Course Contents

Unit-01: Introduction to Microprocessor

Unit-02: Intel 8085

Unit-03: Microoperations

Unit-04: Control Unit and Central Processing Unit

Unit-05: Fixed point Computer Arithmetic

Unit-06: Input and Output Organization

Unit-07: Memory Organization

Unit-08: Pipelining

Course Contents

Unit-06: Input and Output Organization

Introduction to Peripheral Devices, I/O interface, Direct Memory Access (DMA), I/O Processor, Data communication processor

Unit 6	**Input and Output Organization**	**5 Hours**
6.1	Introduction to Peripheral Devices, I/O interface-I/O bus and Interface Modules, Isolated versus Memory Mapped I/O	1 Hour
6.2	Direct Memory Access (DMA): Introduction, Basic DMA Procedures (DMA controller only)	2 Hours
6.3	I/O Processor, Data Communication Processor: Character Oriented Protocol and Bit Oriented Protocol	2 Hours

Course Contents

Unit-06: Input and Output Organization

6.1 Introduction to Peripheral Devices, I/O interface-I/O bus and Interface Modules, Isolated versus Memory Mapped I/O 1 Hour

Introduction to Peripheral Devices

- Several devices can be used to receive data and display processed data. The devices used to perform these functions are called peripherals or I/O devices.
- Peripherals read information from or write in the memory unit on receiving a command from the CPU.
- They are considered to be a part of the total computer system. As they require a conversion of signal values, these devices can be referred to as electromechanical and electromagnetic devices.
- The most common peripherals are a printer, scanner, keyboard, mouse, tape device, microphone, and external modem that are externally connected to the computer.

Introduction to Peripheral Devices

- In addition to the processor and a set of memory modules, the third key element of a computer system is a set of input-output subsystem referred to as I/O, provides an efficient mode of communication between the central system and the outside environment.
- We can broadly classify peripheral devices into three categories:
  - Human Readable: Communicating with the computer users, e.g. video display terminal, printers etc.
  - Machine Readable: Communicating with equipments, e.g. magnetic disk, magnetic tape, sensor, actuators used in robotics etc.
  - Communication: Communicating with remote devices means exchanging data with that, e.g. modem, NIC (network interface Card) etc.

Introduction to Peripheral Devices

I/O interface -I/O bus and Interface Modules:

- Peripherals connected to a computer need special communication links for interfacing them with the central processing unit. The purpose of the communication link is to resolve the differences that exist between the central computer and each peripheral.

Introduction to Peripheral Devices

I/O interface -I/O bus and Interface Modules:

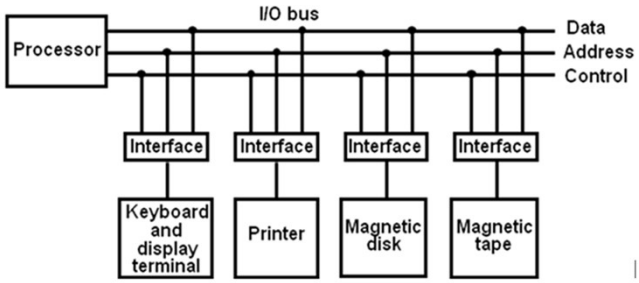
The major differences are:

- **Conversion of signal values -** Peripherals are electromechanical and electromagnetic devices and their manner of operation is different from the operation of the CPU and memory, which are electronic devices. Therefore, a conversion of signal values may be required.
- **Data transfer rate -** The data transfer rate of peripherals is usually slower than the transfer rate of the CPU, and consequently, a synchronization mechanism may be needed.
- **Data codes and formats -** Data codes and formats in peripherals differ from the word format in the CPU and memory.
- **Operating modes -** The operating modes of peripherals are different from each other and each must be controlled so as not to disturb the operation of other peripherals connected to the CPU.

Introduction to Peripheral Devices

I/O interface -I/O bus and Interface Modules:

- To resolve these differences, computer systems include special hardware components between the CPU and peripherals to supervise and synchronize all input and output transfers. These components are called **interface units** because they interface between the processor bus and the peripheral device.



