Q1. What is Direct Memory Access (DMA)? Discuss DMA operation.

Direct Memory Access (DMA)

DMA is a method of transferring data to/from the computer's memory (RAM/ROM) and another part of the computer without processing it using the CPU. While most data that is input or output from your computer is processed by the CPU, some data does not require processing, or can be processed by another device. In these situations, DMA can save processing time and is a more efficient way to move data from the computer's memory to other devices.

For example, a sound card may need to access data stored in the computer's RAM, but since it can process the data itself, it may use DMA to bypass the CPU. Video cards that support DMA can also access the system memory and process graphics without needing the CPU.

The DMA Operation

The 8086 microprocessor receives bus requests through its HOLD pin and issues grants from the hold acknowledge (HLDA) pin. A request is made when a potential master sends a 1 to the HOLD pin. Normally, after the current bus cycle is complete the 8086 will respond by putting a 1 on the HLDA pin. When the requesting device receives this grant signal it becomes the master. It will remain master until it drops the signal to the HOLD pin, at which time the 8086 will drop the grant on the HLDA pin. There are many variations of DMA operations:

1. Single Transfer, 2. Block Transfer, 3. Burst-Block Transfer, 4. Repeated Single Transfer, 5. Repeated Block Transfer, and 6. Repeated Burst-Block Transfer.

DMA block transfer operation is discussed below:

DMA Block Transfer

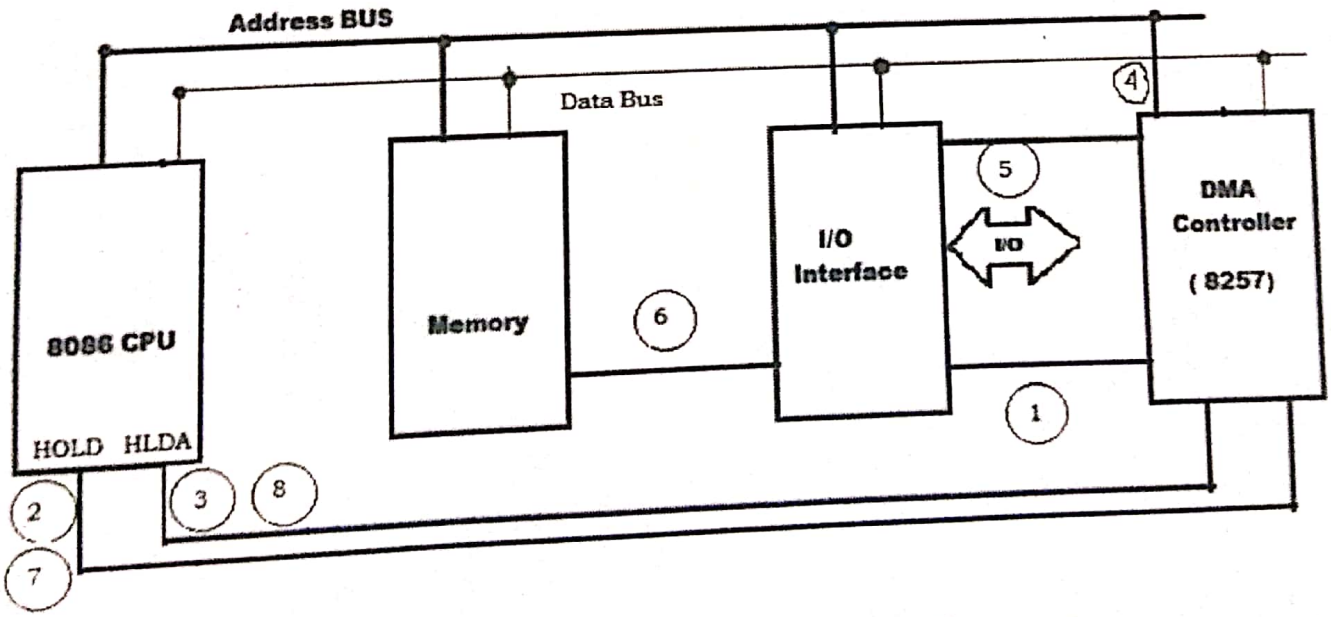

Fig. 1 Data Transfer with DMA Controller

During a block input byte transfer, the following sequence occurs as the data byte is sent from the interface to the memory:

1. The interface sends the DMA controller a request for DMA service.
2. A Bus request is made to the HOLD pin (active High) on the 8086 microprocessor and the controller gains control of the bus.
3. A Bus grant is returned to the DMA controller from the Hold Acknowledge (HLDA) pin (active High) on the 8086 microprocessor.
4. The DMA controller places contents of the address register onto the address bus.
5. The controller sends the interface a DMA acknowledgment, which tells the interface to put data on the data bus. (For an output it signals the interface to latch the next data placed on the bus.)
6. The data byte is transferred to the memory location indicated by the address bus.
7. The Bus request is dropped, the HOLD pin goes Low, and the controller relinquishes the bus.
8. The Bus grant from the 8086 microprocessor is dropped and the HLDA pin goes Low.

Q2. What is interrupt? Name the different types of interrupt. Explain the operation of 8259 Programmable Interrupt Controller (PIC) with block diagram.

Interrupt: An interrupt is a condition that causes the microprocessor to temporarily work on a different task and then return to its previous tasks. There two types of interrupt

1) **Hardware Interrupt:** Hardware interrupts occurs when a peripheral device asserts an interrupt signal to the interrupt input pin of the microprocessor.
2) **Software Interrupt:** Software Interrupts are initiated by instructions.

Intel 8259 Programmable Interrupt Controller (PIC): The Intel 8259 is a Programmable Interrupt Controller (PIC) designed for the hardware interrupts of i8085 and Intel i8086 microprocessors . The block diagram of 8259 contains eight functional blocks as shown in Fig. 2. They are,

1. Control logic
2. Read Write logic
3. Data bus buffer
4. Interrupt Request Register (IRR)
5. In-Service Register (ISR)
6. Interrupt Mask Register (IMR)
7. Priority Resolver (PR)
8. Cascade buffer.

