Quantum Series®
ENGINEERING

QUANTUM *Series*

Semester - 6 **Electrical & Electronics Engineering**

Microprocessor & Microcontroller

Session 2020-21
Even Semester

- Topic-wise coverage of entire syllabus in Question-Answer form.
- Short Questions (2 Marks)

Includes solution of following AKTU Question Papers:
2015-16 • 2016-17 • 2017-18 • 2018-19

Scanned by TapScanner

CONTENTS

KEE 602 : Microprocessor and Microcontroller

UNIT-1 : INTRODUCTION TO MICROPROCESSOR (1–1 C to 1–46 C)

Introduction to Microprocessor: Microprocessor architecture and its operations, Memory, Input & output devices, The 8085 MPU-architecture, Pins and signals, Timing Diagrams, Logic devices for interfacing, Memory interfacing, Interfacing output displays, Interfacing input devices, Memory mapped I/O. Basic Programming concepts: Flow chart symbols, Data Transfer operations, Arithmetic operations, Logic Operations, Branch operation, Writing assembly language programs, Programming techniques: looping, counting and indexing. Additional data transfer and 16 bit arithmetic instruction, Logic operation: rotate, compare, counter and time delays, 8085 Interrupts.

UNIT-2 : INTEL 8086 MICROPROCESSOR (2–1 C to 2–22 C)

Internal architecture (Bus Interface Unit, Execution unit, Pipelining, Register organization), Pin Diagram, Memory addressing, Physical memory organization, Interrupts (hardware & software interrupts).

UNIT-3 : PERIPHERAL DEVICES (3–1 C to 3–52 C)

8237 DMA Controller, 8255 programmable peripheral interface, 8253/8254programmable timer/counter, 8259 programmable interrupt controller, 8251 USART and RS232C. Fundamental of Programming: Program structure & programming techniques for microprocessors, 8085 Addressing modes, 8085 Instruction set, Assembly language programming of 8085 microprocessor with examples (arithmetic operations on 8-bit numbers – add, subtract, multiply, divide, square & square root etc, largest/smallest number; ascending/descending order).

UNIT-4 : 8051 MICROCONTROLLER BASICS (4–1 C to 4–20 C)

Inside the Computer, Microcontrollers and Embedded Processors, Block Diagram of 8051, PSW and Flag Bits, 8051 Register Banks and Stack, Internal Memory Organization of 8051, IO Port Usage in 8051, Types of Special Function Registers and their uses in 8051, Pins of 8051. Memory Address Decoding, 8031/51 Interfacing With External ROM And RAM. 8051 Addressing Modes.

UNIT-5 : ASSEMBLY PROGRAMMING OF 8051 (5–1 C to 5–24 C)

Introduction to 8051 assembly programming, Assembling and running an 8051 program, Data types and Assembler directives, Arithmetic, logic instructions and programs, Jump, loop and call instructions, IO port programming. Programming 8051 Timers. Serial Port Programming, Interrupts Programming, Comparison of Microprocessor, Microcontroller, PIC and ARM processors and their application areas.

SHORT QUESTIONS (SQ-1 C to SQ-13 C)

SOLVED PAPERS (2015-16 TO 2018-19) (SP-1 C to SP-12 C)

Scanned by TapScanner

1 UNIT

Introduction to Microprocessor

CONTENTS

Part-1 : Introduction to Microprocessor : Microprocessor Architecture and its Operations, Memory, Input and Output Devices ..... **1–2C to 1–9C**

Part-2 : The 8085 MPU-Architecture, Pins and Signals, Timing Diagrams ..... **1–9C to 1–22C**

Part-3 : Logic Devices for Interfacing, Memory Interfacing, Interfacing Output Displays, Interfacing Input Devices, Memory Mapped I/O ..... **1–22C to 1–29C**

Part-4 : Basic Programming Concepts : Flow Chart Symbols, Data Transfer Operations, Arithmetic Operations, Logic Operations, Branch Operation ..... **1–29C to 1–37C**

Part-5 : Writing Assembly Language Programs, Programming Techniques : Looping, Counting and Indexing, Additional Data Transfer and 16-bit Arithmetic Instruction, Logic Operation : Rotate, Compare ..... **1–37C to 1–42C**

Part-6 : Counter and Time Delays, 8085 Interrupts ..... **1–42C to 1–45C**

PART-1

Introduction to Microprocessor : Microprocessor Architecture and its Operations, Memory, Input and Output Devices.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.1. **What do you understand by microprocessor ?**

Answer

1. The microprocessor is a programmable integrated device that has computing and decision-making capability similar to that of central processing unit (CPU).
2. A microprocessor is designed to perform arithmetic and logic operations that make use of small number-holding areas called registers.
3. Typical microprocessor operations include adding, subtracting, comparing two numbers and fetching numbers from one area to another.
4. A microprocessor typically serves as a central processing unit microprocessor in a computer system.