Introduction to Peripheral Devices

I/O Bus and Interface Modules:

- Communication link between the processor and several peripherals is called I/O bus. The I/O bus consists of data lines, address lines and control lines.
- Each peripheral device has associated with it an interface unit.

Introduction to Peripheral Devices

I/O Bus and Interface Modules:

So, the function of Interface modules are following:

- Decodes the device address (device code)
- Decodes the commands (operation)
- Provides signals for the peripheral controller
- Synchronizes the data flow and supervises the transfer rate between peripheral and CPU or Memory

I/O commands that the interface may receive:

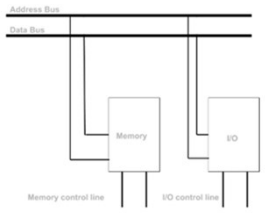
- **Control command**: A control command is issued to activate the peripheral and to inform it what to do.
- **Status command**: A status command is used to test various status conditions in the interface and the peripheral.
- **Output data**: A data output command causes the interface to respond by transferring data from the bus into one of its registers.
- **Input data**: The data input command is the opposite of the data output.

Isolated versus Memory Mapped I/O

I/O versus Memory Bus:

Processor communicate both with the memory unit and I/O devices (peripherals). Like the I/O bus, the memory bus contains **data**, **address**, and **read/write control** lines. There are three ways that computer buses can be used to communicate with memory and I/O:

1. Use two separate buses, one for memory and the other for I/O.
2. Use one common bus for both memory and I/O but have **separate control lines for each**.
3. Use one common bus for memory and I/O with **common control lines**.



Isolated versus Memory Mapped I/O

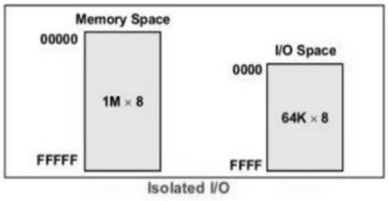
Isolated versus Memory Mapped I/O:

- In the first method, the computer has independent sets of data, address, and control buses, one for accessing memory and the other for I/O.
- This is done in computers that provide a separate I/O processor (IOP) in addition to the central processing unit (CPU). The memory communicates with both the CPU and the IOP through a **memory bus**.
- The IOP communicates also with the input and output devices through a separate I/O bus with its own address, data and control lines.
- The purpose of the IOP is to provide an independent pathway for the transfer of information between external devices and internal memory.

Isolated versus Memory Mapped I/O

Isolated I/O:

- This configuration isolates all I/O interface addresses from the addresses assigned to memory and is referred to as the isolated I/O method for assigning addresses in a common bus.



Isolated I/O

- Many computers use one common bus to transfer information between memory or I/O and the CPU. The distinction between a memory transfer and I/O transfer is made through separate read and write lines. The CPU specifies whether the address on the address lines is for a memory word or for an interface register by enabling one of two possible read or write control lines.

Isolated versus Memory Mapped I/O

Isolated I/O:

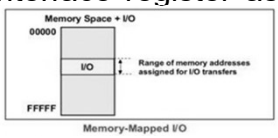
- In the isolated I/O configuration, the CPU has distinct input and output instructions, and each of these instructions is associated with the address of an interface register. When the CPU fetches and decodes the operation code of an input or output instruction, it places the address associated with the instruction into the common address lines. At the same time, it enables the I/O read (for input) or I/O write (for output) control line.

- On the other hand, when the CPU is fetching an instruction or an operand from memory, it places the memory address on the address lines and enables the memory read or memory write control line. This informs the external components that the address is for a memory word and not for an I/O interface.

Isolated versus Memory Mapped I/O

Memory-Mapped I/O:

- The other alternative is to use the same address space for both memory and I/O. This is the case in computers that employ only one set of read and write signals and do not distinguish between memory and I/O addresses. This configuration is referred to as memory-mapped I/O. The computer treats an interface register as being part of the memory system.



