programmed and interrupt I/O**, the processor is responsible for extracting data from main memory for output and storing data in main memory for input.
- The alternative is known as **direct memory access (DMA). In this mode**, the I/O module and main memory exchange data directly, without processor involvement.

Table indicates the relationship among these three techniques.

Table 7.1 I/O Techniques

	No Interrupts	Use of Interrupts
I/O-to-memory transfer through processor	Programmed I/O	Interrupt-driven I/O
Direct I/O-to-memory transfer		Direct memory access (DMA)

Programmed I/O

- When the processor is executing a program and encounters an instruction relating to I/O, it executes that instruction by issuing a command to the appropriate I/O module.
- With programmed I/O, the I/O module will perform the requested action and then set the appropriate bits in the I/O status register .
- The I/O module takes no further action to alert the processor. In particular, it does not interrupt the processor. Thus, it is the responsibility of the processor periodically to check the status of the I/O module until it finds that the operation is complete.
- To explain the programmed I/O technique, we view it first from the point of view of the I/O commands issued by the processor to the I/O module, and then from the point of view of the I/O instructions executed by the processor

Interrupt driven I/O

- The problem with programmed I/O is that the processor must wait a long time for the I/O module of concern to be ready for either reception or transmission of data.
- The processor, while waiting, must repeatedly interrogate the status of the I/O module. As a result, the level of the performance of the entire system is severely degraded.
- An alternative is for the processor to issue an I/O command to a module and then go on to do some other useful work.
- The I/O module will then interrupt the processor to request service when it is ready to exchange data with the processor.
- The processor then executes the data transfer, as before, and then resumes its former processing.

Interrupt driven I/O

- Let us consider how this works, first from the point of view of the I/O module. For input, the I/O module receives a READ command from the processor.
- The I/O module then proceeds to read data in from an associated peripheral. Once the data are in the module's data register, the module signals an interrupt to the processor over a control line.
- The module then waits until its data are requested by the processor. When the request is made, the module places its data on the data bus and is then ready for another I/O operation

Drawbacks of Programmed and Interrupt-Driven I/O

- Interrupt-driven I/O, though more efficient than simple programmed I/O, still requires the **active intervention** of the processor to transfer data between memory and an I/O module, and any data transfer must traverse a path through the processor. Thus, both these forms of I/O suffer from **two inherent drawbacks**:
- 1. The I/O transfer rate is **limited by the speed** with which the processor can test and service a device.
- 2. The processor is tied up in managing an I/O transfer; several instructions must be executed for each I/O transfer
- Using **simple programmed I/O,** the processor is dedicated to the task of I/O and can move data at a rather **high rate**, at the cost of doing nothing else.
- Interrupt I/O frees up the processor to some extent at the expense of the I/O transfer rate. Nevertheless, both methods have an adverse impact on both processor activity and I/O transfer rate.
- **When large volumes of data are to be moved, a more efficient technique is required: direct memory access (DMA)**

Difference between

No.	Programmed I/O	Interrupt Driven I/O
1.	In programmed I/O, processor has to check each I/O device in sequence and in effect 'ask' each one if it needs communication with the processor. This checking is achieved by continuous polling cycle and hence processor can not execute other instructions in sequence.	External asynchronous input is used to tell the processor that I/O device needs its service and hence processor does not have to check whether I/O device needs it service or not.
2.	During polling processor is busy and therefore, have serious and decremental effect on system throughput.	In interrupt driven I/O, the processor is allowed to execute its instructions in sequence and only stop to service I/O device when it is told to do so by the device itself. This increases system throughput.
3.	It is implemented without interrupt hardware support.	It is implemented using interrupt hardware support.
4.	It does not depend on interrupt status.	Interrupt must be enabled to process interrupt driven I/O.
5.	It does not need initialization of stack.	It needs initialization of stack.
6.	System throughput decreases as number of I/O devices connected in the system increases.	System throughput does not depend on number of I/O devices connected in the system.

