UNIT 4

INPUT-OUTPUT ORGANIZATION AND MEMORY ORGANIZATION

PART-A

SHORT QUESTIONS WITH SOLUTIONS

Q1. What are the advantages and disadvantages of handshaking?

Answer : Model Paper-I, Q1(g)

Advantages

1. Handshaking provides the highest degree of reliability and flexibility.
2. Source and destination units actively participate in data transfer.
3. The time out mechanism is used to detect the error occurred in one of the two units.

Disadvantage

The only disadvantage of this scheme is that it requires two additional wires or lines for exchanging handshaking signals.

Q2. Write the steps involved in cyclic redundancy check.

Answer :

The steps involved in cyclic redundancy check are as follows,

Step 1: Redundant bits are appended to frame where the number n is less than the number of bits present in the divisor.

Step 2: Data + Redundant bits are divided by the divisor using binary division. The remainder obtained is CRC.

Step 3: Data + CRC is transmitted to the receiver.

Step 4: After receiving data + CRC, receiver performs division using same divisor that is used to find CRC.

Step 5: If remainder = 0, then frame is accepted as it is free from errors. Otherwise it is rejected.

Q3. Give the disadvantages of programmed I/O.

Answer : Model Paper-II, Q1(g)

The disadvantages of programmed I/O are as follows,

(i) Processor has to wait until the input/output module is ready for performing data transfer i.e., while it is in wait state, no other activities can be performed.

(ii) Processor has to interrogate several times regarding the status of input/output module.

(iii) The level of system performance decreases as the time taken for executing each process is very high.

Q4. Differentiate cycle stealing and burst transfers of DMA.

Answer :

Model Paper-III, Q1(g)

	Cycle Stealing		Burst Transfer
1.	Cycle stealing allows transfer of a single data word at a time.	1.	Burst transfer allows transfer of a number of data words i.e., a block at a time.
2.	It is also called the *single byte transfer mode.*	2.	It is also called the *block transfer mode.*
3.	It decreases the performance of the system	3.	It improves the performance of the system.
4.	It does not allow the processor to become idle for a long period of time.	4.	It makes the processor idle for a long period of time.
5.	It doesn't need any register.	5.	It needs a register called byte count which gets decremented upon each byte transfer.
6.	It is useful for those devices which monitors the real-time data such as controllers.	6.	This mode is useful for fast devices like magnetic disks which does not allow stopping or slowing down of data transfer till the entire block of data is transferred.

Q5. What is the purpose of designing data communication processor?

Answer :

A data communication processor is a specially designed processor which is capable of transmitting as well as receiving data to/from various communication devices connected either through the telephone or other communicating interfaces. It communicates through a single pair of wires and the rate of transfer is very slow because both the data and control are transferred serially.

Q6. Compare between main memory and auxiliary memory.

Answer :

Model Paper-I, Q1(h)

	Main Memory		Auxiliary Memory
1.	A computer cannot exist without a main memory.	1.	A computer can exist without an auxiliary memory.
2.	The memory which is directly accessible by the CPU for storing and retrieving information is known as main memory.	2.	The memory that is not directly accessible by the CPU is known as auxiliary memory.
3.	It is also known as primary memory.	3.	It is also known as secondary memory or backup memory.
4.	It is made up of semiconductor material.	4.	It is made up of magnetic and optical materials.
5.	It is internal to CPU.	5.	It is external to CPU.

Q7. What do you mean by associative memory?

Answer :

Model Paper-II, Q1(h)

Associative memory is the memory stores the content rather than the physical address. The stored data is usually accessed by referring it's address. It often turn outs that to be difficult to access the data by means of it's address, especially when there are huge memories storing large volumes of data. Hence, it is an observed fact that, accessing by means of sample data is fast rather than it's address. Hence, the memories which employ this strategy for accessing the data is usually referred to as associative memory or content addressable memory.

Q8. List the characteristics of memory devices.

Answer :

Model Paper-III, Q1(h)

The characteristics of memory devices are,

(i) Access time
(ii) Access rate
(iii) Alterability
(iv) Permanence of storage
(v) Cycle time.

PART-B
ESSAY QUESTIONS WITH SOLUTIONS
4.1 INPUT-OUTPUT ORGANIZATION
4.1.1 Input-Output Interface

Q9. What are various peripheral devices used in computer system? Explain.

Answer :

Peripheral Device

A peripheral device or simply a peripheral is an external device connected to an I/O module.

Types of Peripheral Devices

The various peripheral devices used in computer system are,

1. Keyboard
2. Monitor
3. Printer
4. Magnetic tape
5. Magnetic disk
6. Mouse
7. Scanners.

1. **Keyboard**

Keyboard is the most basic peripheral device through which data is provided to the computer. It consists of various keys which are nothing but alphanumeric and special characters. Whenever user presses these keys, an input in the form of binary data is provided to the computer.

2. **Monitor**

Monitor forms the basic entities through which data is provided to the outside world. Common monitors are CRTs, (Cathode Ray Tubes). They usually consists of an electron gun which continuously emits electron beam. This electron beam is made to strike on the phosphorus coated screen. Hence, an image is displayed on the screen.

3. **Printer**

Printer is a peripheral device which creates a permanent record on paper with ink. It is often called as *hard copy* of output text, images etc. Various types of printers are available such as dot-matrix printers, daisywheel, inkjet, laser printers etc. The print head of the dot-matrix printer contains a set of dots which helps in printing characters. The laser printer contains a rotating photographic drum which is used in printing images and characters. Later, this pattern is transferred to paper.

4. **Magnetic Tapes**

Magnetic tapes are plastic tapes coated with magnetic material. These tapes are used to store and retrieve files or data. A device called *tape-drive* is used to store and retrieve data on it. It allows sequential access to the stored data. It is the most inexpensive medium of storage.

5. **Magnetic Disk**

Magnetic disks are flat circular disks coated with magnetic material. A disk read/write head stores and retrieves data to/ from it. It allows random access to data. They are usually faster than tapes and are used to store bulky data and programs.

6. **Mouse**

Mouse is a small device which can sense movement of itself. These are widely used in Graphical User Interfaces (GUIs) where users move mouse across to the flat surface to move the cursor and select various GUI objects.

7. **Scanners**

Scanners are used to convert the printed material on paper into its equivalent digital format. There are various types of scanners. A flat-bed scanner consists of a glass plate on which the paper to be scanned is placed. The laser beam present inside the scanner focuses light on the paper and the reflected light is read as signals and later they are converted to binary data.

Q10. What is the need of I/O Interface? Explain.

Model Paper-III, Q8(a)

Answer :

An I/O interface allows the transfer of the data or information between external I/O devices and internal storage. For communication of peripherals with CPU a specific, I/O interface or communication link is required since there are some differences between the central computer and each peripheral. The differences are,

1. Peripheral is an electromechanical and electromagnetic device whereas a computer system is an electronic device. So, peripherals require conversion of signal values since these devices perform their operations in a way which is different from CPU and memory operations.
2. Peripherals require synchronization technique to be applied since their data transfer rate is slower when compared to the transfer rate of the CPU.
3. The word representation in CPU and memory is different from the representation of data codes and formats in peripherals.
4. Peripherals are operated in different modes and each must run without disturbing the functioning of other peripherals connected to CPU.

The above differences can be removed by providing a specific hardware component called interface units between the processor bus and the peripheral devices in order to have interface between them. This interface is needed for supervision and synchronization of all I/O transfers. Digital computer system refers "interface" as a collection of signal connection points between two components of the system. The term "interface" refers to the position of connection between two parts of the system "to interface" means to connect two parts of the system for transfer of data between them using their respective interface points. There are two main types of interfaces - CPU interface and I/O interface.

(i) CPU Interface – It is consistent to system bus.

(ii) I/O Interface – This interface totally rely on the nature and qualities of the input-output devices. For connection of input/output devices to CPU, an input/output interface is needed.

I/O interface performs three main functions like synchronization, data conversion and device selection. Synchronization is the process of matching the functioning speeds of CPU and peripherals. Data conversion is the transformation of digital signals to analog signals and transformation of serial data representation to parallel data representation. Device selection is nothing but selecting an I/O device by CPU on FIFO basis. Hence, I/O interface is not only needed for interfacing I/O devices but also for providing various operations.

Q11. Explain I/O Bus Vs Memory Bus.

Answer :

I/O Bus

The bus used by I/O devices to communicate with the CPU is usually referred to as I/O bus. These buses have got a wide range of applications in a given computer. The major function of this bus is to transfer the data to/from CPU to given I/O devices.

The I/O bus is a combination of data, address and control lines. Each I/O device has an interface which interprets and decodes the control signals from the control lines and drives the peripheral controller. The peripheral controller operates various electromechanical devices. For example, paper motion, print timing etc., of a printer.

All I/O devices are attached to the I/O bus through their respective interfaces. The interface activates itself when it detects its own address on the address bus. At the same time other interfaces disables themselves. The activated/selected interface responds to the function code provided by the control lines, by executing it.

The interface may receive any one of the following four commands,

(i) **Control Command**

It activates, instructs and control a device to perform some operation. For example, move paper, move print head etc.

(ii) **Status Command**

It is used to test the device or to identify its states information. For example, is device ready or in sleep mode etc.

(iii) **Data Output Command**

It is used to transfer data from the bus into the interface registers, which is later stored on the storage device (in case of tapes or disks).

(iv) Data Input Command

It is quite opposite to data output command. Here, interface gets data from the device and stores it in the buffer. Later, processor checks the status of the buffer and accepts the data through the data base.

Memory Bus

The bus which is capable of transmitting data from the main memory to various other peripheral devices is usually referred to as memory bus. These buses carry address signals which reflect particular location in the memory where the given data is stored.

The processor communicates with the memory unit through memory bus. It contains address, data lines and control lines (read/write lines). The memory unit and I/O devices can be accessed by the processor in any of the following ways,

- By maintaining two separate buses for memory unit and I/O device.
- By maintaining a common bus for both I/O and memory, but keeps separate control lines.
- By maintaining only one bus including control lines for both memory and I/O.

To maintain the first method, an I/O Processor (IOP) is required which interacts with the I/O bus independent of the basic processor of the computer. The IOP accesses I/O devices using its own address. The advantage of using IOP is to reduce the processors overhead incurred in performing various input/output operations.

Q12. Explain I/O interface with an example.

OR

With the help of a diagram, explain the interconnection structure of I/O devices with respect to I/O bus.

Answer :

The diagrammatic representation of the interconnection structure of I/O devices and I/O bus is shown in figure (1).

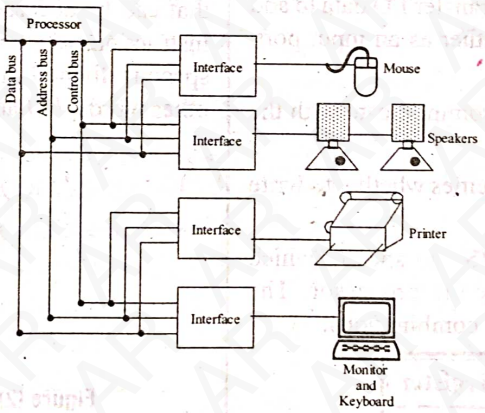


Figure (1): Connection Structure of I/O Bus to Various Peripheral Devices

It can be observed from the figure (1) that the interconnection structure of I/O to various peripheral devices consists of three major signal lines or buses extending from a processor. These buses include address, data and control buses, respectively. To these buses, the required I/O devices are connected. It can also be observed from the above figure that, there is an independent interface unit which is placed before these peripheral devices. These interface units act as decoders, which decodes the information received from the processor and accordingly activates its peripherals. In order to communicate with these I/O devices, the processor sends the device address on the address bus. Each of these interface units collects the address, encodes it and checks to see whether the address belongs to it's own peripheral device. If the received address matches with the address of the given peripheral device, the interface activates the required to peripheral and make it ready to receive further control information. Moreover, the processor now transmits certain special code referred to as I/O command to the required interface unit. The I/O command consists of the required information based on which the given peripheral device has to act upon.

In general terms, these I/O commands are broadly classified into four categories. They are the *control command, the data input command, data output command and the status command.* Each of these commands has it's own significance. For example, the control command consists of data, in order to adhere various control activities, the data output command is used to transmit desired data from the interface unit to the desired peripheral unit. The data input command is used to receive certain data from the given peripheral device into the interface unit and finally the status command is used to check the status of the interface and the peripheral units.

Example

Consider the example of an I/O interface unit shown in figure (2).

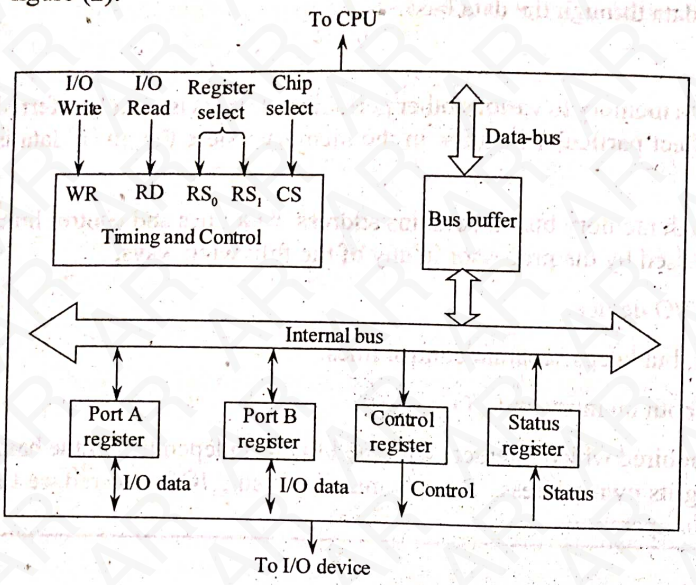


Figure (2): Example of an I/O Interface Unit

One side of I/O interface is connected to CPU and the other side to an I/O device. It consists of a control register, a status register and two bidirectional data ports or registers. It also includes the timing and control unit and data bus buffer.

1. The port *A* and port *B* are used to transfer I/O data to and from the device i.e., it will act either as an input port, output port or both.
2. The data bus is used by CPU to communicate with the interface.
3. The control lines WR and RD specifies whether to write or read data to form the bus.
4. The register select inputs RS_0, RS_1 to specify which register is currently selected by the processor. The following table shows its various combinations.

RS_1	RS_0	Selected Register
0	0	Port A
0	1	Port B
1	0	Control register
1	1	Status register

5. When the Chip Select input (CS) is 0, the whole interface chip is not selected. When it is 1, this chip gets activated.
6. CPU sends the control information to an interface by loading its control register with appropriate bits.
7. The status register stores the current status of the device. CPU reads this register by selecting it ($RS_1 : RS_0 = 1:1$).

Q13. Explain isolated Vs memory mapped I/O.

Answer :

Isolated I/O

As the name suggests isolated I/O is a configuration containing an independent set of input and output instructions. In this case, the central processing unit when receives an opcode of a given instruction, decodes it and then transmits the address of the given peripheral device on the common address lines shared by processor, memory and other I/O peripherals. Hence, the CPU performs the derived operation i.e., either I/O read or I/O write control lines.

In this method, memory addresses and I/O addresses are isolated from each other. Hence, memory address values are not affected by the interface addresses because each of them has its own unique address space. Figure (1) shows the isolated I/O configuration.

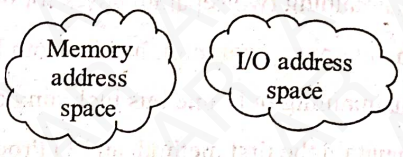


Figure (1): Isolated I/O Configuration

Memory Mapped I/O

In this type of configuration, the CPU maintains common address space for both memory as well as I/O units. Hence, the CPU utilizes the data and status registers belonging to I/O units for storing various addresses.

It utilizes the same instructions for manipulating both I/O data residing in interface registers as well as data residing in the memory. Each interface has to maintain a set of registers that can be accessed by the CPU using the same read, write instructions as it uses for accessing memory. The total address space is divided into I/O addresses and memory addresses. In other words, I/O addresses are part of memory address space.

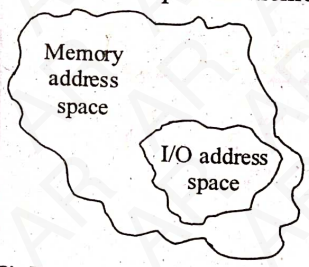


Figure (2): Memory Mapped I/O Configuration

4.1.2 Asynchronous Data Transfer

Q14. Explain the strobe control method of asynchronous data transfer.