Memory-Mapped I/O

- In a memory-mapped I/O organization, there are no specific input or output instructions. The CPU can manipulate I/O data residing in interface registers with the same instructions that are used to manipulate memory words. Each interface is organized as a set of registers that respond to read and write requests in the normal address space.

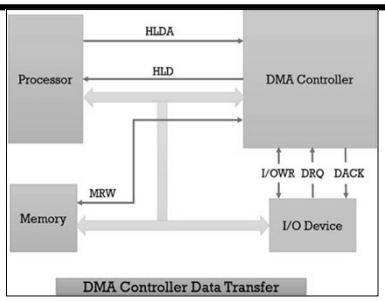
Course Contents

Unit-06: Input and Output Organization

6.2 Direct Memory Access (DMA): Introduction, Basic DMA Procedures (DMA controller only) 2 Hours

Direct Memory Access (DMA)

What is DMA and why it is used?



- Direct Memory Access (DMA) transfers the block of data between the memory and peripheral devices of the system, without the participation of the processor. The unit that controls the activity of accessing memory directly is called a DMA controller.
- Direct memory access (DMA) is a mode of data transfer between the memory and I/O devices. This happens without the involvement of the processor.

Modes of Transfer

Direct memory access (DMA):

The data transfer between a fast storage media such as magnetic disk and memory unit is limited by the speed of the CPU. Thus we can allow the peripherals directly communicate with each other using the memory buses, removing the intervention of the CPU. This type of data transfer technique is known as DMA or direct memory access. During DMA the CPU is idle and it has no control over the memory buses. The DMA controller takes over the buses to manage the transfer directly between the I/O devices and the memory unit.

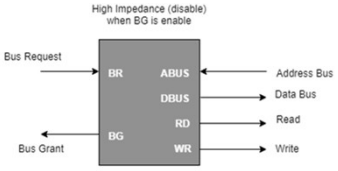


Figure - CPU Bus Signals for DMA Transfer

Modes of Transfer

Direct memory access (DMA):

Bus Request: It is used by the DMA controller to request the CPU to relinquish the control of the buses.

Bus Grant: It is activated by the CPU to inform the external DMA controller that the buses are in high impedance state and the requesting DMA can take control of the buses. Once the DMA has taken the control of the buses it transfers the data. This transfer can take place in many ways.

Modes of Transfer

Direct memory access (DMA):

Types of DMA transfer using DMA controller:

Burst Transfer :

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.

Steps involved are:

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.

Modes of Transfer

Direct memory access (DMA):

Cyclic Stealing :

An alternative method in which DMA controller transfers one word at a time after which it must return the control of the buses to the CPU. The CPU delays its operation only for one memory cycle to allow the direct memory I/O transfer to "steal" one memory cycle.

Steps Involved are:

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.

Modes of Transfer

Direct memory access (DMA):

Interleaved mode:

In this technique , the DMA controller takes over the system bus when the microprocessor is not using it. An alternate half cycle i.e. half cycle DMA + half cycle processor.

Course Contents

Unit-06: Input and Output Organization

6.3 I/O Processor, Data Communication Processor: Character Oriented Protocol and Bit Oriented Protocol
2 Hours

DMA and IOP

- **I/O processors** (IOP) are an extension to the DMA concept of I/O Operation. As the name implies, I/O processors have some processing capacity for serving the device controllers connected to it.
- IOPs can decode and execute I/O instruction. These instructions are brought from Main Memory by the IOP. Data transfer happens with Memory directly. Thus an IOP has a fair amount of control over IO Operations with very minimal initiation from CPU.

IOP

- IOP is a processor with direct memory access capability that communicates with I/O devices. In this configuration, the computer system can be divided into a memory unit, and a number of processors comprised of CPU and one or more IOPs.
- IOP is similar to CPU except that it is designed to handle the details of I/O processing.
- Unlike DMA controller (which is set up completely by the CPU), IOP can fetch and execute its own instructions. IOP instructions are designed specifically to facilitate I/O transfers.
- Instructions that are read form memory by an IOP are called commands to differ them form instructions read by CPU. The command words constitute the program for the IOP. The CPU informs the IOP where to find commands in memory when it is time to execute the I/O program.