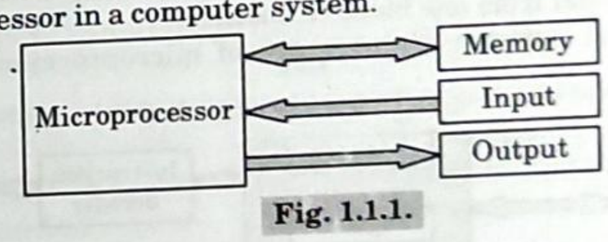


Fig. 1.1.1.

Que 1.2. **Explain the different types of microprocessor.**

Answer

There are three types of microprocessor :

A. Single chip microprocessors : It contains ROM, RAM, I/O ports, Clock and Timer.

B. General purpose microprocessor :

1. It contains ALU with more than one registers that functions as accumulator, a control unit, an instruction decoder which handles a fixed instruction set and special and general purpose registers which varies significantly from one microprocessor to other.

Scanned by TapScanner

2. A microprocessor may have an internal stack of fixed length or use external memory for stack. The general purpose microprocessor is available in word length of 1, 4, 8, 16, 32 and 64-bits.

C. Bit slice microprocessor :

1. The word bit slice comes from the way someone assembles a microprocessor of desired word width by adding number of bit-slice units.
2. The term slice comes from the fact that part like ALU, multiplexer, sequencer, and other part needed for custom building a CPU can be connected in parallel to work with 8-bit word, 16-bit word, or 32-bit word.
3. In other sense, we can say that designer can add as many slices as needed for a particular application. A popular bit slice unit is AMD 2900 series.
4. This microprocessor divides the functions of ALU, general purpose and special purpose registers and control unit into several IC's.
5. Each register of ALU package is essentially equivalent to 2 or 4-bit wide slice of registers and ALU of microprocessor.

Que 1.3. Explain the microprocessor architecture and operation of its components.

Answer

A. Architecture :

1. Microprocessor architecture defines suitable placement of its various functional blocks in the form of required circuitry for efficient flow of data and result from one block to another.
2. The general purpose architecture of microprocessor is shown in Fig. 1.3.1.

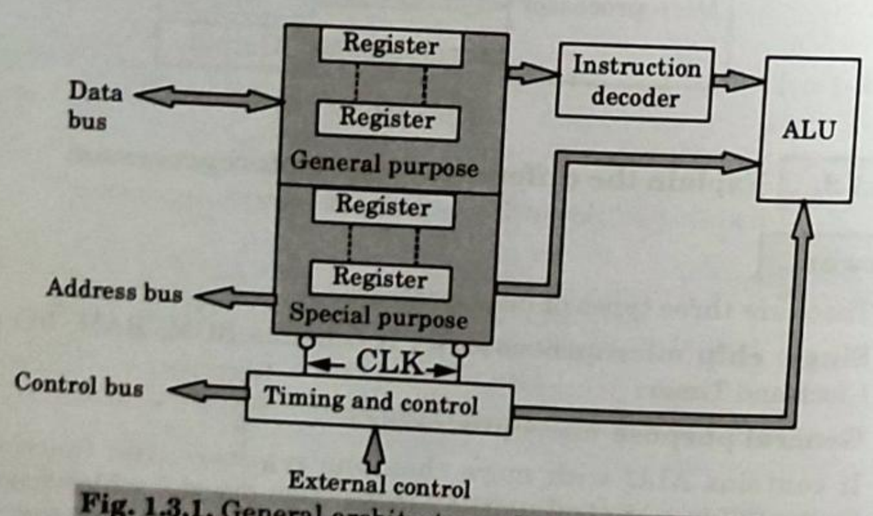


Fig. 1.3.1. General architecture of a microprocessor.

B. Architecture of microprocessor contains :

1. ALU (Arithmetic logical unit) :

i. It consists of an adder, an accumulator, a temporary register, and a shift register and a status register.

ii. This ALU unit performs various arithmetic and logical operation.