Answer :

Strobe control method is one of the methods to transfer data asynchronously. In this method, when a unit desires to transfer data, it sends a strobe pulse to the other unit. This indicates when to initiate the data transfer process. The strobe pulse can be activated either by the destination unit or by the source unit.

1. Source Initiated Strobe

The data is transferred from the source to destination through the data bus. In this mode of transfer, the source unit initiates the strobe which is actually a single line whose job is to inform the destination unit about the presence of valid data word in the bus.

The source initiated strobe for data transfer is shown in figures (1) and (2).

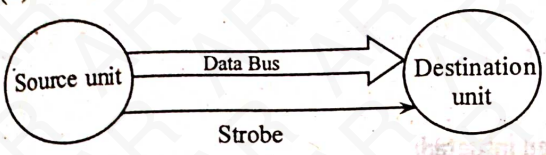


Figure (1): Block Diagram of a Source Initiated Strobe

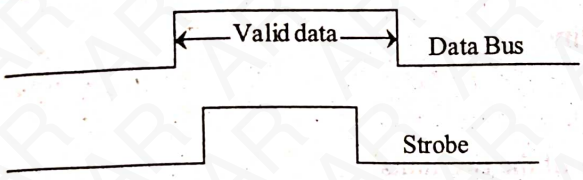


Figure (2): Timing Diagram of Source Initiated Strobe

At first, the source unit will place the data onto the data bus. The strobe pulse is initiated little later so that the data is settled in the data bus to a steady value. The destination unit receives data till the data and strobe pulses both remains in an active state. When the edge of the strobe pulse falls, the destination unit stores the data in any of its internal registers. When the strobe pulse is disabled, it means that there is no valid data within the data bus. Thus, the data will be removed from the data bus after a small period of time. When the new valid data is available in the bus, strobe pulse will be enabled again by the source.

2. Destination Initiated Strobe

In this mode of data transfer, the destination initiates the strobe pulse to inform source-unit when to provide the data. The source unit will then place the valid data onto the data bus. The data remains in the bus till the destination unit accepts it. When the edge of the strobe pulse falls, the destination unit will store the data in any of its internal registers. The strobe will be disabled after storing the data. If strobe is disabled, the data is removed from the data bus.

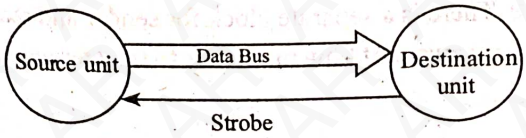


Figure (3): Block Diagram for Destination Initiated Strobe

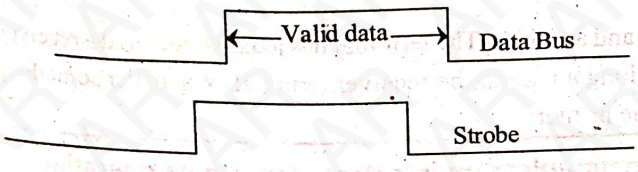


Figure (4): Timing Diagram for Destination Initiated Strobe

The main drawback of the strobe method is that neither the source nor the destination node has a method of determining that the data arrived is from an intended source and is received by an intended destination.

Q15. What is meant by handshaking? Explain.

Answer :

Handshaking

Handshaking is one of the effective methods adopted during asynchronous data transmission process. Here, the source and destination circuits contain three interface lines running between them. The source circuit, is responsible for transmission of data and the destination circuit is responsible for accepting the transmitted data. But, this transmission process is governed by certain strict rules as dictated by handshaking method i.e., whenever there is a need to transmit data, the source circuit directly places the required data on the *data line* and at the same time triggers the *data valid signal*. The destination circuit receives this data and triggers the *data accepted line*. On completion of transmission, the source circuit disables the data valid signal by setting it as invalid and the data signal remains empty. In response, the destination circuit disables data accepted signal. The source circuit cannot initiate new set of data transmission unless it receives data disabled signal from the destination circuit. A diagrammatic overview of the above mentioned scenario followed by it's timing diagram is given below.

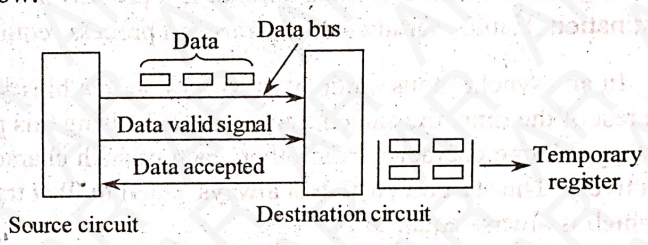


Figure (1): Simple Block Diagram (Source Initiated) using Handshaking Mechanism

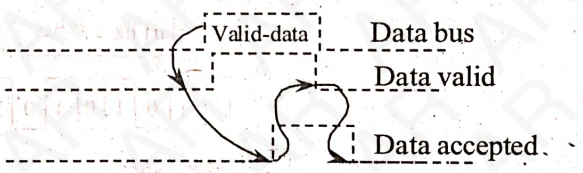


Figure (2): Timing Diagram (Source Initiated) Using Handshaking Mechanism

Now, consider the consequences observed while initiating the transmission process by the destination circuit using handshake mechanism. The only difference in this case is that the destination network initially sends a *ready to accept data signal* and after receiving, the source circuit places the data on the data bus and triggers the data valid signal. The remaining process remains same. This scenario is diagrammatically represented as follows,

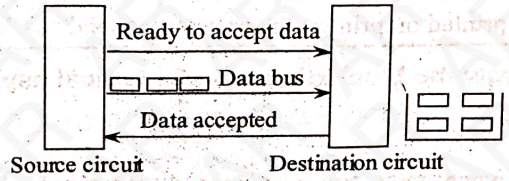


Figure (3): Block Diagram (Destination Initiated) using Handshake Mechanism

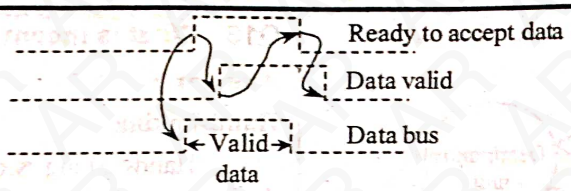


Figure (4): Timing Diagram (Destination Initiated)

Advantages

1. Handshaking provides the highest degree of reliability and flexibility.
2. Source and destination units actively participate in data transfer.
3. The time out mechanism is used to detect the error occurred in one of the two units.

Disadvantage

The only disadvantage of this scheme is that it requires two additional wires or lines for exchanging handshaking signals.

Q16. Write short notes on asynchronous serial data transmission.

Answer :

In general, serial data transmission is a process in which data is transmitted serially over a single interface from source to destination. Hence, serial data transmission process requires single cable with a common ground.

In an asynchronous mode of transmission, the binary data is transmitted through the cable only when it is available and in the rest of the time, the cable remains unused. Using this process only a single character can be transmitted at a time. In order to distinguish one character from other, each of such characters are accompanied by two special bits (i.e.,) start and stop bits, respectively. The start bit (which is always equal to 0) is transmitted first followed by the character bits and finally by the stop bit (which is always equal to 1).

Hence, the receiver can easily distinguish and assemble various characters easily. If there is no data for transmission, the cable remains idle which is indicated by transmitting a series of 1's. Following is a diagrammatic representation of this operation.

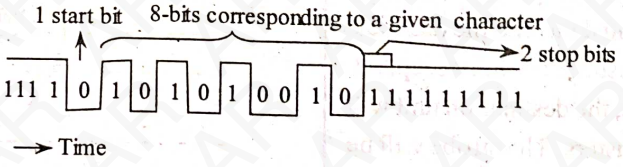


Figure: Principle Implemented Behind Asynchronous Serial Transmission Process

The start bit is detected by the receiver when the line goes from 1 to 0. There is a separate clock for sender and receiver. The receiver's clock helps in receiving the bits at proper bit time. The receiver is aware of how many bits to transfer per second (i.e., the data transfer rate) and how may character bits to receive/accept.

After receiving the character bits, one or two stop bits are detected to signify the end of the character. There should be at least one or two stop bits so that both sender and receiver gets resynchronized.

Consider an example of a computer terminal having a keyboard and a printer. The terminal has a transmitter and a receiver. Whenever a key is pressed on the keyboard, 11-bits are serially sent along a wire to the receiver terminal. Whenever something has to be printed on printer, the transmitter sends 11-bit message to the printer.

Q17. Draw the block diagram of a typical asynchronous communication interface. Explain its operation.

Answer :

Asynchronous communication interface or UART (Universal Asynchronous Receiver Transmitter) is an integrated circuit that allows a computer and similar interactive terminals to communicate by providing an interface between them. Its block diagram is shown below,

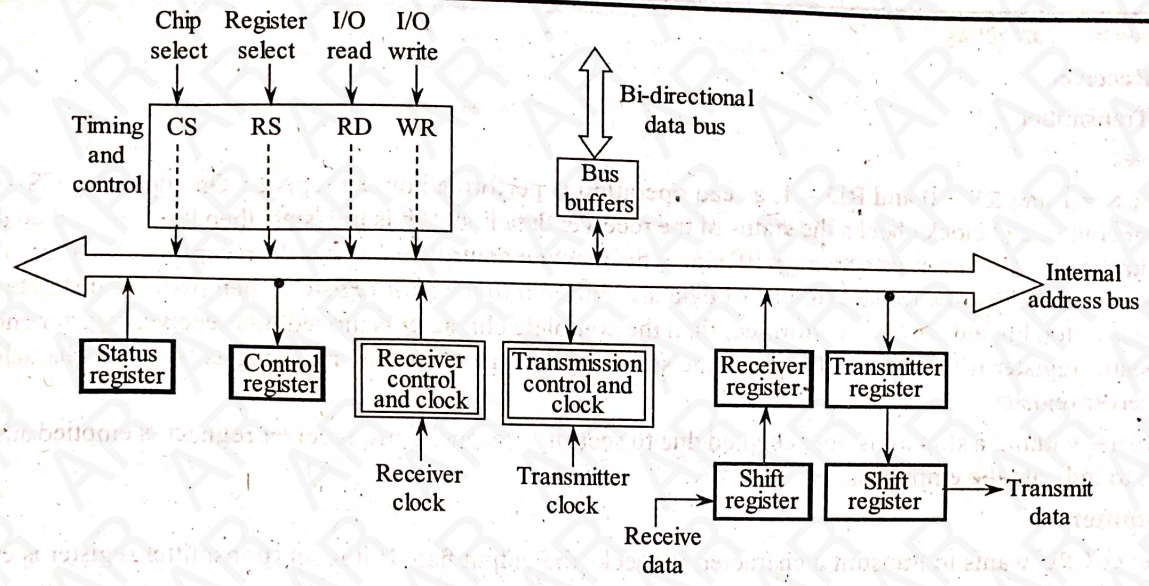


Figure: Block Diagram of UART

The asynchronous communication interface can be used for both transmitting and receiving information. The control lines CS, RS, RD and WR receives input signals from the CPU. The status of these signals determines the operation to be performed by an interface. This is shown in the table below.

CS	RS	Operation to be Done	Register Selected
0	X	X	None
0	0	WR	Transmitter register
1	1	WR	Control register
1	0	RD	Receiver register
1	1	RD	Status register

Table: Relation between CS, RS, WR and RD

CPU selects the interface by using the Chip Select (CS) signal. Writing to and reading from registers is done by using the status of WR and RD lines and Register Select (RS) pin. Only four registers can be directly accessed by the CPU.

The registers are as follows,

(i) Transmitter register

(ii) Control register

(iii) Receiver register and

(iv) Status register.

The first two registers are write-only and the last two registers are read-only.

Bus buffers are used to store data that is read from the receiver register, and has to be written to the transmitter register. The CPU access these buffers through its bidirectional data bus. When this data bus is in high-impedance state, no register is selected and hence no operation is performed. The CPU initializes the interface by writing a control byte to the control specifying few parameters register. The byte specifies few parameters such as length of each character, number of stop bits at the end of each character, baud rate to be used, and parity to be generated and checked.

The status register uses two bits i.e., one for the input flag and the other for the output flag. The input flag is used for checking whether the receiver register is full or empty. The output flag is used for checking whether the transmitter register is full or empty. The remaining bits in status register indicates the occurrence of errors during the transmission process. *Parity error, framing error and overrun error* are the three errors detected by the interface during the data transmission. When the number of 1's is not in accordance with the set parity, then a parity error is detected else a framing error is detected. As soon as a character arrives in the receiver register and if the CPU fails to read it and at the same time if the next character arrives in the status register, then the previous character is lost and overrun error is detected.

The interface can act as,

(a) Receiver

(b) Transmitter.

(a) Receiver

When CS = 1 and RS = 0 and RD = 1, a read operation is performed on the register. On signalling CS = 1 and RS = 0, receiver control and clock checks the status of the receiver data line. If it is in 1-state then it means that no data is received or the line is idle. However, on sensing a '0' signal the receiver control announces the reception of start bit. At the baud rate specified by the CPU, the remaining bits of data are shifted into the shift register. Then parity and stop bits are checked. If there is a stop bit and parity is confirmed, then the complete character is moved into receiver register and the input flag bit in status register is set. Once CPU reads the status register and finds the input flag as 'set', the character is read from the receiver register.

A character without a stop bit is not accepted due to security reasons. Thus, receiver register is emptied and the input flag is reset to indicate the emptiness.

(b) Transmitter

Whenever CPU wants to transmit a character, it checks the output flag. If it is set (transmitter register is empty) then the following actions occur.

(i) CPU transmits a character through the bidirectional data bus into the transmitter register.

(ii) A start bit is generated by making the first bit in the transmitter register as '0'.

(iii) The character along with the start bit is transferred to the shift register and certain number of stop bits are added at the end of the character.

(iv) The output flag is set to indicate that the transmitter register is full.

(v) With the specified baud rate, bits in the shift register are transmitted one at a time.

During data transmission the interface is considered as double buffered due to its ability to load a new character into the transmitter register as soon as the first character bit is transmitted to the shift register.

Q18. What is FIFO buffer? Explain the organization of 4 × 4 FIFO buffer.

Answer :

FIFO Buffer

FIFO buffer is a type of storage location which stores the required data in First-In First-Out (FIFO) manner.

Organization of 4 × 4 FIFO Buffer

Following is the circuit diagram illustrating the organization of 4 × 4 FIFO buffer.

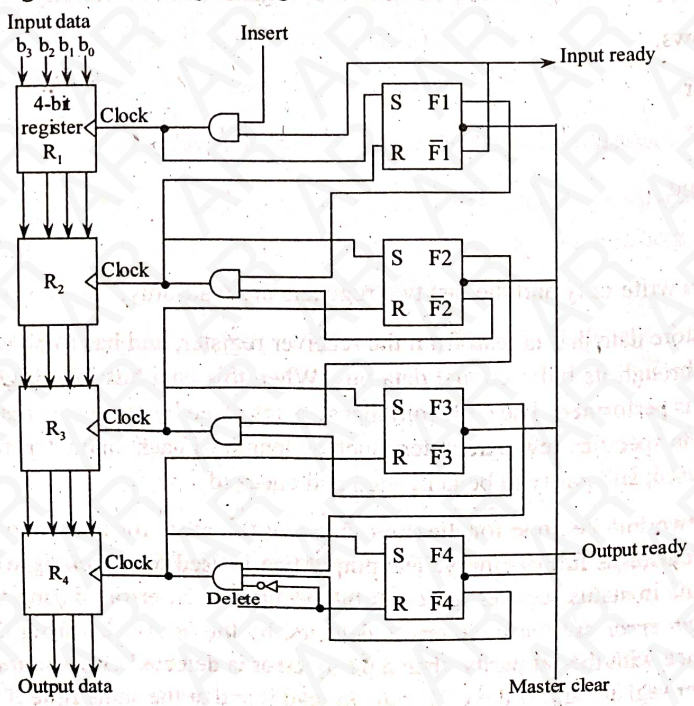


Figure: 4 × 4 FIFO Buffer Circuit Diagram

FIFO (First-In First-Out) is a well-known analogy in any computing discipline. It assumes that the data entered first should be moved out first. The FIFO buffers adopts this analogy. The importance of these buffers is that, they remain compatible with the difference in the rates of insertion and deletion i.e., FIFO buffers work promptly even when there is a difference between insertion and removal of data from the buffer. For example, if these buffers are placed between source and destination units of varying speed, then they must adjust themselves to the situation where the source is of high speed with respect to the destination or destination is of high speed with respect to the source units. It can be observed from the above diagram that each of the four registers, is capable of storing 4 bits of data. In addition to these registers, there are four control registers along with four SR flip-flops. Apart from this circuitry, there are four AND gates and a single NOT gate. In the above diagram, whenever flip-flops (F1, F2, F3, F4) contains the value '1', the corresponding register contains the 4-bit data words and if the flip-flops have value '0', it indicates that the corresponding register contains the invalid data. The storage capacity of these registers can be enhanced by increasing the number of storage blocks and the number of words (combination of four bits) can be enhanced by increasing the number of registers in the above circuit. Assume that F1=1 hence F2 reset its value. A clock is generated and the contents of register 'R_1' gets transmitted to R_2. This process continues till the final register is reached or any intermediatory control register maintains its value F1 as '1'. The master clear line can be utilized to reset the control registers to an initial value of '0'. Moreover, in order to fill the data in these registers the "Input Ready" is enabled which sets the value of F1 to '0'. This indicates that the register R_1 is empty and is ready to accept the data. Hence, the data enters into the register R_1 by passing through an "insert" pin. As soon as the data enters into R_1 the value of F1 is reset to 1, indicating the buffer can accept any more data. Once the data moves from R_1 to R_2, the value of F1 becomes '0' which indicates the insertion of data into the buffer. At the end, as the data successfully processes through all these registers and gets stored at register R_4, the output ready line gets enabled, thereby providing a way for this data to move out of these buffers. Once, the data moves out of these buffers successfully, the ready line gets disabled.