IOP

Figure: Block diagram of a computer with I/0 processor.

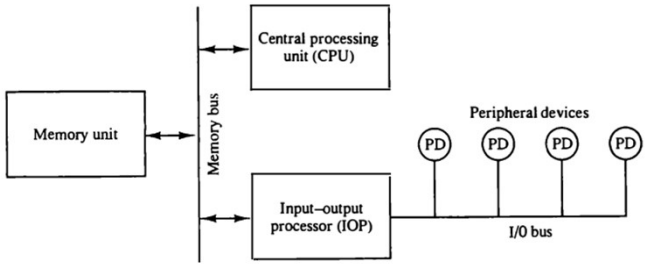


Figure 11-19 Block diagram of a computer with I/O processor.

- The memory occupies a central position and can communicate with each processor by means of DMA. CPU is usually assigned the task of initiating the I/O program, from then on; IOP operates independent of the CPU and continues to transfer data from external devices and memory.

I/O Processor

- A computer may incorporate one or more external processors and assign them the task of communicating directly with the I/O devices so that no each interface need to communicate with the CPU.
- An I/O processor (IOP) is a processor with direct memory access capability that communicates with I/O devices. IOP instructions are specifically designed to facilitate I/O transfer. The IOP can perform other processing tasks such as arithmetic logic, branching and code translation.

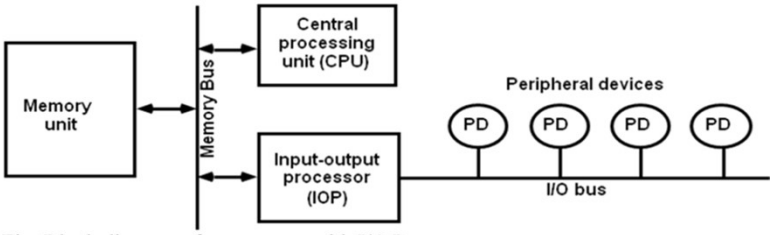


Fig: Block diagram of a computer with I/O Processor

I/O Processor

CPU-IOP Communication:

1. The CPU sends an instruction to test the IOP path.
2. The IOP responds by inserting a status word in memory for the CPU to check. (The bits of the status word indicate the condition of the IOP and I/O device, such as *IOP overload condition*, *device busy with another transfer* or *device ready for I/O transfer*.
3. The CPU refers to the status word in memory to decide what to do next.
4. If all right up to this, the CPU sends the instruction to start I/O transfer.
5. The CPU now continues with another program while IOP is busy with I/O program.
6. When IOP terminates the execution, it sends an interrupt request to CPU.
7. CPU responds by issuing an instruction to read the status from the IOP.
8. IOP responds by placing the contents to its status report into specified memory location. (Status word indicates whether the transfer has been completed or with error)

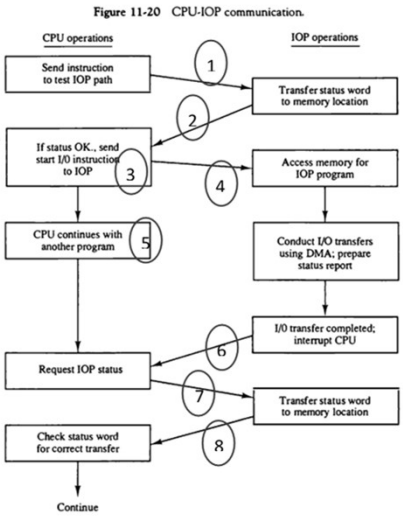


Figure 11-20 CPU-IOP communication.