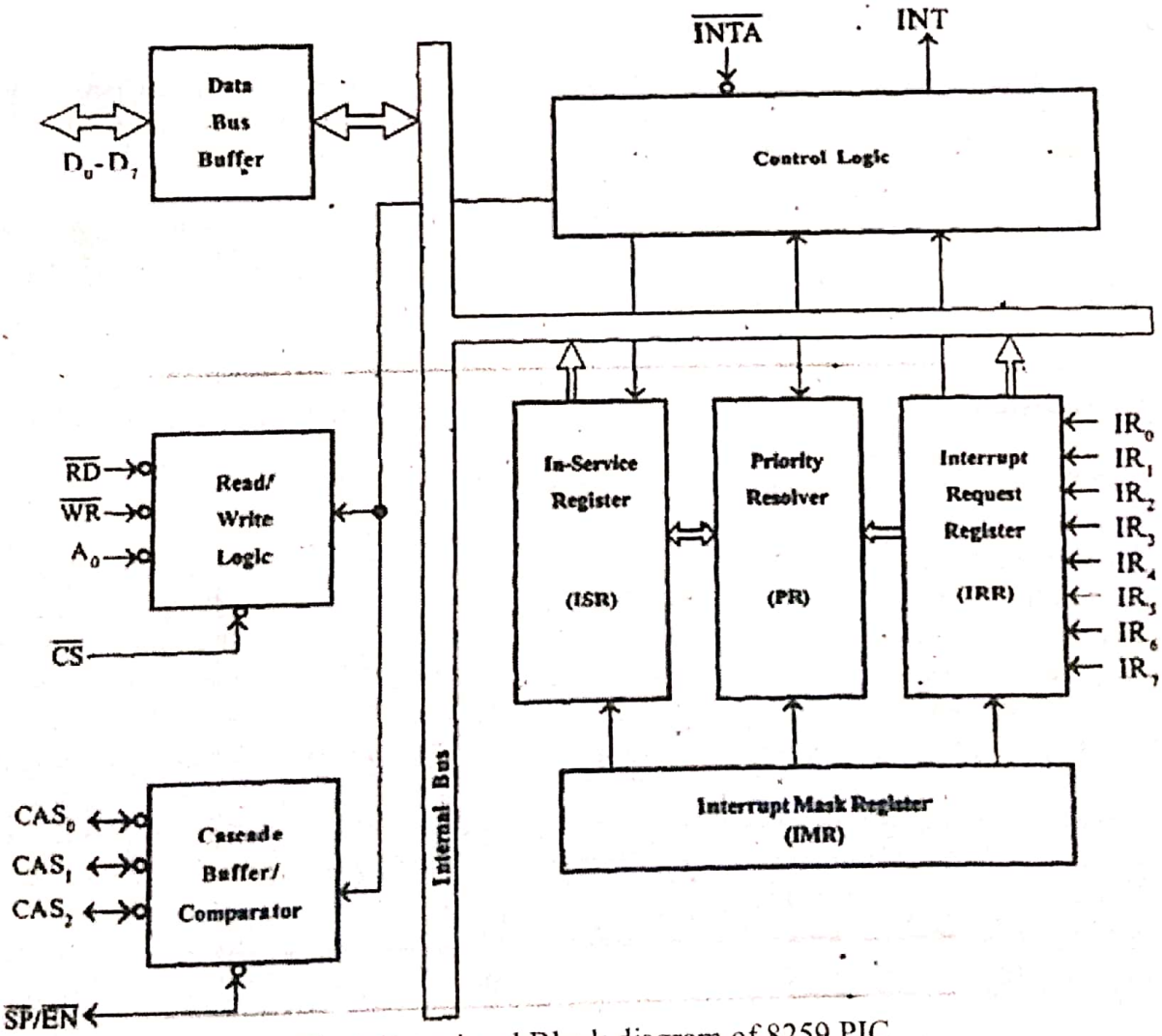

Fig. 2 Functional Block diagram of 8259 PIC

1. **Control logic:**

INT (Interrupt)
This output goes directly to the CPU interrupt input. The VOH level on this line is designed to be fully compatible with the 8080A, 8085A and 8086 input levels.
INTA (Interrupt Acknowledge)
INTA pulses will cause the 8259A to release vectoring information onto the data bus. The format of this data depends on the system mode of the 8259A.

2. **Read/Write Control Logic:** The function of this block is to accept output commands from the CPU. It contains the Initialization Command Word (ICW) registers and Operation Command Word (OCW) registers which store the various control formats for device operation. This function block also allows the status of the 8259A to be transferred onto the Data Bus.

   CS (CHIP SELECT)
   A LOW on this input enables the 8259A. No reading or writing of the chip will occur unless the device is selected.
   WR (WRITE)
   A LOW on this input enables the CPU to write control words (ICWs and OCWs) to the 8259A.
   RD (READ)
   A LOW on this input enables the 8259A to send the status of the Interrupt Request Register (IRR), In Service Register (ISR), the Interrupt Mask Register (IMR), or the Interrupt level onto the Data Bus.
   A0
   This input signal is used in conjunction with WR and RD signals to write commands into the various command registers, as well as reading the various status registers of the chip. This line can be tied directly to one of the address lines.

3. **The data bus buffer:** The data bus and its buffer are used for the following activities.
   1. This 3-state, bidirectional 8-bit buffer is used to interface the 8259 to the system Data Bus.
   2. The processor sends control word to data bus buffer through D0-D7
   3. The processor read status word from data bus buffer through D0-D7

4. **Interrupt Request Register (IRR):** The IRR is used to store all the interrupt levels which are requesting service. The IRR has eight input lines (IR0-IR7) for interrupts. When these lines go high, the request is stored in IRR. It registers a request only if the interrupt is unmasked. Normally IR0 has highest priority and IR7 has the lowest priority. The priorities of the interrupt request input are also programmable.
5. **In-Service Register (ISR)**: The ISR is used to keep track of which interrupt is currently being serviced.
6. **Interrupt Mask Register (IMR):** The IMR stores the bits which mask the interrupt lines to be masked. The IMR operates on the IRR. Masking of a higher priority input will not affect the interrupt request lines of lower quality.
7. **Priority Resolver:** This logic block determines the priorities of the bits set in the IRR. The highest priority is selected and stored into the corresponding bit of the ISR during INTA pulse.