4.1.3 Modes of Transfer

Q19. What are the three modes of data transfer? Explain each mode in detail along with its advantages, disadvantages.

Answer : Model Paper-II, Q8

The three modes of data transfer or three possible techniques for I/O operations are as follows.

(a) Programmed I/O

(b) Interrupt driven I/O and

(c) DMA.

(a) Programmed I/O

It is a technique of organizing an I/O activity such that the processor gives full right to a separate unit called *I/O module* to take charge of the entire I/O activity, which (I/O module) in turn does not take any responsibility to intimate the processor about completion of it's task.

In case of programmed I/O, whenever a processor indulges itself in the program execution, (say) encountering an I/O activity, it calls the desired I/O module and authorizes it. The I/O module after taking charge executes the required I/O activity depending on the availability of the processor. After completion of task, the I/O module sets the contents of I/O status register and switches to other activities. It neither intimates the processor nor interrupts it to provide the status of execution, rather it is the responsibility of the processor to check the status of the I/O module periodically. Also, if there is any requirement of data to be collected from the main memory, the processor collects it and provides it to the given I/O module. Also here, the processor performs various other operations involving, checking of the I/O, device, status, transferring data as well as initiating read/write command, etc.

This process is ineffective since the time of the processor is wasted in handling various meagre activities rather than utilizing it in completing the high-valued tasks. Hence, the usage of program driven I/O has declined to greater extent.

Advantages

(i) I/O command is issued by the processor to the respective I/O module.

(ii) Requested I/O instruction is executed at the earliest.

(iii) I/O module switches to the next task as soon as it completes the assigned task by setting the bit in its status register.

Disadvantages

(i) Processor has to wait until the input/output module is ready for performing data transfer i.e., while it is in wait state, no other activities can be performed.

(ii) Processor has to interrogate several times regarding the status of input/output module.

(iii) The level of system performance decreases as the time taken for executing each process is very high.

(b) Interrupt Driven I/O

This technique is used to overcome the limitations of programmed I/O. In short, in interrupt driven I/O instead of making the processor to check the status of the I/O module, it is the responsibility of the I/O module to intimate the processor by issuing an interrupt signal. The possible cases are given below,

Case 1

Here the interrupt driven I/O process is analyzed from the I/O modules view i.e., the I/O module receives a READ signal from the processor. After receiving the signal, the I/O module gets activated and receives the data from the corresponding I/O unit. Meanwhile, the processor resumes to some other task. As the I/O module complete it's task, it intimates the processor about it's status by sending an interrupt through control lines and wait until it receives the response from the processor. The processor responds back to the I/O module by sending the address of the memory (where the required data has to be stored). The I/O module places it's data to the data lines and reverts back to the same address.

Case 2

Here, the interrupt driven I/O is analyzed from the processors view. Whenever the processor indulges itself in executing a given program and encounters an I/O activity, it temporarily sustain it's activity (by placing its status in the PC and stack pointer), transfers a READ command to the I/O module and resumes back to it's execution by popping the data from PC and SP registers. The I/O module on receiving the command collects the required data from the I/O devices, interrupts the processor, and the entire activity goes on as it was observed in case-1.

Advantages

(i) I/O command is issued by processor to the respective I/O module.

(ii) Processor need not wait for the completion of I/O module i.e., in the meanwhile it performs some other task.

(iii) I/O module intimates the processor about the completion of its task.

Disadvantages

(i) Processor's intervention is required every time when data transfer takes place between memory and I/O module.

(ii) The rate of I/O transfer is restricted by the functioning speed of the processor.

(iii) Processor executes large amount of instructions for each I/O transfer, i.e., more processor time is consumed.

(c) Direct Memory Access or DMA

This technique is more efficient than the programmed I/O and interrupt driven techniques. DMA or Direct Memory Access is generally granted to an independent block which resides either on the system bus or near the I/O module. Whenever the processor analyzes the requirement of performing an I/O activity, it activates the given DMA module by transmitting special information consisting of data such as the address of the given I/O device, status of read/write operations, address of the memory location, amount of data to read/write etc., and the processor resume back to its execution. The DMA module based on this information performs the given task and interrupts the processor about it's status. Hence, in this process, the processor is involved only at the beginning and at the end. As the I/O devices can now transmit their data or interact with the memory directly, the process is called DMA.

Advantages

(i) Data in large volumes can be moved by utilizing system bus.

(ii) DMA module transfers the entire block to and from the memory and informs the processor.

Disadvantages

(i) For transferring the data, DMA module needs the total control of the system bus.

(ii) Processor must wait when it urgently needs the system bus.

(iii) Processor cannot indulge in other task while waiting for the bus.

Q20. Explain programmed I/O in detail.

Answer :

Programmed I/O

The I/O devices can't access the memory directly when programmed I/O method is employed. In order to access the memory, the CPU will have to execute many instructions. These instructions include the following,

1. **An Input Instruction**

   This instruction is used to perform data transfer from the device to the CPU.

2. **A Store Instruction**

   This instruction is used to perform data transfer from the CPU to the memory.

3. **Additional Instructions**

   These are used for,

   (a) Checking whether or not the data is available.

   (b) Counting the total number of words that have been transferred.

Data Transfer from I/O Device to CPU using Programmed I/O Method

The data transfer between the I/O device and a CPU is carried out with the help of an interface. This interface contains two registers, one is the data register and the other is the status register. The status register also contains a flag bit 'F' which indicates the status.

When an I/O device wishes to transfer data to the CPU, it first places the data byte on the I/O bus and then it enables the *data valid line*. This data byte is stored in the data register within the interface. Then, the device will enable its *data accepted line*. After setting the flag bit 'F' of the status register, the data-valid line is disabled. If the device wants to transfer the next byte, it waits until the data accepted line is also disabled.

A flowchart of the CPU program that checks the status register flag to know whether the data register holds the byte from the I/O device is shown in figure (1).

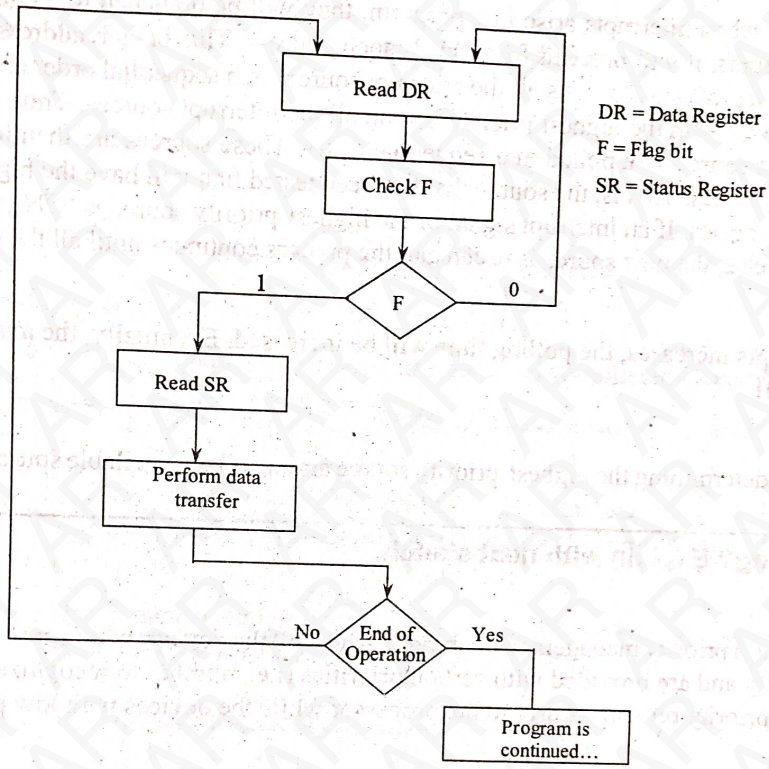


Figure (1): Flowchart to Input Data

Thus, when a device transfers a byte sequence required to be stored in memory, the following three instructions are carried out by each byte.

1. Read the status register
2. Check the flag bit,
   if set, then go to step 3
   else go to step 1.
3. Read the data register.

The data transfer is shown in figure (2).

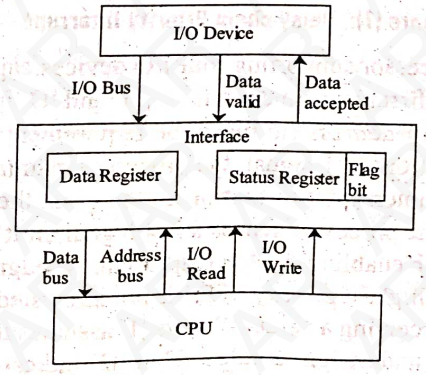


Figure (2): Data Transfer between I/O Device and CPU

To store the data transferred as a sequence of bytes in the memory, it is first read into a CPU register one byte at a time.

4.1.4 Priority Interrupt

Q21. What is priority interrupt? Define polling mechanism.

Answer :

Model Paper-II, Q9(b)

Priority Interrupt

When two or more requests arrive at the same time from various sources, the priority interrupt system establishes the priority for them in order to determine which condition need to be serviced first. The high speed transfers such as magnetic disks are assigned high priority whereas, low speed devices such as keyboards are given low priority. Once the priorities are set, the highest priority devices are serviced first.

According to this method, when interrupts arise in a program, they will be branched to a common branch address. If this program want to handle the interrupts, it will proceed from this branch address. This branch address is the starting address of an interrupt service routine, which is a program that tests all the interrupt sources in a sequential order and will then branch to a single service routine belonging to a source with the highest priority among all the interrupt sources. Thus, when a program branches to this routine, the required interrupt sources are polled in a sequential order. These sources are then tested in some order, and this order corresponds to their priority levels. That is, the source that has been tested first will have the highest priority, and the sources that are tested later have lower priorities. If an interrupt signal of the highest-priority source is 'ON', the control will branch to the corresponding service routine or else, the next source is tested and the process continues until all the sources have been tested.

Disadvantage

As the number of interrupts increases, the polling time will be increased. Eventually, the available service time of an I/O device may be elapsed (finished).

Polling

Polling is a procedure of determining the highest-priority source among all the available sources with the help of a software means.

Q22. What is daisy-chaining? Explain with neat sketch.

Answer :

Daisy-chaining priority is a priority management technique where all the devices which are capable of interrupting a given processor are placed sequentially and are provided with certain priorities (i.e., number to recognize the importance of a device). Here, the devices with highest priority remains nearer to the processor while the devices with low priority are placed away from the processor.

Figure (1) demonstrates the concept of Daisy-chaining priority.

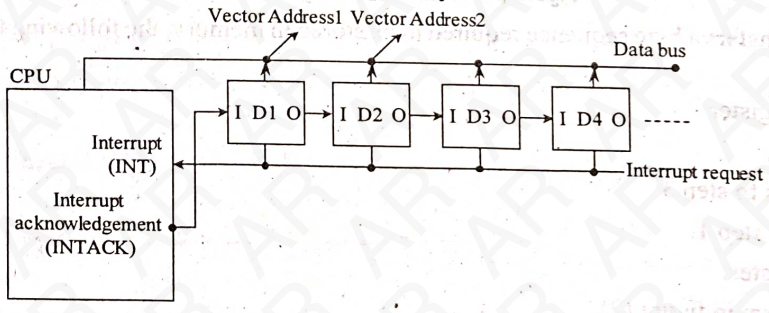


Figure (1): Daisy-chain Priority Interrupt

In the above figure, there is a single processor supporting four I/O devices capable of interrupting the processor. Each of these I/O devices posses four connections, with first two connections i.e., 'I' and 'O' indicating *priority input* and *priority output*, respectively and the other two represents *vector address* (to place the requirement of interrupt) and a *connection to INT line* which is running through the processor. Apart from INT signal, the processor maintains the *vector address signal* and *interrupt acknowledgment signal* in order to collect the interrupt information and to grant the interrupt request. In case of no interrupts, the interrupt request line remains high (1). If the processor senses a low signal on it's interrupt request line, it analyzes that the interrupt has occurred and responds to it through enabling it's Interrupt Acknowledgment line (INTACK). Initially the INTACK line connects to D1 since it is a device with the highest priority. If D1 is not interested in an interrupt, it simply regrets by placing a value '0' at the output terminal 'O'. D2, on receiving a value '0' at its 'I' assumes that it can request for an interrupt and places it's vector address on the processor's data bus and assigns a value '1' at 'O' indicating the device D3 that, the INTACK line is already held by other devices. It may also happen that, the interrupt may be in pending state. In such conditions, the given device disables the INTACK signal required to be transmitted to other devices by placing '1' at it's 'O'.

The device which receives interrupt acknowledgment from the CPU gets the highest priority in daisy chaining. The nearer the device to the CPU, the higher will be its priority. We can assign priority by physically connecting devices in some order.

Consider an example of internal logic of one-stage of priority arrangement as shown in figure (2).

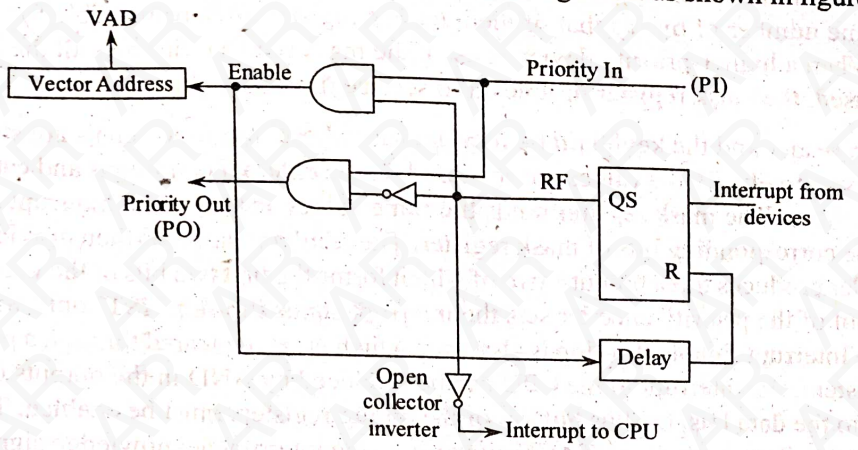


Figure (2): One Stage of Daisy Chaining

1. Whenever a device wants to interrupt CPU, it sets its RF flip-flop.
2. The output of RF goes to CPU via a common interrupt line and an open-collector inverter.
3. When PI is zero, both PO and enable line are zero. It means, this device has not received any acknowledgment i.e., it is in idle state.
4. When PI is 1 and RF is zero, it means this device has received an acknowledgement even though it had not sent any interrupt request. Then, in this case this acknowledgement is passed to the next device via PO. It is done by making PO to 1 and disabling the vector address.
5. When PI is 1 and RF is also 1 then it indicates that, this device has sent an interrupt request and had received an acknowledgement. The PO will become zero and vector address is enabled.

The above steps can be summarized in the following truth table.

PI	RF	PO	Enable
0	0	0	0
0	1	0	0
1	0	1	0
1	1	0	1

Q23. What is an Interrupt? Explain parallel priority interrupt mechanism.