DMA Function

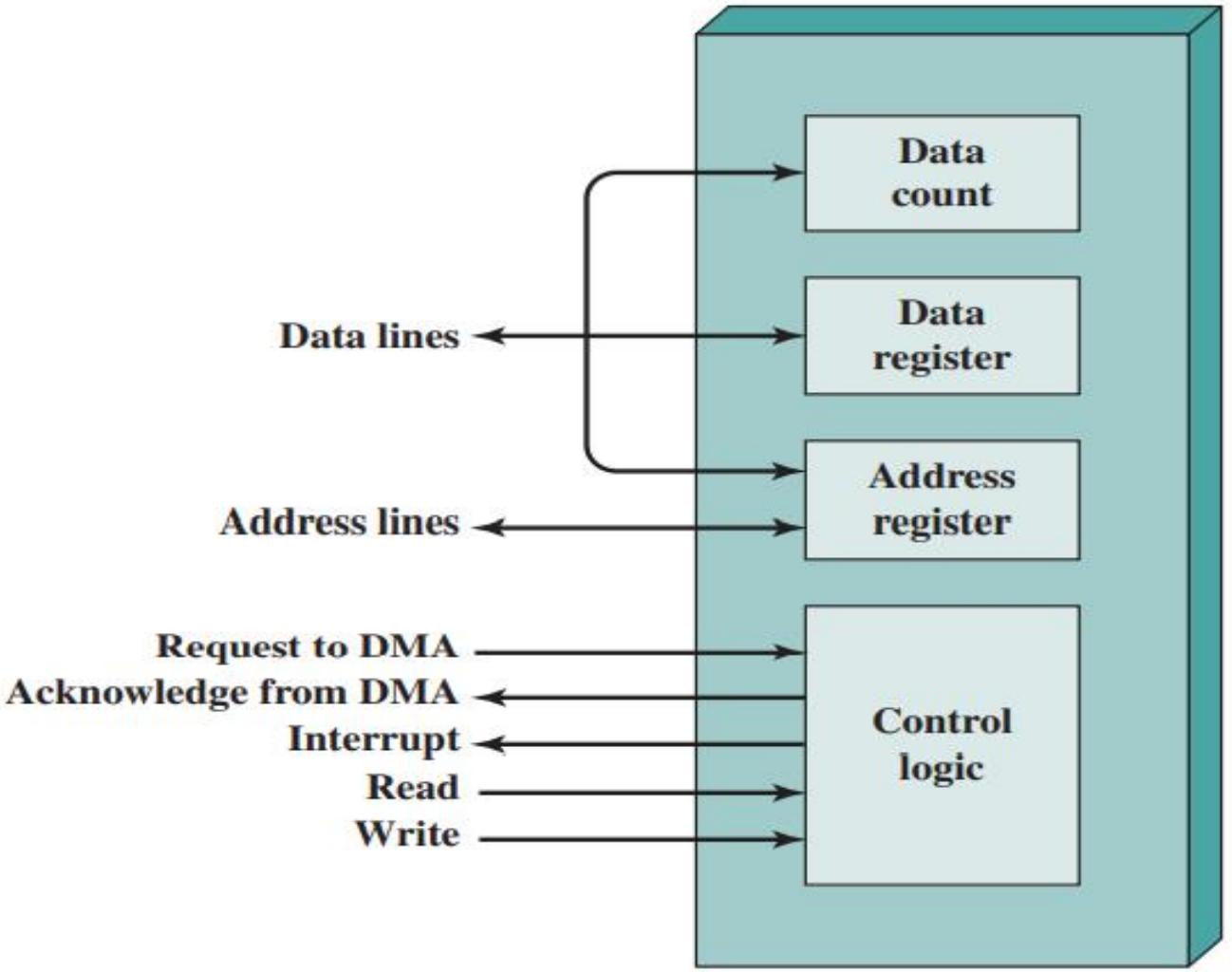


Figure 7.11 Typical DMA Block Diagram

DMA Function

- DMA involves an additional module on the system bus. The DMA module (Figure 7.11) is capable of mimicking the processor and, indeed, of taking over control of the system from the processor.
- 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, or it must force the processor to suspend operation temporarily.
- The latter technique is more common and is referred to as cycle stealing, because the DMA module in effect steals a bus cycle.

DMA Function

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

DMA Function

- 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

Summary

- Advantages of DMA
  - Computer system performance is improved by direct transfer of data between memory and I/O devices, bypassing the CPU.
  - CPU is free to perform operations that do not use system buses.
- Disadvantages of DMA
  - In case of Burst Mode data transfer, the CPU is rendered inactive for relatively long periods of time.