8. **Cascade Buffer:** The cascade buffer is used to expand the interrupts of 8259.

Q3. What is peripheral? What is interface? Draw the pin diagram of 8255 Programmable Peripheral Interface (PPI) and write down the function of pins.

Ans:

Peripheral:

A computer device, such as a CD-ROM drive or printer, that is not part of the essential computer, i.e., the memory and microprocessor is called peripheral. Peripheral devices can be external -- such as a mouse, keyboard, printer, monitor, external Zip drive or scanner -- or internal, such as a CD-ROM drive, CD-R drive or internal modem. Internal peripheral devices are often referred to as integrated peripherals.

Interface:

The point of interaction or communication between a computer and any other peripherals.

8255 Programmable Peripheral Interface (PPI)

The 8255A is a programmable peripheral interface (PPI) device designed for use in Intel microprocessor systems. Its function is that of a general purposes I/O component to Interface peripheral equipments to microcomputer system bus. The functional configuration of the 8255A is programmed by the software so that normally no external logic is necessary to interface peripheral devices. Figure 3 shows the pinouts of 8255 PPI.

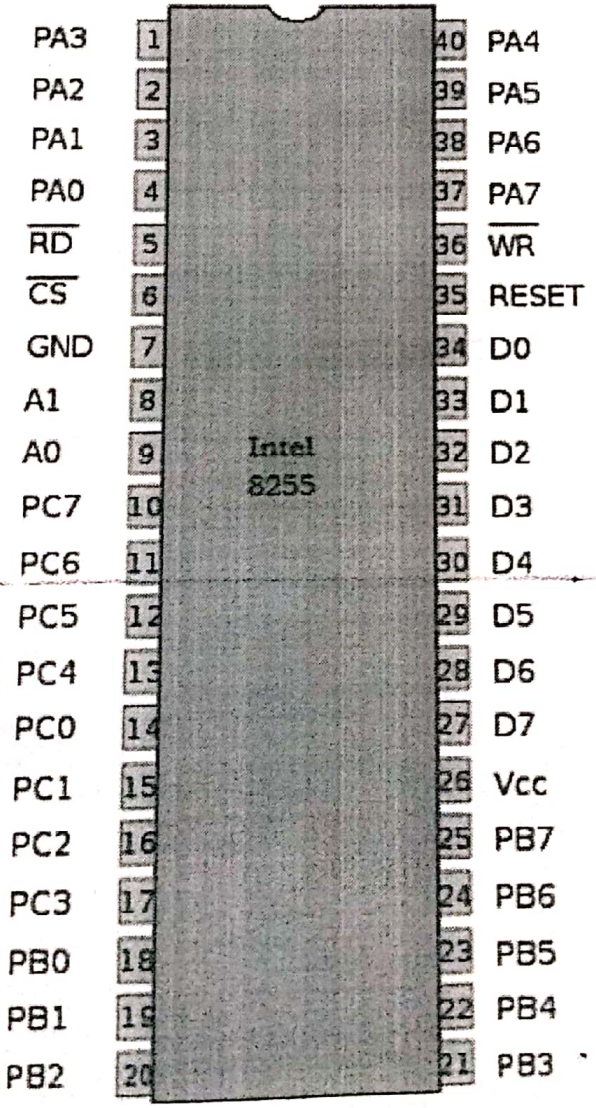

Fig. 3 Pinouts of 8255 PPI

D0 - D7: These are the data input/output lines for the device. All information read from and written to the 8255 occurs via these 8 data lines.
CS (*Chip Select Input*): If this line is a logical 0, the microprocessor can read and write to the 8255.
RD (Read Input): Whenever this input line is a logical 0 and the RD input is a logical 0, the 8255 data outputs are enabled onto the system data bus.
WR (Write Input): Whenever this input line is a logical 0 and the CS input is a logical 0, data is written to the 8255 from the system data bus
A0 - A1 (*Address Inputs*) : The logical combination of these two input lines determines which internal register of the 8255 data is written to or read from. The ports of 8255 can be decoded by A0 and A1 as:

A1	A0	
0	0	Port A
0	1	Port B
1	0	Port C
1	1	Control Register

RESET: The 8255 is placed into its reset state if this input line is a logical 1. All peripheral ports are set to the input mode.
PA0 - PA7, PB0 - PB7, PC0 - PC7: These signal lines are used as 8-bit I/O ports. They can be connected to peripheral devices. The 8255 has three 8 bit I/O ports and each one can be connected to the physical lines of an external device. These lines are labeled PA0-PA7, PB0-PB7, and PC0-PC7. The groups of the signals are divided into three different I/O ports labeled port A (PA), port B (PB), and port C (PC).