Answer :

Interrupt

Whenever an event occurs during the normal execution of a program, the execution of the program is delayed. Such condition refers to an interrupt. After the program is interrupted, a service procedure is called to handle the interrupt. During the execution of a service program, the control is transferred to the interrupted program which can be resumed after its completion.

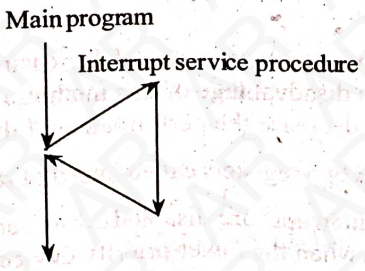


Figure (1): Program Sequence in the Presence of an Interrupt

Parallel Priority Interrupt

To employ the parallel priority interrupt mechanism, two registers called an *Interrupt register* and a *Mask register* are used. The interrupt register stores the interrupt signals of different devices as bits whose position denotes the priority level. The mask register holds the same number of bits as that of the interrupt register bit is responsible for controlling the status of each of the interrupt requests. When a higher-priority device is used, the mask register disables all the lower priority devices. When a lower priority device is used, the mask register disables it to service the higher priority device.

Let the disk, printer, reader and the keyboard be four devices whose interrupt signals are stored in the interrupt registers with values 0, 1, 2 and 3 respectively. These values are set based on the external conditions and can be cleared by providing the appropriate program instructions. The mask register holds the same values as that of the Interrupt register. The bits of Interrupt register are ANDed with the corresponding bits of mask register. The resultant output is then provided as the input to the priority encoder. The priority encoder produces three outputs, two of which forms the first two bits of the vector address and are forwarded to the CPU. The third output of the priority encoder sets the interrupt status flip-flop 'IST' only when an unmasked interrupt is occurred. The IEN, i.e., the Interrupt Enable flip-flop is also used which is set or cleared through a program and is responsible for controlling the interrupt system. An interrupt to the CPU can be produced by AND in the outputs of IST and IEN flip-flops. To place the vector address into the data bus, the bus buffers of the output registers must be enabled. This can be done by ANDing the outputs of IST and IEN flip-flops with the INTACK signal (i.e., the interrupt acknowledge signal) from the CPU.

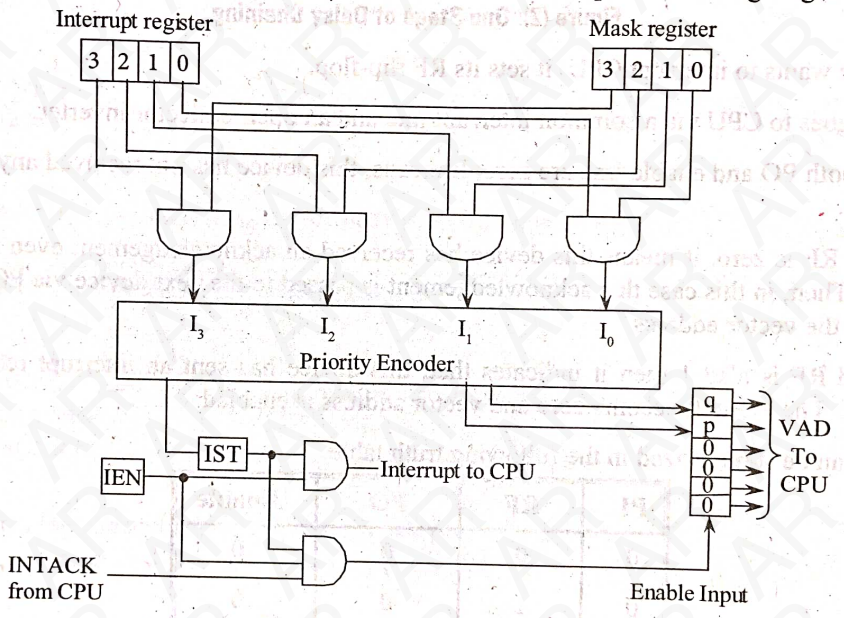


Figure (2): Hardware of the Priority Interrupt

Q24. Out of all Interrupt handling methods, which is not efficient? Justify your answer.

Answer :

The interrupt handling methods are as follows,

(i) Software polling

(ii) Hardware polling (daisy-chaining)

(iii) Parallel priority.

Software polling method polls the interrupts or interrupting devices. But the polling process consumes lot of time. Thus, for large number of interrupts, the time required for polling is greater than the time required to service the devices. This method decreases the overall performance of the system.

Hardware polling method services the devices in a sequence of decreasing priority i.e., the highest priority device is served first followed by the lowest priority device. The disadvantage of this method is that the lowest priority devices will be serviced after a long time. But, still the method does not decrease the performance of the system.

The parallel priority interrupt method uses two registers called interrupt register and mask register for servicing the devices.

In this technique, all the lower priority interrupts are disabled while servicing the higher-priority devices. It also allows the higher priority devices to interrupt the CPU when the lower priority devices are being serviced.

Thus, it can be said that the software polling method is not efficient.

Q25. Explain the operation of a priority encoder.

Answer :

A priority encoder circuit is used to implement the priority function. The logic of this circuit is that, when more than one input is arrived simultaneously then the input that has the highest priority will be processed first. When the priority encoder has four inputs then its truth table is given as,

Inputs				Outputs		
I_0	I_1	I_2	I_3	p	q	IST
1	×	×	×	0	0	1
0	1	×	×	0	1	1
0	0	1	×	1	0	1
0	0	0	1	1	1	1
0	0	0	0	×	×	0

The internal logic can be represented using the following boolean equations for p, q and IST,

$$p = I_0' I_1'$$

$$q = I_0' I_1 + I_0' I_2'$$

$$\text{IST} = I_0 + I_1 + I_2 + I_3$$

I_0, I_1, I_2 and I_3 are the Inputs to the priority encoder. The highest priority input among these inputs is I_0 followed by I_1, I_2 and I_3, respectively. p, q and IST are the outputs of the priority encoder. The Xs present in the table represent the don't care conditions. When the input I_0 is 1, the output will be $pq = 00$, when the input I_1 is 1, the output will be $pq = 01$ only when I_0 is 0. When the I_2 input is 1, the output will be $pq = 10$ only when I_0 and I_1 are zeroes. When the input I_3 is 1, the output will be $pq = 11$ only when I_0, I_1 and I_2 are all zeroes. The IST flip-flop is set to 1 if any of the inputs is 1. The IST will be cleared to 0 if all the inputs are zeroes and the corresponding p and q outputs are marked as don't care conditions.

Q26. Explain the following,

(i) Interrupt cycle

(ii) Software routines.

Answer :

(i) Interrupt Cycle

The sequence of actions performed by a given processor during the time of interrupt is referred to as *interrupt cycle*. Following are the sequence of steps.

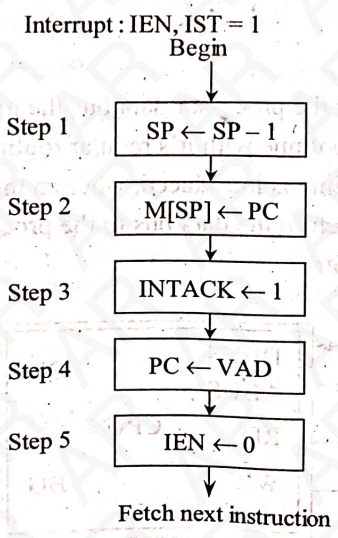


Figure: Interrupt Cycle

In step1, the stack pointer is decremented by 1. Later, the contents of program counter are transmitted into stack pointer. In step 3, the interrupt acknowledgment signal is activated by a given device and it's vector address is transmitted through the data lines of the processor into the program counter. Finally, the IEN value is cleared which disables all the interrupts and the processor switches to the execution of the next instruction.

(ii) Software Routines

Whenever an interrupt occurs, both the hardware and software are responsible for handling it. The software programs, which are resident of the main memory are responsible for directing the interrupts to their respective service programs are referred to as *software routines*. Basically, the branch to these service programs are made by executing the JMP instructions. The exact path to reach a destination service program can be collected from the vector address which is placed by a given I/O device, requesting for an interrupt.

4.1.5 Direct Memory Access

Q27. Explain the following terms with respect to DMA controller,

(i) Bus request

(ii) Bus grant

(iii) Burst transfer

(iv) Cycle stealing.

Answer :

(i) Bus Request

In Direct Memory Access (DMA), whenever the DMA circuitry intends to transmit certain data to be stored or retrieved from the memory, it initially request the processor to grant it a data bus. This is done by interrupting the processor where the entire activity is referred to as bus request.

(ii) Bus Grant

Whenever the given processor receives the bus request signal, it makes it's data bus available to DMA by making its signals to go in high impedance state. The processor activates a signal referred to as 'bus grant' to indicate the high impedance state.

(iii) Burst Transfer

As the DMA attains the control of the processor's data bus, it places it's block of data on the given bus and starts transmitting the data in the form of continuous bursts. This process of data transfer is referred to as burst transfer.

(iv) Cycle Stealing

As the DMA circuit has taken charge over the processor data bus the given processor has to wait for a longtime so that the DMA may release the bus and it can continue with it's regular routines. As a result of it, longer delays are caused. A remedy for this approach called cycle stealing is introduced, wherein the DMA circuitry is allowed to transmit only one block of data at each bus grant and later, return the data bus to the processor. Hence, the data bus switches periodically between the processor and the DMA circuitry.

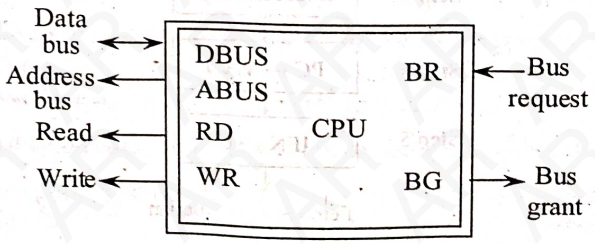


Figure: Bus Signals of CPU for DMA Transfer

Q28. Describe the organization of DMA.

Answer :

Model Paper-I, Q8

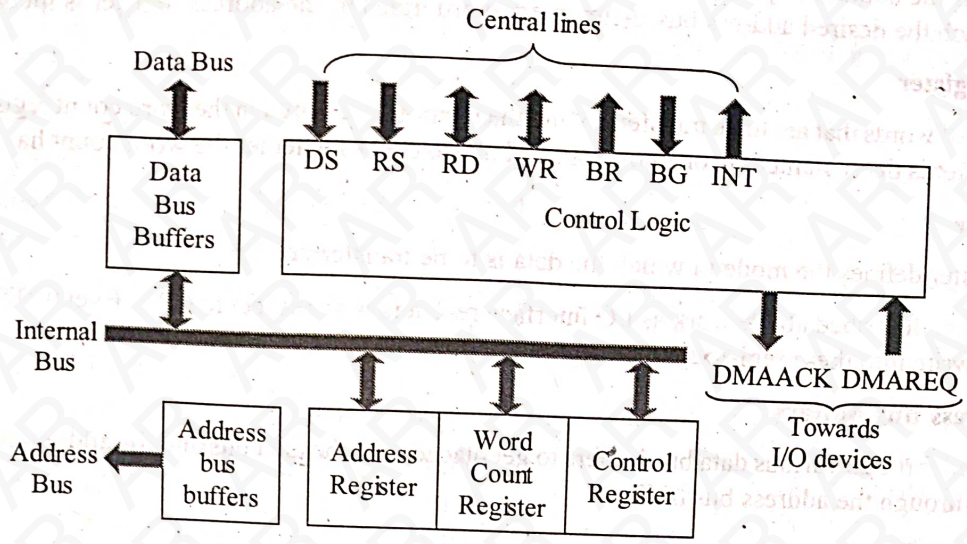


Figure (1): DMA Block Diagram

The DMA controller consists of,

1. Control logic
2. Registers
3. Data and address bus buffers
4. Address and data bus.

1. **Control Logic**

The control logic consists of seven control lines. They are,

(a) DMA Select (DS)

(b) Register Select (RS)

(c) Read (RD)

(d) Write (WR)

(e) Bus Request (BR)

(f) Bus Grant (BG)

(g) Interrupt (INT).

The DS and RS lines are the two select inputs to the control logic. The CPU enables these lines for selecting DMA registers.

The RD and WR lines are bidirectional and can thus work as an input or an output to the control logic.

The BR and BG lines are related to the bus. A request to the bus is made through BR (Bus Request) line. If the BG (Bus Grant) input line is 0 then, the CPU will be able to read or write from the DMA registers using the data bus. If BG input line is 1 then it means that the CPU has handed over the buses to the DMA and the DMA will be able to read or write from the memory. It can be done by placing an address on the address bus and then activating RD and WR control lines. The *interrupt* control line is used to specify the occurrence of the interrupt inside the DMA. In addition to the above control lines, the control logic contains two other lines, one is to get the DMA request and the other is to get a DMA acknowledgment.

2. **Registers**

The three different registers that are used in DMA controller are,

(a) Address register

(b) Word count register

(c) Control register.

(a) Address Register

This register holds the address to point to a specific location inside the memory. The address bits are passed through the bus buffers to reach the desired address bus. After every word transfer, the address register is incremented by one.

(b) Word Count Register

The total number of words that are to be transferred into the memory are stored in the word count register. After transferring a word, this register is decremented by one position, and it also checks whether the word count has reached the value '0'.

(c) Control Register

The control register defines the mode in which the data is tc be transferred.

The DMA registers described above work as I/O interface registers with respect to CPU. Eventually, the CPU can be able to read from or write into these registers based on the program control through the data bus.

3. Data and Address Bus Buffers

The data is passed through various data bus buffers to get placed over the data bus and the address is placed on the address bus by passing through the address bus buffers.

4. Address and Data Bus

The data is transferred to/from the memory using the data bus and the address is transferred to/from the memory using an address bus.

Operation of DMA

CPU initializes the DMA using a program that consists of I/O instructions to specify the address of a particular DMA register. Then the data is transferred between the memory and the peripheral device and this transfer continues till the whole block is transferred.

The CPU sends the following information through the data bus to initialize the DMA.

1. The initial address of memory block defines from where the data has to be read or written.
2. The word count, that is, the number of words contained within the given memory block.
3. A control that specifies whether to perform a read or a write operation.
4. Another control that starts the DMA transfer.

After initializing the DMA, the CPU will not start communicating with the DMA. It communicates only when it receives an interrupt signal or the information about how many words have been transferred.

The computer system consists of RAM, CPU and I/O peripheral devices. The DMA controller is placed along with these components within a computer system as shown in figure (2).

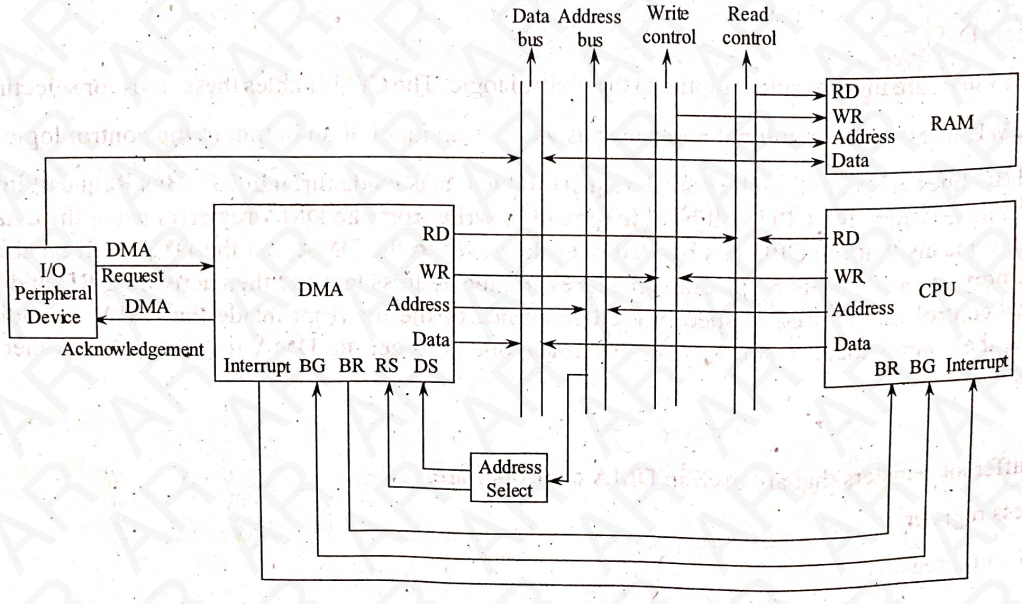


Figure (2): Data Transfer using DMA

The CPU and the DMA can communicate with each other through the address and data buses. The DMA is initialized by the CPU when it sends the start control command to the DMA through the data bus.