2. General purpose register :

i. In the microprocessor, there are 8-bit general purpose register or as a 16-bit register pairs, when used in register pair mode.

ii. These are used for both storing data as well as the address.

3. Special purpose register :

i. This consist of accumulator, program counter (PC), stack pointer (SP) and status flag register.

ii. These registers are used for some specific applications designated by the manufacturers.

4. Instruction decoder : This receives the contents of instruction register and develops control signals that enable data paths necessary to execute the instruction.

5. Timing and control unit :

i. This unit controls and synchronizes all the operations inside and outside the microprocessor.

ii. The timing and control signals are derived from the master clock.

iii. The control unit also accepts the control signals generated by the other devices associated with microprocessor system.

6. Address bus :

i. This is used for transmitting address information.

ii. The address bus will normally contain 16-bits to provide for addressing and addressing capability of up to 64 kB of memory.

7. Control bus : This comprises of various signal lines used for carrying synchronization signals. The microprocessor uses such lines for providing timing signals.

Que 1.4. Explain CPU and buses of the microprocessor.

Answer

A. CPU (Central processing unit) :

1. Its purpose is to fetch binary coded instruction from memory then decodes the fetched signal and generates the control signal required to execute the instructions.
2. It executes instructions, controls and directs the inputs, informs ALU (Arithmetic and logic Unit) for processing, arranges the storage and directs the data to the output device.

Scanned by TapScanner

3. It is responsible for directing the flow of instruction and data within control signals.
4. It contains the necessary logic to interpret instructions and to generate the signals necessary for the execution of those instructions.

B. System buses : Buses are data communication path over which information is transferred a byte or word at a time. The system Bus consists of three types of buses :

i. Data bus :

1. A type of bus that is used to transfer data from memory to CPU and I/O device to memory is called Data Bus.
2. Data bus is the bidirectional bus *i.e.*, it can communicate in two ways, but in one direction at a time.
3. It handles the transfer of data and instructions.
4. It carries data (Operands) to and from the CPU and memory as required.
5. It is also used to transfer data between memory and I/O devices during input output operations.

ii. Address bus :

1. A type of bus that is used to carry the address (not data) is called Address Bus.
2. It carries address between memory and CPU (central processing unit) and between memory and I/O devices.
3. An address is defined as a label, symbol, or other set of characters used to designate a location or register where information is store.
4. Before data or instructions can be written into or read from memory by the CPU or I/O sections, an address must be transmitted to memory over the address bus.
5. The number of lines on the bus determines the number of addressable memory elements.

iii. Control bus :

1. Control bus is a type of bus that carries control instructions to run operates hardware.
2. The control bus is used by the CPU (central processing unit) to direct and monitor the actions of the other functional areas of the computer.
3. It is used to transmit a variety of individual signals (read, write, interrupt, acknowledge) and coordinate the operations of the computer.
4. The size of control bus is from 8 to 16 bits.

Que 1.5. **List the components of computer and explain each in brief. What is the difference between a microprocessor and a CPU ?**

AKTU 2018-19, Marks 07

Answer

A. Components of computer : Computer systems consist of three components : Central processing unit, Input devices and Output devices.

1. Central processing unit :

i. The Central Processing Unit (CPU) is called "the brain of computer" as it controls operation of all parts of computer.

ii. It consists of two components :

a. Arithmetic logic unit (ALU) : Data entered into computer is sent to RAM, from where it is then sent to ALU, where rest of data processing takes place. All types of processing, such as comparisons, decision-making and processing of non-numeric information take place.

b. Control unit : This part of CPU extracts instructions, performs execution, maintains and directs operations of entire system.

2. Input and output devices : Input devices provide data input to processor, which processes data and generates useful information that's displayed to the user through output devices. This is stored in computer's memory.

B. Difference

S. No.	Microprocessor	CPU
1.	In microprocessor all the logic circuitry, including the control unit are in the form of single chip.	In CPU all the logic circuitry, including the control unit are not in the form of single chip.

Que 1.6. **What do you understand by memory ? Also give its classification.**

Answer

A. Memory : A memory unit is a device to which binary information is transferred for storage and from which information is retrieved when needed for processing.