Data Communication Processor:

(CPU-IOP Communication):

1. The CPU sends an instruction to test the IOP path.
2. The IOP responds by inserting a status word in memory for the CPU to check. (The bits of the status word indicate the condition of the IOP and I/O device, such as *IOP overload condition*, *device busy with another transfer* or *device ready for I/O transfer*.
3. The CPU refers to the status word in memory to decide what to do next.
4. If all right up to this, the CPU sends the instruction to start I/O transfer.
5. The CPU now continues with another program while IOP is busy with I/O program.
6. When IOP terminates the execution, it sends an interrupt request to CPU.
7. CPU responds by issuing an instruction to read the status from the IOP.
8. IOP responds by placing the contents to its status report into specified memory location. (Status word indicates whether the transfer has been completed or with error)

Data Communication Processor:

- A data communication (command) processor is an I/O processor that distributes and collects data from remote terminals connected through telephone and other communication lines. In processor communication, processor communicates with the I/O device through a common bus i.e. data and control with sharing by each peripherals. In data communication, processor communicates with each terminal through a single pair of wires.
- The way that remote terminals are connected to a data communication processor is via telephone lines or other public or private communication facilities. The data communication may be either through synchronous transmission or through asynchronous transmission. One of the functions of data communication processor is check for transmission errors. An error can be detected by checking the parity in each character received. The other ways are checksum, longitudinal redundancy check (LRC) and cyclic redundancy check (CRC). Data can be transmitted between two points through three different modes. First is simplex where data can be transmitted in only one direction such as TV broadcasting. Second is half duplex where data can be transmitted in both directions at a time such as walkie-talkie. The third is full duplex where data can be transmitted in both directions simultaneously such as telephone.

Data Communication Processor:

- The communication lines, modems and other equipment used in the transmission of information between two or more stations is called data link. The orderly transfer of information in a data link is accomplished by means of a protocol. A data communication (command) processor distributes and collects data from many remote terminals connected through telephone and other communication lines.

Transmission:

- Synchronous
- Asynchronous

Transmission Error:

- Parity
- Checksum
- Cyclic Redundancy Check
- Longitudinal Redundancy Check

Transmission Modes:

- Simples
- Half Duplex
- Full Duplex

Data Link & Protocol

Data Communication Processor:

Character Oriented Protocol:

The character-oriented protocol is based on the binary code of a character set. The code most commonly used is ASCII (American Standard Code for Information Interchange). It is a 7-bit code with an 8^{th} bit used for parity. The code has 128 characters, of which 95 are graphic characters and 33 are control characters. The graphic characters include the upper- and lowercase letters, the ten numerals, and a variety of special symbols.

The control characters are used for the purpose of routing data, arranging the test in a desired format, and for the layout of the printed page. The characters that control the transmission are called communication control characters. These characters are listed in Table 11-4. Each character has a 7-bit code and is referred to by a three-letter symbol. The role of each character in the control of data transmission is stated briefly in the function column of the table.

Data Communication Processor:

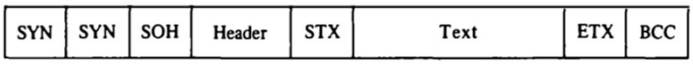
Character Oriented Protocol:

TABLE 11-4 ASCII Communication Control Characters

Code	Symbol	Meaning	Function
0010110	SYN	Synchronous idle	Establishes synchronism
0000001	SOH	Start of heading	Heading of block message
0000010	STX	Start of text	Precedes block of text
0000011	ETX	End of text	Terminates block of text
0000100	EOT	End of transmission	Concludes transmission
0000110	ACK	Acknowledge	Affirmative acknowledgement
0010101	NAK	Negative acknowledge	Negative acknowledgement
0000101	ENQ	Inquiry	Inquire if terminal is on
0010111	ETB	End of transmission block	End of block of data
0010000	DLE	Data link escape	Special control character

Transmission Example:

Figure 11-25 Typical message format for character-oriented protocol.

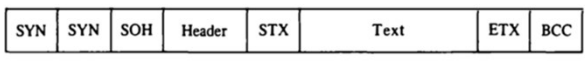


Data Communication Processor:

Character Oriented Protocol:

- The two SYN characters are used to synchronize the receiver and transmitter. The heading (Header) starts with the SOH character and continues with two characters that specify the address of the terminal.
- The STX character terminates the heading and signifies the beginning of the text transmission. The individual characters for this message are not listed in the table because they will take too much space. It must be realized, however, that each character in the message has an 8-bit code and that each bit is transmitted serially.
- The ETX control character signifies the termination of the text characters. The next character following ETX is a longitudinal redundancy check (LRC).