As soon as the DMA controller receives the DMA request from the peripheral device, it activates BR line requesting CPU to hand over all the buses. CPU replies through BG line by setting it to '1' and disables all the buses. The DMA will then place the current value of the address register into the address bus and then it initiates either RD or a WR signal. The DMA will then send a DMA acknowledgment signal to the peripheral device. The RD and WR lines of the DMA controller are bidirectional and their directions depends on the BG value. If BG holds '0' then the RD and WR lines acts as inputs to the DMA controller. If BG holds '1' the RD and WR lines will be output from the DMA controller by indicating the read or a write operation within RAM.

The peripheral device can directly put a word in the data bus or it can receive a word from the data bus if acknowledgment is received from DMA. Thus, the peripheral device and the memory can communicate with each other without involving the CPU.

The address register is incremented and the word count register is decremented each time when a word is transferred. If the word count has a value other than zero, then the DMA checks whether there is any other request coming from a peripheral device. If there are any further requests, the DMA will process the request. If there are no requests, then the DMA will hand over all the buses to CPU. When the value in count register becomes zero, the DMA stops and disables all the buses. It will then send an interrupt signal to the CPU to take over all the buses.

Applications

1. It is used for regularly updating the display of an interactive terminal.
2. The information can be transferred between the magnetic disks and the memory with a high speed using DMA transfer.

Q29. What are the different kinds of DMA transfers? Explain.

Answer :

Basically there are three possible configurations of DMA namely,

1. Single bus, detached DMA controller
2. Single bus, integrated DMA-I/O controller
3. I/O bus.

Diagrammatic overview of these three configurations are as follows,

1. Single Bus, Detached DMA Configuration

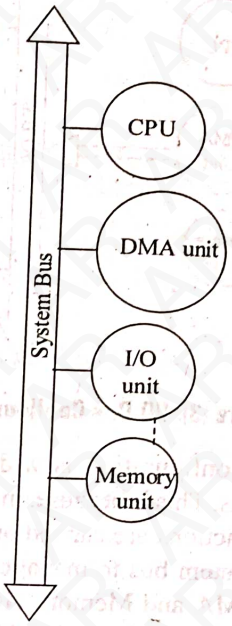


Figure (1): Single Bus, Detached DMA Configuration

As seen from the above configuration, all the CPU units share a single system bus for their processing The DMA unit prefers the programmed I/O technology for supporting data transmission between memory and I/O units. Here, the bus is utilized two times for data transmission and the execution of CPU is halted twice.

2. Single Bus, Integrated DMA-I/O Configuration

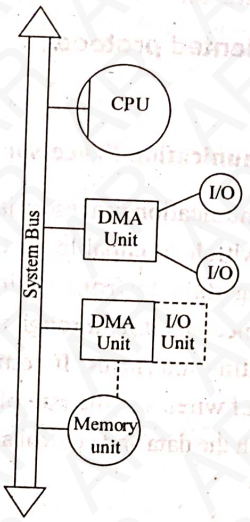


Figure (2): Single Bus, Integrated DMA-I/O Configuration

The given configuration is one step ahead of the configuration discussed previously. Here the DMA unit can be a part of I/O unit or it can manage I/O units separately by different interfaces (other than the system bus). In order to transfer data, the bus is utilized once and the CPU execution is also halted for one time.

3. I/O Bus Configuration

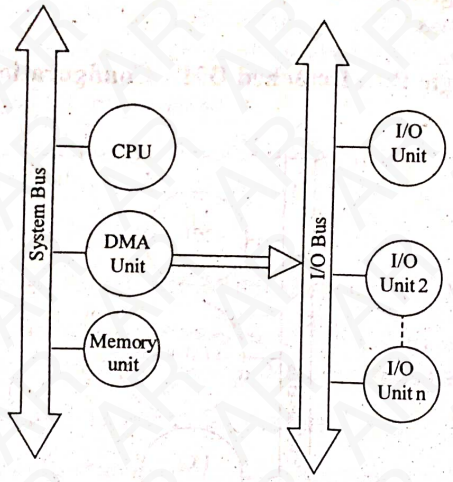


Figure (3): I/O Bus Configuration

The above configuration is widely used due to less number of drawbacks. This is because, in current configuration, the DMA-I/O transactions are carried out using I/O bus only, hence restricting system bus to manage transactions between other units (Say DMA and Memory etc.). Data transfer uses the bus for a single time with single CPU suspension.

Q30. Explain the following with respect to serial communication:

(a) Data communication processor

(b) Modem

(c) Block transfer

(d) CRC

(e) Full-duplex

(f) Bit-oriented protocol.

Answer :

(a) Data Communication Processor

A data communication processor is a specially designed processor which is capable of transmitting as well as receiving data to/from various communication devices connected either through the telephone or other communicating interfaces. It communicates through a single pair of wires and the rate of transfer is very slow because both the data and control are transferred serially.

(b) Modem

A modem is a device (normally used in communication networks) which converts the data from analog to digital (at the transmitter) and digital to analog (at the receiver). The conversion is done to transfer digital data over telephone lines thereby allowing only analog signals through them. A modem uses various signals. It supports various communication grades and transmission speeds.

(c) Block Transfer

When block transfer is done, errors are detected using a character called Longitudinal Redundancy Check (LRC). The sender calculates XOR of all block of characters and sends this character (LRC) with block of data. At the receiving end, the XOR of block of characters is recalculated and compared with the received LRC. If both are identical then there are n errors. Otherwise data is erroneous.

(d) Cyclic Redundancy Check (CRC)

Cyclic Redundancy Check (CRC) is the most powerful error detecting technique wherein the remainder which acts as redundant bits is appended to the end of frame so that the frame obtained by appending the redundant bit should be divisible by a binary number. When the receiver receives the frame, it is divided using the same binary number. If the reminder obtained after division is equal to zero, then the received frame is free from error and is accepted. Otherwise, it is rejected. Cyclic redundancy check is performed by following the below steps,

Step 1: Redundant bits are appended to frame where the number n is less than the number of bits present in the divisor.

Step 2 : Data + Redundant bits are divided by the divisor using binary division. The remainder obtained is CRC.

Step 3 : Data + CRC is transmitted to the receiver.

Step 4 : After receiving data + CRC, receiver performs division using same divisor that is used to find CRC.

Step 5 : If remainder = 0, then frame is accepted as it is free from errors. Otherwise, it is rejected.

(e) Full-duplex

In full-duplex mode, the information can be forwarded in both directions simultaneously. It requires four different wires so that they form pairs. One pair of wires is used for transmitting data in one direction and the other pair of wires is used for transmitting data in the opposite direction.

A full-duplex communication can also be carried out using two-wire circuit. For this, the frequency spectrum must be divided into non-overlapping frequency bands. This will result in the creation of two different transmitter and receiver channels within the same pair of wires.

(f) Bit-oriented Protocol

Bit-oriented protocol is independent of any specific code and doesn't use characters in its control field. Using this protocol, one can transmit serial bit stream of any length. Here each message has a specific format called *Frame*. A frame contains several fields as shown below,

Following is the frame format representing the bit-oriented protocol.

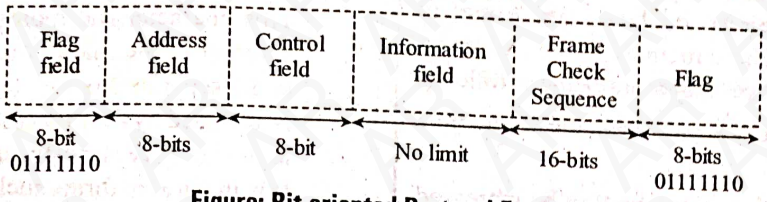


Figure: Bit-oriented Protocol Format

(i) Flag Field

Flag Field marks the beginning and end of the frame format. It is provided at both ends of the frame in order to distinguish between various continuous frames. The standard format of bits representing the flag field is 01111110.

(ii) Address Field

The address field contains 8-bits which represents the address of the destination.

(iii) Control Field

Control field can also accommodate 8-bits of data, which represents the control information like flow control etc.

(iv) Information Field

Actual user data is stored in the information field of the format. It is not fixed rather it is variable. Hence, users are privileged to add large volumes of data.

(v) Frame Check Sequence

The Frame Check Sequence (FCS) field is responsible to ensure an error free transmission of data. In this regard, it utilizes the error detection codes like CRC (Cyclic Redundancy Check) etc.

Examples of bit-oriented protocols includes SDLC (Synchronous Data Link Control) or HDLC (High level Data Link Control). In this case, the bit-oriented frames can be transmitted to multiple destinations.

4.2 MEMORY ORGANIZATION

4.2.1 Memory Hierarchy

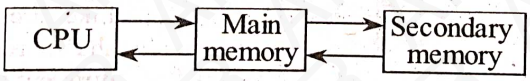
Q31. Show the memory hierarchy and give the brief explanation.

Answer :

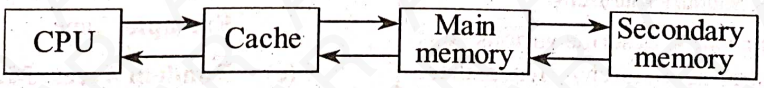
Model Paper-III, Q8(b)

Memory Hierarchy

Memory hierarchy is an arrangement of various memory types with different performance and capacities. The two most common memory hierarchies are shown in the figure (1).



(a) Two-level



(b) Three-level

Figure (1): Common Memory Hierarchies

Typical technologies used in these hierarchies are bipolar semiconductor RAMs for cache memory, MOS semiconductor RAMs for main memory and magnetic disk units for secondary memory.

During the execution of programs, the CPU generates a continuous stream of logical memory addresses. At any time, these addresses are distributed in some way throughout the memory hierarchy. If an address is generated and is currently assigned only to M_i where $i \neq 1$, the address must be reassigned to M_1, the level of the memory hierarchy that the CPU can access directly. This reallocation of logical addresses generally requires the transfer of information between levels M_i and M_1, which is relatively a slow process. For a memory hierarchy to work efficiently, the addresses generated by the CPU should be found in M_1 as often as possible. This requires that future addresses must be predictable to some extent so that information can be transferred to M_1 before it is actually referenced by the CPU. If the desired information cannot be found in M_1, then execution of the program that has initiated the memory request must be suspended until an appropriate reallocation of storage is made.

The memories form an essential part of every computer. The high storage and fast retrieval of data is important in measuring the performance of these memories. The designers while designing them will face three major problems as follows,

1. If the capacity of a memory is increased, the access time will also be increased
2. If access time is reduced, cost per bit will be increased
3. If the capacity of a memory is increased, the cost per bit will be decreased.

The compromise to above mentioned facts gave rise to different levels of technologies known as memory hierarchy.

Formally, the memory hierarchy is divided into two different types mainly,

(i) Contemporary memory hierarchy and

(ii) Traditional memory hierarchy.

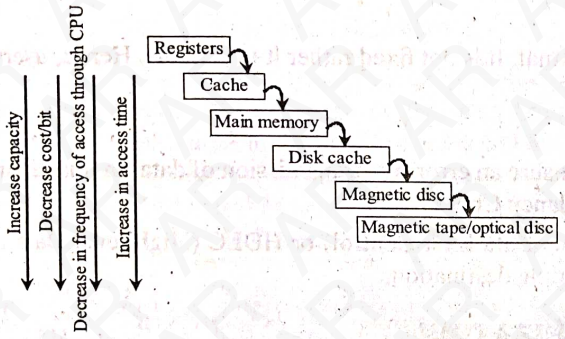


Figure (2): Contemporary Memory Hierarchy

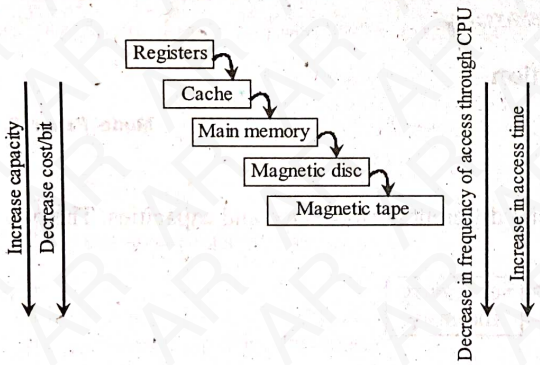


Figure (3): Traditional Memory Hierarchy

The arrows carrying information describe various facts which occur while one goes down the hierarchy. Hence, there is an increase in capacity and access time and decrease in cost/bit and the CPU accessing the memory when proceeding downwards from registers to magnetic tape.

Q32. List and explain the different characteristics of memory system.

Answer :

The different characteristics of memory system includes,

(i) Location (ii) Capacity

(iii) Access method (iv) Unit of transfer

(v) Performance (vi) Physical type

(vii) Organization

(viii) Physical characteristics.

(i) Location

This characteristic represents whether the location is external or internal to the computer. External memory is the separate storage device such as disk and tape. It is accessed by the processor through I/O controllers. Internal memory is mostly embedded within the computer. It is in various forms such as registers, cache and main memory.

(ii) Capacity

Capacity is the most important characteristic of the memory system. It is different for external and internal memory. Capacity of the external memory is measured in terms of bytes, whereas the capacity of internal memory is measured in terms of bytes or words and the word length may be 8, 16 or 32 bits.

(iii) Access Method

This characteristic has four types of methods.

(a) Direct

(b) Sequential

(c) Random

(d) Associative.

(a) Direct Access Method

In this method, every block has its own unique address. Here, the access is made directly by first reaching to vicinity and then following sequential search. Access time of this method is concerned with current and previous location of data.

Example: Hard disk.

(b) Sequential Access Method

In this method, access to data is made sequentially. Like direct access method, it also makes use of shared read-write mechanism. Access time is different for direct access method but depends on same parameters.

Example: Tape.

(c) Random Access Method

In this method, each block of memory has its unique and physically wired-in addressing mechanism. Unlike direct and sequential access, the access time of this method not depend on location.

Example: RAM.

(d) Associative Access Method

In this method unlike other methods, access to data is made by comparing the required bit location within a word based on the specified data element. Therefore, the access time of this method also does not depend on location.

Example: Cache memory.

(iv) **Unit of Transfer**

This characteristic measures the transfer rate of data in and out of the memory. It is different for both the internal and external memory.

(a) For internal memory, it is the number of electrical lines going into and coming out of the memory. These lines can carry the data of word length.

(b) For external memory, it is the number of bits read out or written into the memory at a time.

(v) **Performance**

It is also the most important characteristic of the memory system. The following parameters need to be considered for performance,

(a) **Transfer Rate**

It can be defined as the rate at which data is transmitted into or out of the memory. It is different for different types of memory.

(b) **Memory Cycle Time**

It refers to the time required for accessing a block as well as the period prior to the start of the second access. It is mostly required for random access memory. It is related with the system bus but not the processor.

(c) **Access Time**

It is different for different types of memory. In random access memory, it is the time required to carry-out read or write operation. In non-random access memory, it is the time required to place the read-write mechanism at the desired place.

(vi) **Physical Type**

There are various physical types of memory such as,

(a) Magnetic

(b) Semiconductor

(c) Magneto-optical

(d) Optical.

(vii) **Organization**

It describes the physical arrangement of the bits. It is very important for random access memory.

(viii) **Physical Characteristics**

It describes the nature of memory that whether the memory is volatile, non-volatile erasable or non-erasable. Non-volatile memory does not ensure the persistence of data unlike volatile memory. Erasable memory allows one to change or modify data unlike non erasable memory.

4.2.2 Main Memory

Q33. Write short notes on,

(i) RAM

(ii) ROM

(iii) Bootstrap loader.

Answer :

(i) RAM

Today, Random Access Memories (RAMs) form one of the basic memories of computers. They cannot hold data permanently. Hence, they are utilized to store several intermediate results and other temporary data. Data remains in these memories as long as there is constant supply of power i.e., once the power is turned off data inside it gets lost.

Data inside these memories can be accessed randomly from any location, hence the name of the memory.