Scanned by TapScanner

B. Classification :

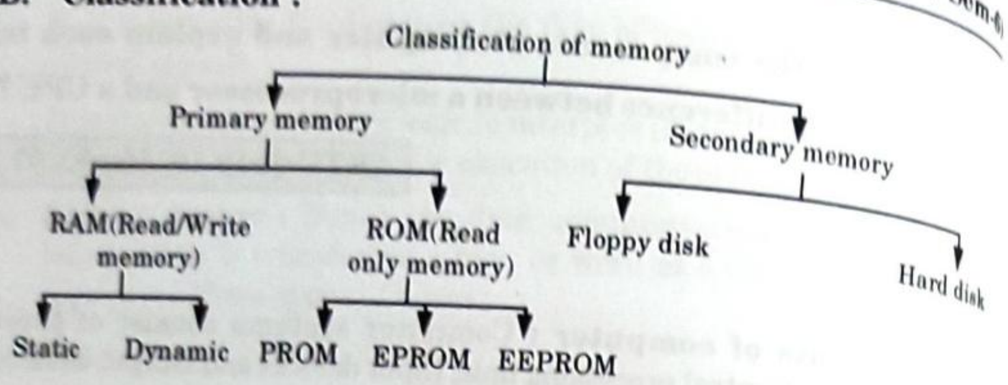


Fig. 1.6.1.

i. RAM :

1. It is the memory device which stores the binary information permanently.
2. It allows only retrieval of permanently stored data and does not permit modifications of the stored information during normal operation.
3. ROMs are non-volatile memories, *i.e.*, the stored data are not lost even when the power supply is OFF and refresh operation is not required.

ii. ROM :

1. Read-only memory (ROM) is a class of storage media used in computers and other electronic devices.
2. Because it cannot (easily) be written to, its main use lies in the distribution of firmware (software that is very closely related to hardware and not likely to need frequent upgrading).
3. Modern semiconductor ROMs typically take the shape of IC packages.
4. ROM can only be read from, but all ROMs allow data to be written into them at least once, either during initial manufacturing or during a step called "programming".

iii. PROM :

1. A programmable read-only memory (PROM) is a form of digital memory where the setting of each bit is locked by a fuse or antifuse. Such PROMs are used to store programs permanently.
2. A typical PROM comes with all bits reading as 1, burning fuse during programming causes its bit to read as 0.
3. The memory can be programmed just once after manufacturing by "blowing" the fuses, which is an irreversible process. Blowing a fuse opens a connection while blowing an antifuse closes a connection.

iv. EPROM :

1. An EPROM, or erasable programmable read-only memory, is a type of computer memory chip that retains its data when its power supply is switched off. In other words, it is non-volatile.

2. Once programmed, an EPROM can be erased only by exposing it to strong ultraviolet light.

v. EEPROM :

1. An EEPROM (also called an E2PROM) or electrically erasable programmable read-only memory is a non-volatile storage chip used in computers and other devices to store small amounts of volatile (configuration) data.
2. The main advantage of EEPROMs over EPROMs is that they are erased electrically instead of by ultraviolet light; this is faster and can be done in circuit.

vi. Flash memory :

1. It is a variation of EEPROM.
2. The difference between the flash and EEPROM is in erasure procedure.
3. EEPROM can be erased at a register level, but the flash must be erased either entirely or block level.

Que 1.7. **What do you understand by input/output devices of microprocessor ?**

Answer

1. Input/Output devices are the devices by which microprocessor communicates with the outside world.
2. The microprocessor accepts data in form of binary digits as an input from the devices such as keyboard, and A/D converters and it sends data to the output devices such as LEDs or printers.
3. There are two different methods to identify these input/output devices. One uses 8-bit address and the other uses 16-bit address.

I/O with 8-bit addresses (peripheral-mapped I/O) :

1. In this type of input/output devices, 8 address lines are used to identify an input or output device, this is known as peripheral mapped I/O (also known as I/O mapped I/O).
2. This is an 8-bit numbering system for I/O used in conjunction with input and output instruction. This is also known as I/O space which is 16-bit numbering system.
3. The 8 address lines can have 256 addresses. Thus, microprocessor can identify 256 input and 256 output devices with an address ranging from 00H to FFH.
4. The microprocessor uses I/O read control signals for the input devices and I/O write control signal for output devices.

I/O with 16-bit addressing (memory-mapped I/O) :

1. In this type of I/O, 16 addressing lines are used by the microprocessor to identify an I/O device.

Scanned by TapScanner

2. An I/O device is connected as if it is a memory register. This is known as memory-mapped I/O. The microprocessor uses same control signal and instructions as those of memory.
3. In some microprocessor, all I/Os have 16-bit addresses : I/O and memory share same memory map.
4. The entire range of memory-mapped I/O address is from 0000H to FFFFH.