Figure 11-25 Typical message format for character-oriented protocol.



Data Communication Processor:

Character Oriented Protocol:

- The data communication processor receives this message and proceeds to analyze it. It recognizes terminal address and stores the text associated with the message. While receiving the characters, the processor checks the parity in each character and also computes the longitudinal parity. The computed LRC is compared with the LRC character received. If the two match, a positive acknowledgment (ACK) is sent back to the terminal. If a mismatch exists, a negative acknowledgment (NAK) is returned to the terminal, which would initiate a retransmission of the same block. If the processor finds the message without errors, it transfers the message into memory and interrupts the CPU. When the CPU acknowledges the interrupt, it analyzes the message and prepares a text message for responding to the request. The CPU sends an instruction to the data communication processor to send the message to the terminal.

Data Communication Processor:

Bit Oriented Protocol:

The bit-oriented protocol does not use characters in its control field and is independent of any particular code. It allows the transmission of serial bit stream of any length without the implication of character boundaries. Messages are organized in a specific format called a frame. In addition to the information field, a frame contains address, control, and error-checking fields. The frame boundaries are determined from a special 8-bit number called a flag.

Examples of bit-oriented protocols are SDLC (synchronous data link control) used by IBM, HDLC (high-level data link control) adopted by the International Standards Organization, and ADCCP (advanced data communication control procedure) adopted by the American National Standards Institute.

Data Communication Processor:

Bit Oriented Protocol:

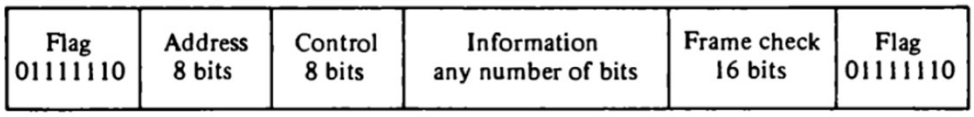
Any data communication link involves at least two participating stations. The station that has responsibility for the data link and issues the commands to control the link is called the primary station. The other station is a secondary station. Bit-oriented protocols assume the presence of one primary station and one or more secondary stations. All communication on the data link is from the primary station to one or more secondary stations, or from a secondary station to the primary station.

Data Communication Processor:

Bit Oriented Protocol:

The frame format for the bit-oriented protocol is shown in Fig. 11-26.

Figure 11-26 Frame format for bit-oriented protocol.



A frame starts with the 8-bit flag 01111110 followed by an address and control sequence. The information field is not restricted in format or content and can be of any length. The frame check field is a CRC (cyclic redundancy check) sequence used for detecting errors in transmission. The ending flag indicates to the receiving station that the 16 bits just received constitute the CRC bits.

Data Communication Processor:

Questions:

1. What is peripheral devices? Explain the categories of peripheral devices.
2. What is Interface unit? Explain the function of Interface unit.
3. What is I/O Bus? Explain the types and function of I/O buses.
4. What is I/O Command? Explain different types of I/O Commands.
5. Indicate whether the following constitute a control, status, or data transfer commands.
    a. Skip next instruction if flag is set.
    b. Seek a given record on a magnetic disk.
    c. Check if I/O device is ready.
    d. Move printer paper to beginning of next page.
    e. Read interface status register.
7. What is the difference between isolated I/O and memory-mapped I/O? What are the advantages and disadvantages of each?
8. What is DMA and why it is used?
9. What is the typical message format for the character-oriented protocol? Explain function of each character for the character-oriented protocol?

Data Communication Processor:

Questions:

9. What is the minimum number of bits of a typical message frame in the character-oriented protocol?
10. What is the frame format for the bit-oriented protocol?
11. What is the minimum number of bits that a frame must have in the bit-oriented protocol? character for the character-oriented protocol?