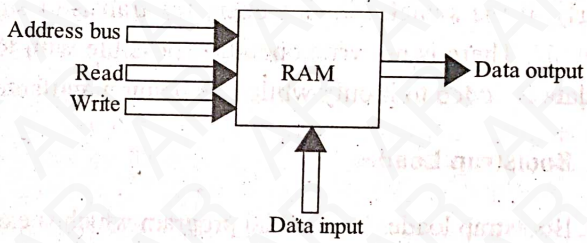


Figure: Random Access Memory

Data reading and writing mechanisms in this memory are quite simple. In order to write the data, initially the address (which describes the position where the data is to be loaded) is placed on the address bus. Later, the required data is placed on the data lines. Finally, write line is activated. In the same way, in order to read the data from this memory, required address is placed on the address lines and read line is activated. So, the data can be collected through data output lines. Hence, in this way reading and writing tasks can be accomplished.

(ii) ROM

Read Only Memories (ROMs) are those memories which do not lose data, even though the power is turned off. Due to this nature, these memories are also referred to as non-volatile memories.

There are many types of read only memories with differing capabilities. Some of them are as follows,

(a) Programmable Read Only Memory (PROM)

(b) Electrically Erasable and Programmable Read Only Memory (EEPROM)

(c) Flash Memories etc.

Operation of ROM

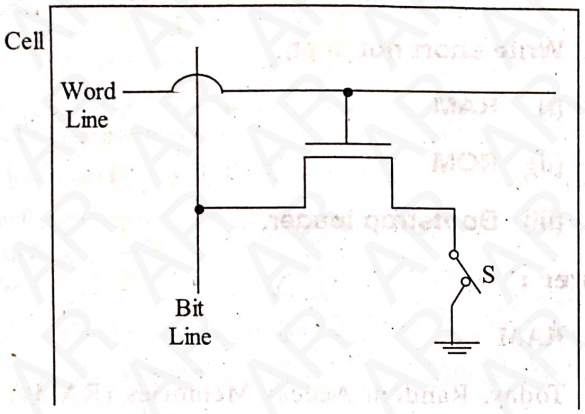


Figure: Single ROM Cell

Here, the transistor is connected to ground by a switch. Hence, whenever the switch is closed, the value of the transistors switches to zero else it remains '1'. In order to read the status of a cell, the wordline is turned high. As a result, if the switch of the transistor is closed, a value '0' flows out of the circuit. Similarly, if the switch remains open, the transistor supplies a value '1'. There is no write operation possible with ROMs, since data is added to it only while it is being manufactured.

(iii) Bootstrap Loader

Bootstrap loader is an initial program which is executed once a computer is turned ON and loads the operating system into the memory. It is stored on non-volatile memory such as ROM or EPROM and hence does not lose its information even if power is turned OFF. When the power is turned ON, the program counter is set to the first address of loader by the computer. Therefore, the operating system is loaded into main memory from the hard disk followed by the transfer of control to the operating system for further processing.

Q34. Draw and explain the block diagram of RAM and ROM chips.

Answer :

RAM Chip

The block diagram of a typical RAM chip is shown in the following diagram,

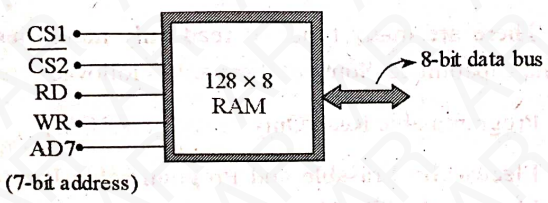


Figure: Block Diagram Representing 128 × 8 RAM (Random Access Memory)

In this block diagram representation, the given RAM chip consists of two chip select lines, a read line, write line, address lines and a bidirectional 8-bit data bus. Here, two Chip Select (CS) lines are used in order to ensure that only the required chip is selected from a given array of chips when required by the microprocessor. Also, it helps in address decoding in the similar conditions.

As usual, the RD and WR lines refer to read and write lines for reading/writing data to/from this memory (RAM). Here, the configuration 128 × 8 signifies that, the current memory element is capable of storing about 128 words with each word consisting of 8-bits. Hence, in order to access the data stored in this memory a 7-bit unidirectional address bus is required.

Operation of RAM circuit can be analyzed by considering the following function table.

CS1	$\overline{CS2}$	RD	WR	Type of function	State (databus)
0	0	X	X	I	HI
0	1	X	X	I	HI
1	0	0	0	I	HI
1	0	0	1	WR	Write data (into RAM)
1	0	1	X	RD	Extract the data (from RAM)
1	1	X	X	I	HI

Notations Used

I → Inhibit

WR → Write

RD → Read

HI → High Impedance.

ROM Chip

The block diagram representation of ROM (Read Only Memory) chip is shown in the following diagram,

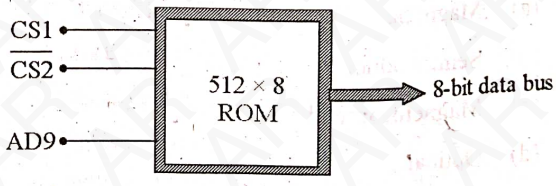


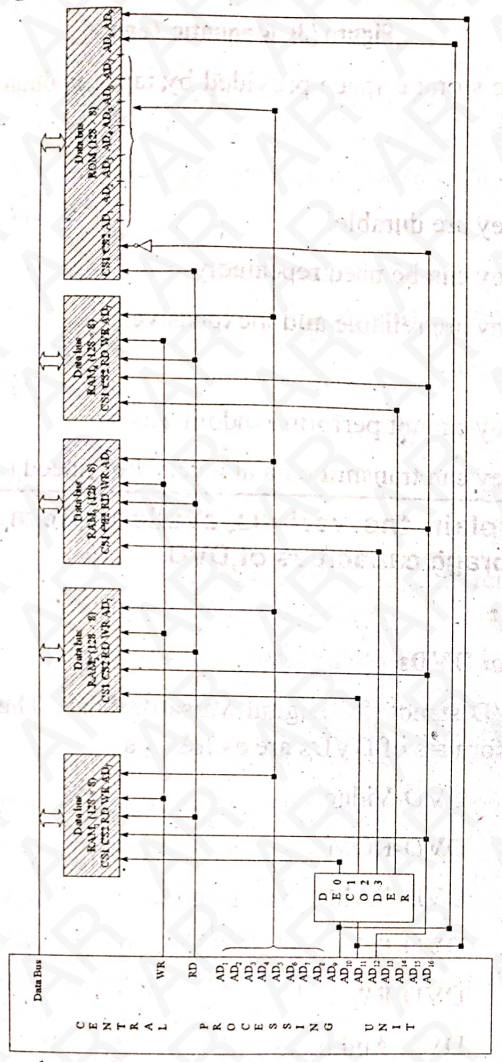
Figure: Block Diagram Representing 512 × 8 ROM Chip

In this block representation, the block of ROM significantly differs from RAM (i.e.,) as ROM refers to read only memory, Hence, its configuration lags write line and also the data bus is unidirectional. The reason for considering 512 × 8 is that, often ROM can accommodate more number of memory cells as that of RAM, hence the space occupied by 512 × 8 by ROM is equivalent to 128 × 8 as that of RAM (since ROM is read only memory, hence large amount of data can be stored within less space).

Q35. Explain memory connection to CPU with 4 RAM chips and 1 ROM chip.

Answer :

Memory Connection to CPU with 4 RAM Chips and 1 ROM Chip



The above figure shows the connection of four RAM chips and one ROM chip to CPU using address bus and databus. With the connection through address bus, each low-order lines select a specific byte in all four chips (i.e., AD7). On the other hand the high-order bits of address bus connects to the chips through chip-select inputs i.e., (CS1 and $\overline{CS2}$). This entire arrangement accounts upto a memory capacity of 512 bytes of RAM and 512 bytes of ROM. As each RAM chip has one to seven lower order bits of address bus connected to it, this allows the RAM to select any one of the 128 possible bytes. Now the selected RAM chip is specified through 8 and 9 address lines with the use of 2 × 4 decorder. And the generated output is directly connected to CS1 input present in each RAM.

So, during the processing, the selection of RAM chip is based on the value displayed by 8 and 9 address lines. That is when address lines 8 and 9 are 00 first RAM chip is selected. When they display 01, selection of second RAM is made and when it is 10, third RAM is selected. And the last RAM is selected when address lines are 11. Also, the read (RD) and write (WR) outputs generated from microprocessor of CPU behaves as input to each RAM chip. In addition to this, the databus connected to RAMs can perform two-way data transfer.

Depending upon these bits, the selection of RAM and ROM is made. So, when the bit across bus line is '0' RAM chip is selected and when the bit is '1' ROM chip is selected. Unlike RAM, the ROM has address bus lines 0-9 connected to it but without passing through decorder. On the other hand, the other Chip Select Input (CSI) is directly connected to RD port of CPU and gets activated when read operation is invoked. Therefore, the entire set up of address bus lines and chip select inputs allocates addresses of 0 to 511 to RAM and 512 to 1023 to ROM. Also, the databus in ROM has only a one-way data transfer (i.e., output).

4.2.3 Auxiliary Memory

Q36. Discuss in detail the two most common auxiliary memory devices used in computer systems.

Answer : Model Paper-II, Q9(b)

The two most common auxiliary memory devices used in computer system are magnetic disks and magnetic tapes.

1. Magnetic Disk

Magnetic disks are used to store bulk of data for long-term in a computer system. A disk is made up of one or more platters which are flat and circular in shape like a CD and have diameters in the range of 1.8 to 5.25 inches. Each platter is a metal disk that is covered with magnetic material on both surfaces. The data is stored on the surface in magnetic form. All platters are fastened to a central spindle that rotates. There is a read-write head which flies above platter. The gap between the head and platter is extremely thin (a few microns) and there is a risk of the head making contact with magnetic surface that may destroy platter. This situation is also called as head crash. The structure of magnetic disk is shown below,

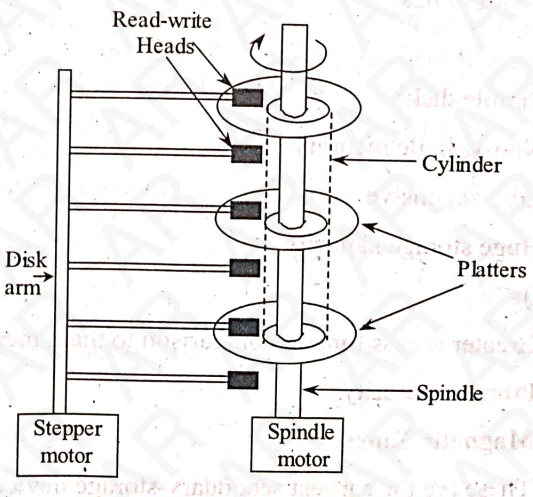


Figure (1): Magnetic Disk Structure

For every platter, there are two read-write heads, one for the top surface and the other for the bottom. All heads are attached to a disk arm (or disk actuator) which is connected to a stepper motor to advance or move heads in steps.

The surface of a platter is logically divided into *tracks* which are circular and that are subdivided into several *sectors*. The set of tracks on the same position in each platter makes one logical *cylinder* as shown in figure (1). The structure of disk platter is shown below,

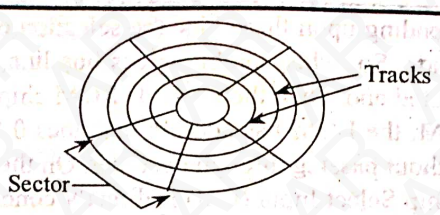


Figure (2): Disk Platter

Disk drives usually come in gigabyte capacity and speed of rotation of spindle motor is nearly 7200 r.p.m (rotations per minute). The speed of operation has the following parameters,

(i) Transfer Rate

It refers to the rate at which the data travels between the computer memory and the disk drive.

(ii) Seek Time or Random Access Time or Positioning Time

It is the time required to move the head to the desired cylinder or track is called as seek time or random access time or positioning time.

(iii) Rotational Latency

It is the time required to move the head to the desired sector by spinning the platter is called as rotational latency.

Now-a-days several megabytes are transferred in one second. There are several disks like floppy disks that are removable. They are enclosed in a plastic case and are cheaper. These disks are to be inserted into a disk drive for reading and writing purposes. They are relatively slow compared to hard disks. A disk drive is connected to a computer via a set of wires called as an I/O bus.

Merits

(i) Simple disks

(ii) Non-volatile memory

(iii) Less expensive

(iv) Huge storage capacity.

Demerits

(i) Greater access time in comparison to main memory

(ii) Prone to security.

2. Magnetic Tapes

These are the earliest secondary-storage devices. They are very slow in operation because they read and write data sequentially and random access to data is not possible. Now-a-days, they are used only to keep huge data which is not at all used but has to be kept as a record or history like census information of a country, stock market history, backup data etc.

The tape has to be inserted in a tape drive to perform read or write. It is a plastic ribbon with magnetic coating on the surface to store data. It consists of two spools one to wind and other to rewind the tape. The read-write head reads or writes on the tape using electromagnetic signals.

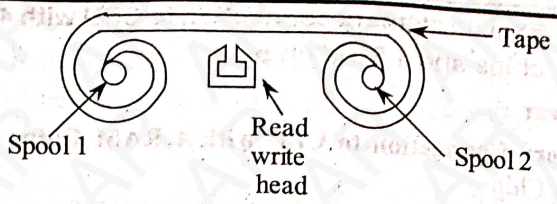


Figure (3): Magnetic Tape

The storage space provided by tapes is usually 20 GB to 200 GB.

Merits

(i) They are durable.

(ii) They can be used repeatedly.

(iii) They are reliable and inexpensive.

Demerits

(i) They cannot perform random access.

(ii) They can transmit data at a very low speed (i.e, slow).

Q37. Explain the various available formats and storage capacities of DVD.

Answer :

Formats of DVDs

DVD stands for Digital Versatile Disk. The different available formats of DVDs are as follows,

1. DVD-Video
2. DVD-ROM
3. DVD-RAM
4. DVD-R
5. DVD-RW
6. DVD-Audio.

1. DVD-Video

This type of format called as DVD format stores full-length high quality movies and its corresponding menus and extras. It is a digital video that allows users to navigate through other interactively. All movie videos and audio files are stored in the folder VIDEO-TS. It can incorporate multiple audio tracks for different languages and commentaries.

It is usually referred to as DVD and it has significantly minimized the use of VHS tapes at homes. Also, most of the education system is carried with the help of DVDs.

2. DVD-ROM

This type of format is called as read-only-media, wherein a disc once written can not be burnt or modified similar to DVD-video format, this format also incorporates computer files such as digital video. In addition to this, it supports enhanced multimedia game applications and instructional content in non-linear fashion. However, it is more or less analogous to CD-ROM but with high storage capacity.

3. DVD-RAM

This type of format is called as random access memory format. It is considered as special format wherein the disk exist inside a special cartridge. However, the disk does not have any access for read or write in any other drive. It is mostly implemented as peripheral storage device such as additional hard drive as it enables the user to re-record the files and also allows to perform random access. However, it is not widely used in schools.

4. DVD-R

This type of format is called as DVD-recordable format. It is a 4.7 GB single-layer DVD. It allows the user to burn (copying files) on to the DVD and enables it to be played in any computer. It is available in DVD-R (General) and DVD-R (Authoring) forms. It requires special software such as Apples DVD in order to create own DVD-videos. Also, it doesn't support addition/deletion of data i.e., the data in DVD-R format can not be overwritten.

5. DVD-RW

This type of format is called as DVD-rewritable. It can record files upto 4.7 GB and is a rewritable version of DVD-R. Here, the disk can be erased and burnt many times making it suitable for testing purposes. These discs are very viable and re-usuable but expensive.

6. DVD-Audio

This type of format is called as DVD-audio format. It is specific for storing audio-files on DVD. It can be used instead of music CDs.

Storage Capacities of DVDs

The storage capacity of DVDs can be different depending upon their structure. Basically most commonly used DVDs are of 4.7 GB because of the following reasons,

1. A red light laser with a wavelength of 635 mm is used to focus the light on to a smaller spot.
2. Smaller pits with a minimum length of 0.4 microns is used.
3. The distance between the tracks are shortened to 0.74 microns.

The use of the above design strategies will yield a DVD of capacity 4.7 G bytes.

However, the storage capacity of the DVD can be improved by applying different design strategies as follows,

1. Use of a Two-layered and Two-sided Disk

In this strategy, the DVD disk consists of two layers of pits and lands on both sides. Top of each layer contains data tracks. Both the lands and pits of the first layer are covered by a translucent material so as to function as a semi-reflector. The surface of this material is then designed with intended pits for storing the data. The top of the second layer (consisting of pits and lands) is covered with a reflective material. The data stored on the disk can be read by focusing the laser beam onto the desired layer. If the beam is focused on the first layer, the translucent material present in it reflects the sufficient light which helps in detecting the stored binary patterns. If the beam is focused onto the second layer, the reflective material of this layer reflects the light corresponding to the information stored in it. This type of design strategy leads to a DVD of 8.5 G bytes which is called DVD-9 in the standard.