Q4. Explain the functions of bits of control register of 8255 PPI.

Ans:

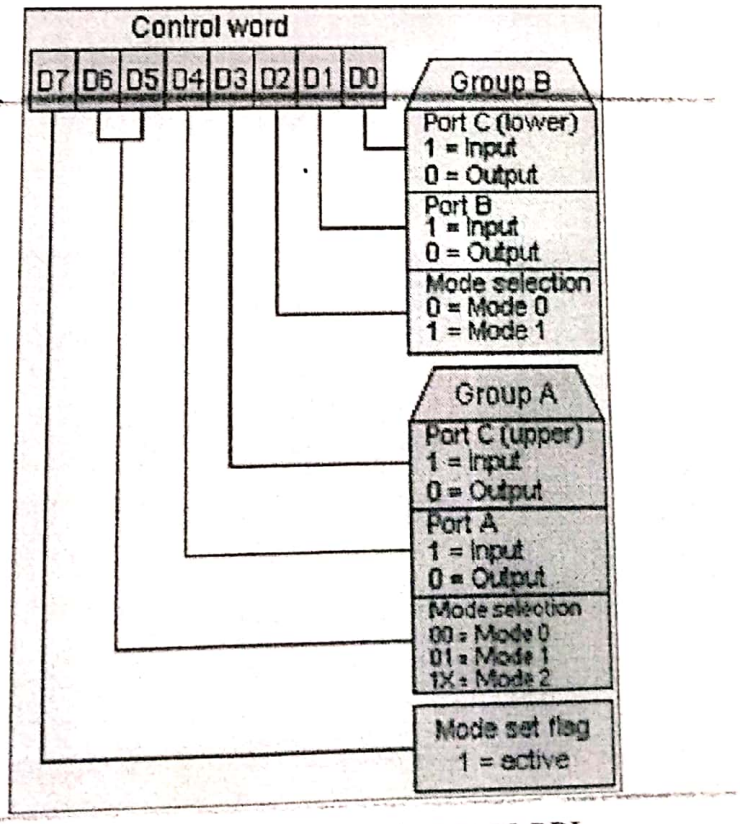

Fig. 4. The control Word of 8255 PPI

Configuration control byte

When bit 7 is set to 1, the command byte operates in the following way:

Bit 7: Control byte function (1=Configuration control byte)

Bit 5-6} Operating Mode

Bit 6	Bit 5	Mode
0	0	Mode 0
0	1	Mode 1
1	x	Mode 2

Mode 0: Mode 0: Ports A and B operate as either inputs or outputs and Port C is divided into two 4-bit groups either of which can be operated as inputs or outputs

Mode 1: Same as Mode 0 but Port C is used for handshaking and control

Mode 2: Port A is bidirectional (both input and output) and Port C is used for handshaking. Port B is not used.

Bit 4: Port A I/O Status (0: Output, 1: Input)

Bit 3: Port C (upper) I/O Status (0: Output, 1: Input)

Bit 2: Operating Mode selection (0: Mode 0, 1: Mode 1)

Bit 1: Port B I/O Status (0: Output, 1: Input)

Bit 0: Port C (lower) I/O Status (0: Output, 1: Input)

Q5. What is microcontroller? Draw the basic block of microcontroller.

Ans: A microcontroller is a small, low-cost computer-on-a-chip which usually includes:

- An 8 or 16 bit microprocessor (CPU).
- A small amount of RAM.
- Programmable ROM and/or flash memory.
- Parallel and/or serial I/O.
- Timers and signal generators.
- Analog to Digital (A/D) and/or Digital to Analog (D/A) conversion.