2. Use of two Single-sided Disks

In this strategy, two single-sided and single layered disks are arranged in such a way that the structure looks like a sandwitch. This idea leads to a storage capacity of 9.4 G bytes. However, sandwitching of double-layered disks is also possible resulting in a total capacity of 17 G bytes.

The following are the different DVD standards available in the market.

Standard		Specification	Storage Space
1	DVD-5	Single-sided, single layer	4.7 GB
2.	DVD-9	Single-sided, double layer	8.5 GB
3.	DVD-10	Double-sided, single layer	9.4 GB
4.	DVD-18	Double-sided, double layer	17 GB
5.	HD-DVD	Double-sided, double layer	12 to 30 GB
6.	Blu-Ray DVD	Single-sided, double layer	25 to 50 GB

4.2.4 Associative Memory

Q38. What do you mean by associative memory? Give the hardware organization of associative memory.

Model Paper-I, Q9(a)

Answer :

Associative Memory

Associative memory is the memory that stores the content rather than the physical address. The stored data is usually accessed by referring it's address. It often turnouts to be difficult to access the data by means of it's address, especially when there are huge memories storing large volumes of data. Hence, it is an observed fact that, accessing by means of sample data is fast rather than it's address. Hence, the memories which employ this strategy for accessing the data is usually referred to as associative memory or content addressable memory.

Hardware Organization

Block diagram of associative memory is shown in following figure,

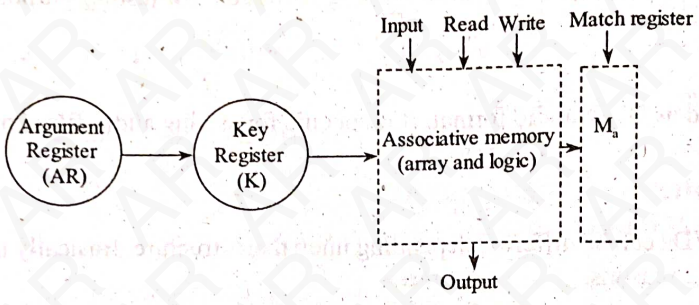
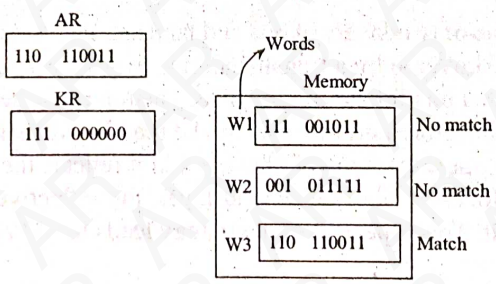


Figure: Block Diagram of Associative Memory

The block diagram of associative memory consists of an argument register, key register and a match register. At the central position, there exists a memory array and a logic of m words, with each word comprising n-bits. Argument register holds certain set of bits which are matched with the content of the memory parallely. If a match is observed, corresponding bit in the match register is set. The key register holds certain mask of bits depending on which the comparison is made. This entire process can be analyzed by considering the following example.

Suppose that argument register and key register consists of the following data.



Number of 1's in the key register determines the number of bits in argument register to be matched with the equivalent number of bits in the memory words. As the key register holds three consecutive 1's from the left most position, hence first three consecutive bits from the argument register are matched with each word parallely. First two matches fails and success is attained in the third match. Hence, a corresponding bit in the match register is set.

Q39. Write the match logic of associative memory.

Answer :

To derive a match logic for a word, the comparison algorithm for two binary numbers is used. In the first step, the argument present in X is compared with the bits that are stored in the word's cells. A word i is said to be equal to the argument in X only when $X_j = C_{ij}$ where j can have any value from 1 to n.

Two bits are said to be equal if they both have the values as 0's or 1's. The logical representation for the equality of two bits is shown below as a boolean function.

$$P_j = X_j C_{ij} + X_j' C_{ij}'$$

Thus,

$P_j = 1 \rightarrow$ when both the bits at position j are equal

$P_j = 0 \rightarrow$ When both the bits at position j are not equal.

The word i will be equal to the argument in X only when all the P_j variables have the values as '1'. Eventually, its related match bit 'M_i' will be set to 1.

Thus, the corresponding Boolean function for the match bit M_i is $M_i = P_1, P_2, P_3, ..., P_n$ which is actually the result after ANDing all the matched bits present in a word.

In the second step, the key bit K_j must be included in the comparison logic. The logic is that, when $K_j = 1$, the corresponding bits of X_{ij} and C_{ij} must be compared and when $K_j = 0$, they should not be compared. To do this, each term is ORed with K_j' as,

$$X_j + K_j' = \begin{cases} 1 & \text{if } K_j = 0 \\ X_j & \text{if } K_j = 1 \end{cases}$$

Thus, if $K_j = 1$,

$\Rightarrow \quad K_j' = 0 \Rightarrow P_j + 0 = X_j$

if $K_j = 0$,

$K_j' = 1 \Rightarrow P_j + 1 = 1$

Thus, the boolean function of match logic M_i for word i present in an associative memory will be,

$$M_i = (P_1 + K_1')(P_2 + K_2')(P_3 + K_3') \ldots (P_n + K_n')$$

If $K_j = 0$, then value of each term in the above expression is '1'.

If $K_j = 1$, then value of each term is equal to value of P_j.

For a match to occur, all the terms of the expression must be equal to 1.

Substituting the P_j's original definition in the Boolean function of M_i, we get,

$$M_i = \prod_{j=1}^{n} (X_j C_{ij} + X_j' C_{ij}' + K_j')$$

AND operation is performed on all the n terms to find out the product of n terms. The π symbol is used in the above boolean function. If there are 'w' number of words, then there will be 'w' number of match logics.

The corresponding circuit for matching a single word is shown in below figure.

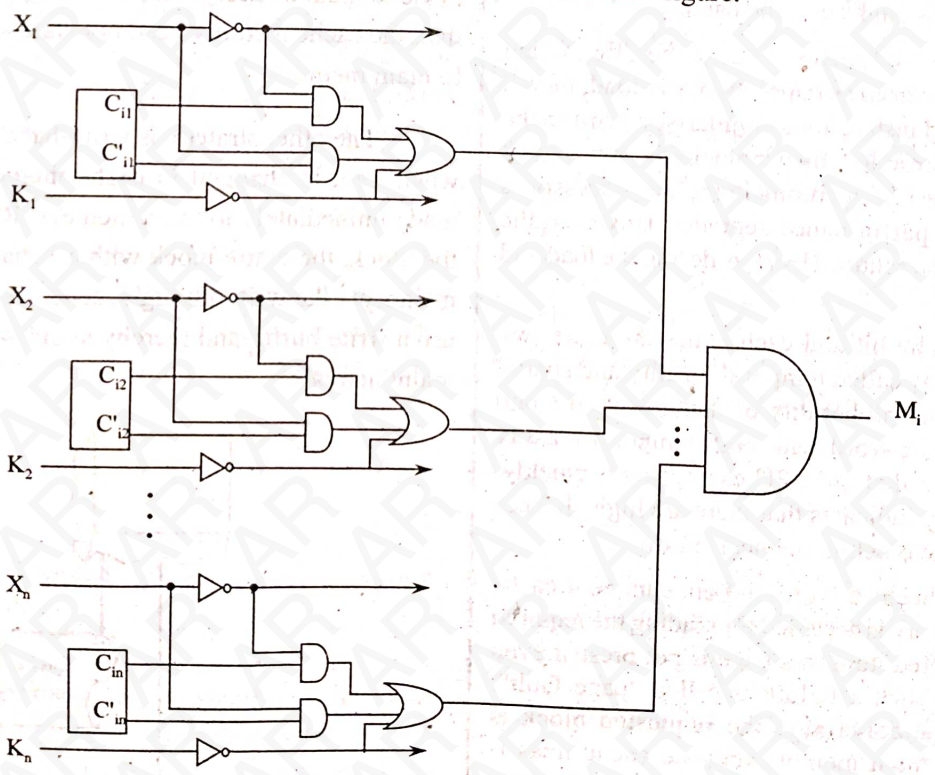


Figure: Match Logic for One Word of Associative Memory

In this circuit, there are n cells, each cell contains one OR gate and two AND gates. In addition to that, two inverters are also used for K_j and X_j, respectively. The OR gates of each of the cells present in the word are ANDed to produce M_i. If the value of M_i is 1, it is said that match has occurred. If the value of M_i is 0, then it is said that match is not occurred.

4.2.5 Cache Memory

Q40. Explain cache memory system with the help of a block diagram.

Answer :

Cache Memory System

Cache is the first level of memory in the memory hierarchy. It is based on the principle of locality of reference. The term cache is also used with those buffers that hold commonly occurring data. For example, file caches and name cache. Whenever CPU needs a word, it accesses the cache memory to check whether the requested word is available or not. If the requested word is present, then it is fetched from a cache. This is referred to as a "cache hit". On the contrary, the unavailability of the requested word in the cache leads to a cache miss. Since CPU references cache when it needs a word, it is necessary that the speed of cache must match with the speed of the CPU. On a cache miss, the block is a fixed-size collection of data that has the requested item. It is moved from main memory to cache and the CPU gets the requested item. Cache misses are handled by the hardware.

The time taken to process a cache miss is dependent on memory bandwidth and memory latency. Memory latency is the time taken to fetch the first word from a block. Memory bandwidth is the time taken to fetch the remaining words from the block in cache. Due to these factors, a cache miss consumes more CPU time than a cache hit. In case of inorder execution of processors, a cache miss can even pause the processor until the required block is not brought into the cache.

Cache misses affect the performance of the CPU. When a cache miss occurs for one type of operation, say load, then it blocks the subsequent load instructions. Similarly, when a cache miss occurs for a store instruction, then it blocks the subsequent store instructions. Because of such blocking, the processor is paused, due to which the performance degrades. However, the level of degradation can be reduced by re-ordering the load and store instructions.

In addition to cache hit and cache miss there are two important aspects of cache called temporal locality and spatial locality. They refer to the probability of using a word again in the near future. Thus, a word with high temporal locality is placed in the cache, so that the CPU can access it quickly. However, spatial locality indicates that there are high chances of using other data in the block in the near future.

When the CPU needs a word and encounters a cache miss, then the main memory is accessed for reading the required word. But, if the requested item in a page is not present even in main memory, then such a failure is called "page fault". Consequently, the page containing the requested block is entirely moved to the main memory and the cache miss is serviced. Page faults are handled in the software, (as they take too many clock cycles) that prevents the CPU from being paused while the page faults are processed. Instead of waiting, the CPU performs some other tasks till the pages are moved from disk to main memory.

Hit Ratio

Hit ratio is defined as the number of hits divided by the sum of hits and misses. Mathematically,

$$H_r = \frac{H}{H+M}$$

Where, H is the number of hits and M is the number of misses. Higher hit ratios are always desired as they affect the average memory access time.

Cache Reads/Writes

It is simple to cache data that are retrieved only for performing read operation because the copies of data in cache as well as main memory are identical and do not require any updation. However, caching writes is not as easy as caching reads. This is because the copies of data in cache and main memory must be made consistent, which is done by updating both copies (one in main memory and the other in cache), whenever changes are made to any one of the two copies.

There are two major strategies for keeping the copies consistent while caching the writes. The first strategy is write through cache. In this strategy, whenever the data in cache are updated, the respective copy in main memory is also updated. A write buffer can be used to store the changes that need to be made to main memory. The advantage of using write buffer is that, the cache memory need not wait while changes are made to main memory.

The other strategy is write-back cache. In this strategy, when item is changed in cache memory, the change is not made immediately to main memory. Rather, before replacing the block, the entire block with all changes is copied to main memory. Like write-through cache, write-back cache can also use a write buffer and thereby avoid waiting while writing to main memory.

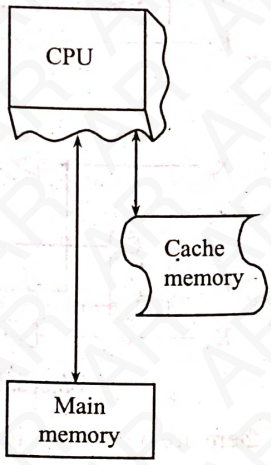


Figure: Cache Memory System

Q41. Describe briefly the following terms,

(i) Cache memory

(ii) Locality of reference

(iii) Cache line

(iv) Replacement algorithms

(v) Write hit

(vi) Dirty bit

(vii) Read miss

(viii) Write miss.

Answer :

(i) **Cache Memory**

It is a small memory placed in between the CPU and main memory. The principal requirement of cache memory is to speed up the processor, thereby allowing frequently used data to be stored in it. It is a compromise between the speed of processor and the main memory.

(ii) **Locality of Reference**

Locality of reference gives the location of frequently executed/required instructions. When any of the computer programs are considered, their execution time is mainly due to the execution of routines present in them. These routines are nothing but simple loops, multiple loops or branch instructions etc. Hence, in any computer programs, these instructions are of primary focus, which gets repeatedly executed all the time and rest of the instructions are executed rarely. This behaviour of computer programs are often referred to as locality of reference.

(iii) **Cache Line**

Cache line is the storage for data in cache memory in the form of blocks, which are addressable.

(iv) **Replacement Algorithms**

Replacement algorithms are the algorithms devised to relate to the choice of which data word to be eliminated. At certain times, the cache may be full and the processor may seek certain block of data which does not exist in cache. Hence, the processor collects the data word from the main memory and places its copy in the cache by eliminating already existing word (in the cache).

(v) **Write Hit**

Read or write hit refers to the purpose of requirement of data. Whenever a processor is in the need of certain data, it just issues either read/write signal. The control circuitry supported by cache memory receives this request and accordingly checks to see whether this required data exists with it or not. If the data exists, then it is known as data hit.

(vi) **Dirty Bit**

Dirty bit is a flag bit used to indicate the enhancement when a certain block of data is written only to cache memory. This enhancement is indicated by setting required bit of flag associated with the block.

(vii) **Read Miss**

Read miss is a situation that occurs when an assumption is made that, a processor in indulged in a read operation and it requires certain block of data. The processor initially searches the cache for the required block. If the block does not exists then such situation is called as read miss.

(viii) **Write Miss**

Write miss is a situation that occurs when an assumption is a made that, a processor is indulged in a write operation for which it searches the cache memory for the required word. If the data does not exist in cache, then such situation is referred as write miss.

Q42. Explain the following terms,

(i) Hit rate

(ii) Miss penalty.

Answer :

Cache mechanism plays an important role in increasing the efficiency of the processor, since it forms a major source to comprise with the speed of processor in effect with main memory. While dealing with cache mechanisms, there are two issues to be considered. These issues are "Hit rate" and "Miss penalty".

(i) **Hit Rate**

Hit rate is defined as a ratio of number of hits or successes to the total number of attempts made.

Hence,

$$\text{Hit rate} = \frac{\text{Number of successes or hits}}{\text{Total number of attempts made}}$$

In above ratio, the number of successes or hits are nothing but the number of times there was success to the processor in finding the desired data in the cache memory. It is estimated that, to increase overall performance of the system, the hit rate should be greater than 0.9.

(ii) Miss Penalty

Miss penalty is the time wasted in transferring the required data from the main memory to cache memory. For overall increase in the performance of a system, the hit ratio should be greater than 0.9. It itself reflects that there will be cases where the processor may fail to find the desired data in the cache memory. Hence, as a penalty, the processor should take initiative steps not only in locating the desired data in the main memory and its purpose but also it has to place the copy of same data in the cache memory. The term penalty in miss penalty indicates that, the processor has to resist the burden of stalling many of its time slots, while it was engaged in transferring the copy of required data into the cache memory. Hence, in order to enhance the overall performance of a system, the miss penalty should be reduced to greater extent.

There are many mechanisms adopted by the system programmers to avoid miss penalty. "Memory interleaving" is one of such mechanisms.

Q43. What are the different mapping techniques of cache memory? Describe each technique with suitable diagram.

Answer : Model Paper-III, Q9

Basically there are three mapping techniques of cache memory. They are,

(a) Direct mapping

(b) Associative mapping

(c) Set associative mapping technique.

In these three techniques, the cache size is considered as 128 blocks, each of which is divided into 16 words. The main memory is assumed to have 4K blocks with each block corresponding to 16 words.

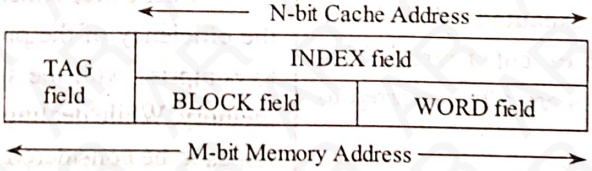
(a) Direct Mapping

A cache mapping technique that maps blocks of main memory into distinct cache lines is termed as a 'direct mapping' technique.

This mapping is defined using a single modulo division operation as follows,

Required cache line = (Block number) modulo (Total number of cache lines)

An address can also be used in this mapping technique. If cache memory contains 2^N words, then N-bits address is required to access the cache memory. If main memory contains 2^M words, then M-bits address is required to access the main memory. A main memory address contains two fields, a TAG field and an INDEX field. $(M-N)$ bits specify a TAG field and N-bits specify an INDEX field. The INDEX field is in turn divided into a BLOCK field and a WORD field. If the direct mapping organization uses a block size of w words, then the BLOCK field will contain $(N - \log_2 w)$ bits and the WORD field will contain $\log_2 w$ bits. All the words in a block will have the same TAG field.



In this mapping technique, M-bit address is used to access the main memory and N-bit INDEX field is used to access the cache memory. Consider a 512 × 12 cache memory and 32 k × 12 main memory, then,

Cache memory address = N-bits = 9-bits [$\because 2^9 = 512$]

Main memory address = M-bits = 15-bits [$\because 2^{15} = 32K$]

Index field = N-bits = 9-bits.

Tag field = $M - N$ bits = 15 – 9 = 6-bits

If w = 8 words,

Word field = $(\log_2 w)$ bits = $\log_2 8 = 3 \log_2 2 = 3$-bits

Block field = $(N - \log_2 w)$ bits = $(9 - \log_2 8) = 9 - 3 = 6$-bits

When a 15-bit address is read from the main memory, the 9-bits in the INDEX field are used to access a cache line (Each cache line contains a tag and its associated data). The tag stored in this cache line is then compared with the tag of the given 15-bit address. If they both match, the 3-bit word is used to identify one of the 8 words from that line. If they do not match, the complete block of data is read from the main memory and is replaced with the previous cache line (block).

Example

The organization of direct mapping in cache is shown in figure below.

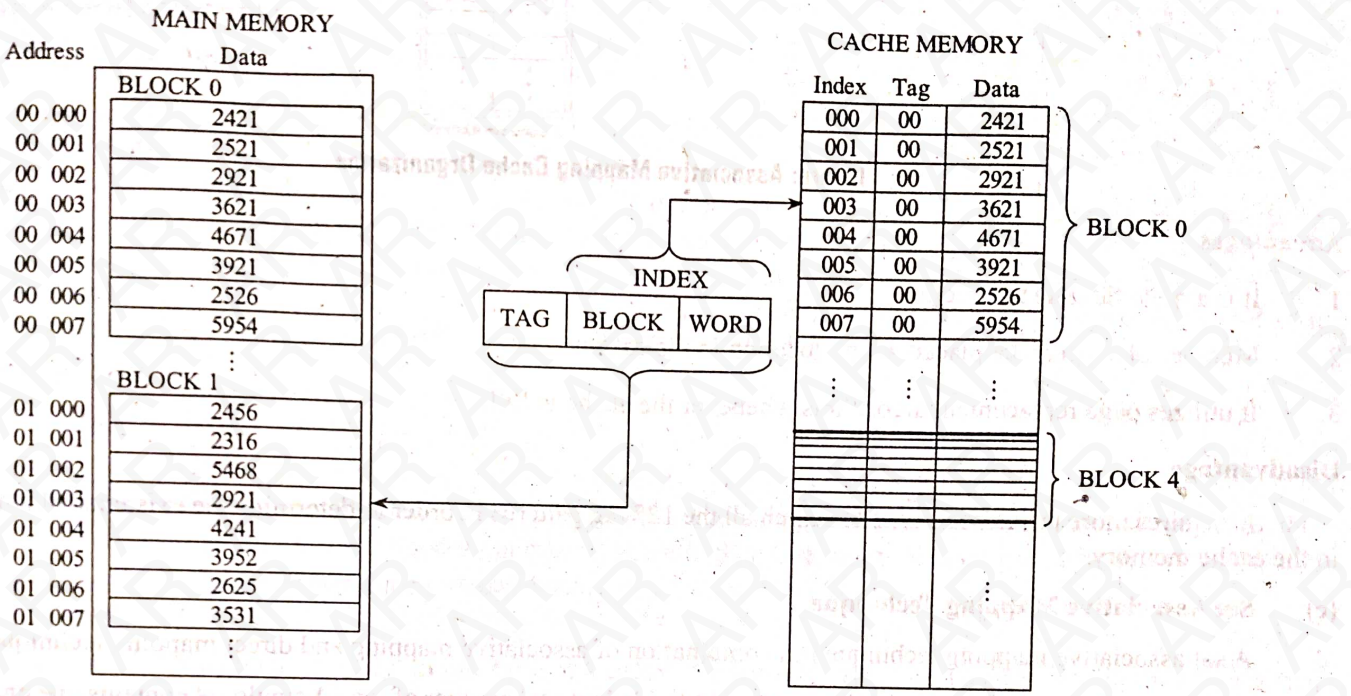


Figure: Direct Mapping Cache Organization

The above diagram shows that, cache memory stores BLOCK 0 from main memory. If any word from that block is required by CPU, then it can be easily read from the cache memory. Suppose, CPU requires a word from Block 1 with an address 01003. As the index address is 003, the cache is accessed with that address. Then, the TAG fields of each of these addresses are compared. They do not match as the required address contains 01 as its tag and, the cache line contains 00 as its tag. Thus, the complete BLOCK 0 in cache memory is replaced with BLOCK 1 by accessing it from main memory.

Advantage

It is very simple to implement.

Disadvantages

1. It is not effective.
2. It causes contention of data.
3. It utilizes page replacement algorithm to avoid contention, which itself gives rise to many other problems.

As the extent of disadvantages is greater than advantages. Hence, its usage is near to extinct.

(b) Associative Mapping Technique

Associative mapping technique is quite complicated when compared to direct mapping. Here, the memory address consists of two major fields i.e., *tag field* and *word field*, together contributing to 16 bits (where 12 bits correspond to tag field and 4 to word field). The following figure shows memory address of associative mapping.

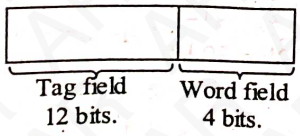


Figure: Memory Address in Case of Associative Mapping

The 12-bit tag field determines the main memory block to be placed in the cache memory. Hence, whenever processor intends to find any kind of data, it initially examines the tag field in order to gain the information on the existence of block in the cache memory. The following figure shows cache organization of associative mapping.

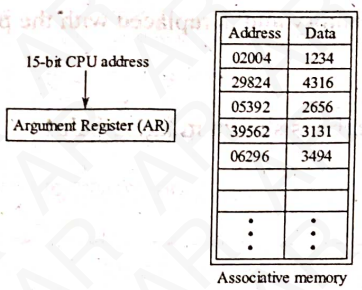


Figure: Associative Mapping Cache Organization

Advantages

1. It is a quite flexible technique.
2. Memory blocks can be placed at any location in the cache.
3. It utilizes page replacement algorithms whenever the cache is full.

Disadvantage

It requires more processor's time to search all the 128 tag patterns in order to determine the existence of a memory block in the cache memory.

(c) Set Associative Mapping Technique

A set associative mapping technique is a combination of associative mapping and direct mapping techniques.

In this mapping technique, the cache memory is divided into 's' number of sets. A single set contains one or more tag data pairs in one word of cache. Thus, an index address will point to 's' number of tag data pairs.

If the index address contains N-bits, then it can point to 2^N words and the length of a word, (L_w) will be,

$$L_w = \text{Number of sets} \times (\text{Number of tag bits} + \text{Number of bits in a data word}) \text{ bits}$$

Then, the size of the cache memory will be $(2^N \times L_w)$. Thus, a cache memory that employs set associative mapping contains 's' number of sets and each set will hold 's' number of memory words in a single word of cache.

Example

An organization of a 3-way set associative mapping cache is shown in figure below.

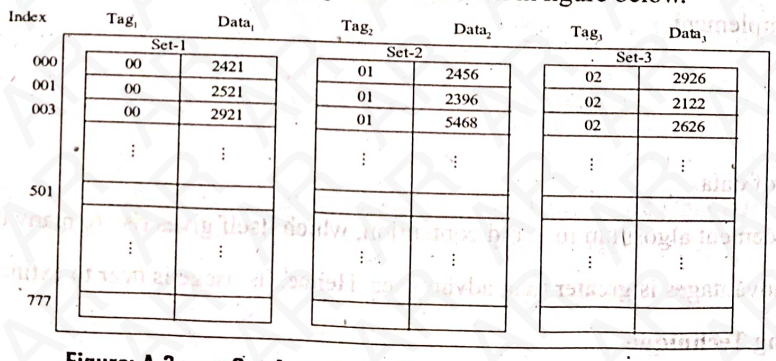


Figure: A 3-way Set Associative Mapping Cache Organization

In the above figure, a single index address will point to three different tag data pairs.

Consider a 512 × 12 cache memory and 32 k × 12 main memory, then,

Cache memory address = N-bits = 9-bits $[\because 2^9 = 512]$

Main memory address = M-bits = 15-bits $[\because 2^{15} = 32\text{ k}]$

Index field = N-bits = 9-bits

Tag field = $M - N$ bits = 15 – 9 = 6-bits

When a 15-bits address is generated by the CPU to access data, the cache memory is accessed using the 9-bit index address. Suppose, a data word at address 01003 is requested by the CPU, then the cache line with index address 003 is accessed. At this address, there are three different tag data pairs. Thus, an associative search is carried out to compare the tag value of the required data word with the three different tags in the current cache line. If a match is obtained, then its corresponding data word is accessed. If a match does not occur, and if set is full, then one of the tag data pairs in the set is replaced with a new tag data pair using any replacement algorithm among Random replacement algorithm, First-In First-Out replacement algorithm (FIFO) and Least Recently Used replacement algorithm (LRU).

Q44. Write about the following,

(a) Block replacement

(b) Writing to cache.

Answer :

(a) Block Replacement

Block replacement is a method in which the cache controllers replaces a missing block with another block using the requested data. This happens when a request for a block is made, and if a miss occurs. In order to replace the block, the controller must initially select the desired block for replacement. Replacing the block using direct-mapping scheme is simpler than fully associative and set associative schemes. This is because, the former requires selection of only one block (and only that selected block is replaced), whereas the latter requires selection of more than one block whenever a miss is encountered.

Block Replacement Methods

The three primary block replacement methods are,

(i) Random

In random block replacement, blocks that are to be replaced are randomly chosen. In order to achieve reproducible behavior, some systems generate pseudorandom number that are highly desired for debugging hardware. An advantage of this method is the ease of implementing it in hardware.

(ii) Least Recently Used (LRU)

Least recently used replacement is based on principle of locality of reference, which states that recently used blocks are highly probable to be used again and therefore cannot be replaced. Thus, instead of replacing those blocks, LRU replaces the block that has not been used for a longer period of time.

Since, LRU needs to track the recently used blocks, the method becomes too expensive whenever the number of blocks to be tracked increases.

(iii) First In First Out (FIFO)

To reduce the computation complexity of LRU, FIFO replaces the block that is most recently used.

Table below gives the cache misses for 1000 instructions for each of the blocks replacement methods. The comparison uses different cache sizes and associativities. From the table, it can be observed that random and LRU perform almost the same in case of the largest cache size. However, in case of the smallest cache size i.e., 16 kB, LRU performs much better than random and FIFO. Moreover, FIFO performs better than random.

Cache Size	Associatively								
	2-way			4-way			8-way		
	LRU	Random	FIFO	LRU	Random	FIFO	LRU	Random	FIFO
16 KB	114.1	117.3	115.5	111.7	115.1	113.3	109.0	111.8	110.4
64 KB	103.4	104.3	103.9	102.4	102.3	103.1	99.7	100.5	100.3
256 KB	92.2	92.1	92.5	92.1	92.1	92.5	92.1	92.1	92.5

Table: Comparison of LRU, Random, and FIFO for Different Cache Sizes

(b) Writing to Cache

When the CPU wants to write data to a block in the cache, first it checks whether the address or tag of the desired block matches any of the block addresses in cache. Then, depending on the result of address comparison, the CPU does necessary processing. That is, if the address matches, then the block is written to the cache. Otherwise, the block is not written to the cache. Writing to cache has some difficulties than reading from cache. Writing to cache takes more time than reading from cache. The major reason is the inability to check blocks and perform writes parallely. Another problem that CPU faces while writing a block is, it must write only the specified number of bytes. Usually, this number is between 1 and 8 bytes. On the other hand, while reading a block, the CPU can read as many bytes as required even after specifying a read limit.

Write Policies

Writing to cache can be done in one of the two basic ways, write-through and write-back.

1. **Write-through**

In write-through cache, whenever a write is made to cache, immediately the write is also made to main memory. While writing, the CPU is said to be paused or stalled, it does not do anything until the write is completed. The write stall can be reduced by using a write buffer that enables the CPU to switch to some other task till the write is completed.

2. **Write-back**

In write-back cache, the writes are first made to the block in cache. When the written block is about to be replaced, then the cache block is written entirely to main memory. The write-back cache method reduces the frequency of writing blocks generated during replacing a block, by using a status bit called "dirty bit". A dirty bit is assigned to every block in the cache. Whenever a block is modified then the block is said to be dirty. However, if a block is not modified, then it is said to be clean. The dirty/clean status of a block is indicated by the dirty bit. Thus, before writing blocks to the main memory, their dirty bits are checked. If the dirty bit of a block is set, then the block is written to the main memory, otherwise it is not written.

Write-back cache uses less memory bandwidth, saves power and uses few memory buses. Write-through cache also has some advantages over write-back. The main advantage is, write-through is easy to implement than write-back because the latter uses an additional bit-dirty bit. Another advantage is that, the cache is always in clean state. In addition to these advantages, write through is also capable of simplifying data coherency since the next lower level stores the latest copies of data.

Multiprocessors and I/O devices use both write-through and write-back cache. They use write-through for keeping the data in cache consistent with data in main memory. They use write-back for reducing memory traffic.

Write Miss

When the CPU wants to write to cache, it first checks if the element is present. A write miss occurs when the CPU does not find the desired element in cache. On a write miss, the block containing the desired element can be copied in two ways.

1. **Write Allocate**

In this method, the block is loaded in cache and a write is made to the element in cache and in main memory.

2. **No-write Allocate**

In this method, the block is not loaded in cache and no changes are made in cache. Rather, the element is changed only in main memory.

The write policies, write-back and write-through can use any of the write miss policies. Usually, write-through cache uses no-write allocate and write-back cache uses write-allocate. Choosing a write policy depends on the number of misses and hits for a set of operations. Consider a fully associative write-back cache that has many entries beginning with empty slots for the set of operations,

WriteMem[90];

WriteMem[90];

ReadMem[150];

WriteMem[150];

WriteMem[90];

If the CPU uses write allocate, then it encounters two misses and three hits. However, if the CPU uses no-write allocate then it encounters four misses and one hit. It is obvious for the CPU to use write allocate for the fully associative cache.

Q45. Describe briefly the following terms,

(i) Cache hits

(ii) Cache miss

(iii) Hit ratio

(iv) Average memory access time.

Answer : Model Paper-I, Q9(b)

(i) Cache Hits

When data is needed by CPU, it accesses the cache memory and checks if the requested data is present in it or not. If the requested data is found, it is called cache hits.

(ii) Cache Miss

When data is needed by CPU, it accesses the cache memory and checks if the requested data is present in it or not. If the requested data is not found, it is called cache miss.

(iii) Hit Ratio

For answer refer Unit-IV, Q40, Topic: Hit Ratio.

(iv) Average Memory Access Time (AMAT)

This method is considered as the best approach for measuring cache performance, because in the expansion of CPU execution time method, both Instruction Count (IC) and miss rate are hardware independent that may mislead the calculation.

In order to avoid this misleading scenario, average memory access time method is used that is defined as,

$$AMAT = Ht + M_r * M_p$$

Where,

Ht (Hit_time) = Time consumed by CPU to hit cache.

This method is an indirect measure of cache performance. Though this method is better than miss-rate, it cannot be considered as the perfect alternative to increase CPU execution time method.