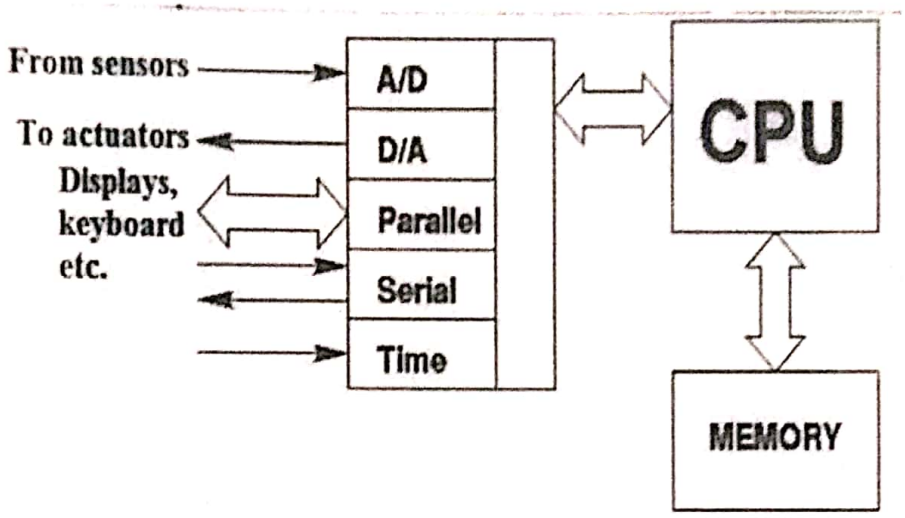

Fig. 5 Block Diagram of Microcontroller

Q6. What are the applications of Microcontroller?

Ans: Applications of Microcontroller

- Building control
  - Access control
  - Temperature sensing
  - Lighting
  - Fire detection
  - Camera
  - Cable TV tuner
  - Toys
  - Fax machine
  - Microwave oven
  - Weather Control
- Industrial control
  - Process control
- Instrumentation
  - Industrial instrumentation
- Metering
  - Handheld metering systems
- Motor speed control
  - AC/DC motor control
  - Steppers
- Security System

Q7. What are the difference between Microcontroller, Microprocessor?

Ans: The Difference between microprocessor and microcontroller are as follows

Microprocessor is an IC which has only the CPU inside them i.e. only the processing powers such as Intel's Pentium 1,2,3,4, core 2 duo, i3, i5 etc. These microprocessors don't have RAM, ROM, and other peripheral on the chip. A system designer has to add them externally to make them functional. Application of microprocessor includes Desktop PC's, Laptops, notepads etc.

But this is not the case with Microcontrollers. Microcontroller has a CPU, in addition with a fixed amount of RAM, ROM and other peripherals all embedded on a single chip. At times it is also termed as a mini computer or a computer on a single chip. Today different manufacturers produce microcontrollers with a wide range of features available in different versions. Some manufacturers are ATMEL, Microchip, TI, Freescale, Philips, Motorola etc.

Microcontrollers are designed to perform specific tasks. Specific means applications where the relationship of input and output is defined. Depending on the input, some processing needs to be done and output is delivered. For example, keyboards, mouse, washing machine, digicam, pendrive, remote, microwave, cars, bikes, telephone, mobiles, watches, etc. Since the applications are very specific, they need small resources like RAM, ROM, I/O ports etc and hence can be embedded on a single chip. This in turn reduces the size and the cost.

Microprocessor find applications where tasks are unspecific like developing software, games, websites, photo editing, creating documents etc. In such cases the relationship between input and output is not defined. They need high amount of resources like RAM, ROM, I/O ports etc.
The clock speed of the Microprocessor is quite high as compared to the microcontroller. Whereas the microcontrollers operate from a few MHz to 30 to 50 MHz, today's microprocessor operate above 1GHz as they perform complex tasks.

Comparing microcontroller and microprocessor in terms of cost is not justified. Undoubtedly a microcontroller is far cheaper than a microprocessor. However microcontroller cannot be used in place of microprocessor and using a microprocessor is not advised in place of a microcontroller as it makes the application quite costly. Microprocessor cannot be used stand alone. They need other peripherals like RAM, ROM, buffer, I/O ports etc and hence a system designed around a microprocessor is quite costly.