PART-2

The 8085 MPU-Architecture, Pins and Signals, Timing Diagrams.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.8. **Define the features of 8085 μP.**

Answer

1. It is an 8-bit microprocessor.
2. It operates on clock cycle with 50 % duty cycle.
3. It can operate with 3 MHz clock frequency.
4. It provides 8-bit I/O addresses to access (2^8) 256 I/O ports.
5. It provides five hardware interrupts : TRAP, RST 7.5, RST 6.5, RST 5.5, and INTR.
6. It provides control signal (IO/$\overline{M}$, $\overline{RD}$, $\overline{WR}$) to control the bus cycle and hence external bus controller is not required.
7. It has 8-bit accumulator, flag register, instruction register, six 8-bit general purpose register and two 16-bit registers (SP and PC).

Que 1.9. **Draw the pin diagram of 8085 and specify the function and direction of information flow of address bus, data bus and control bus.**

AKTU 2018-19, Marks 07

Answer

The signals of 8085 can be classified into seven groups according to their functions :

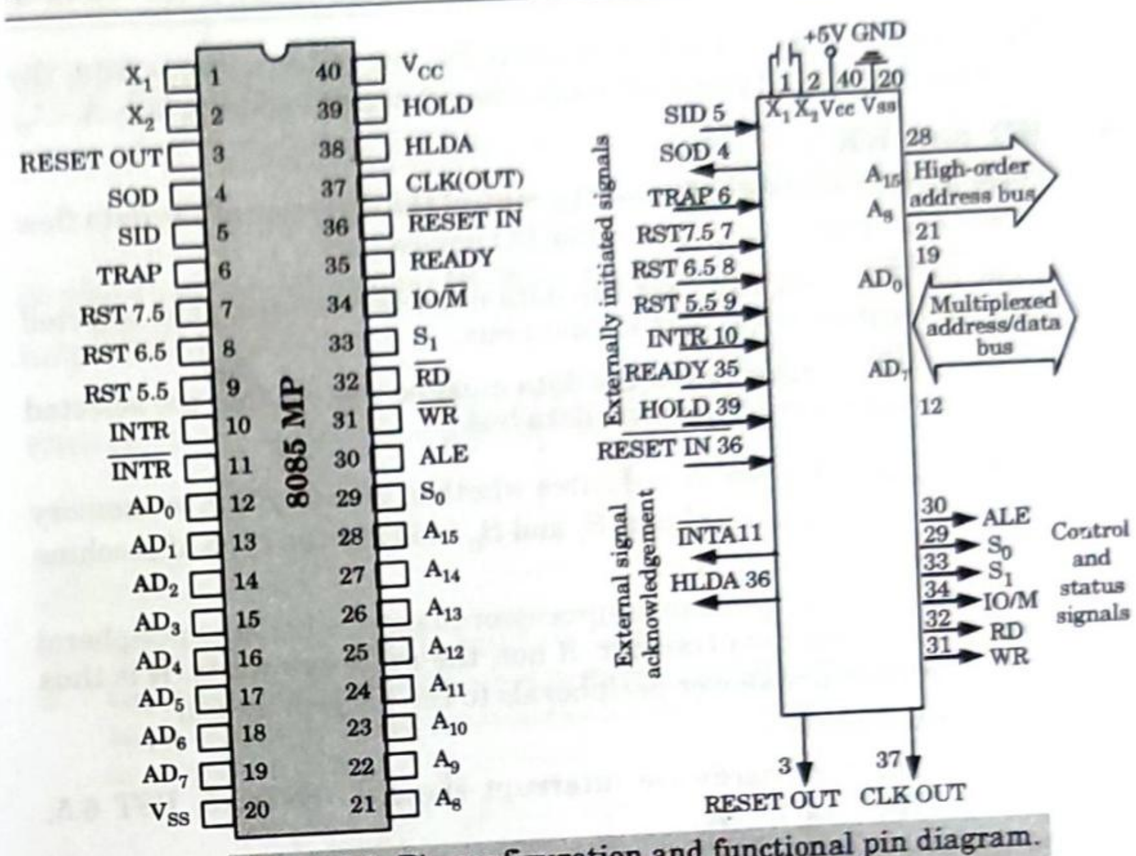


Fig. 1.9.1. Pin configuration and functional pin diagram.

1. **Power supply and frequency signals :**

a. **V_{CC} :** It requires a single + 5 V power supply.

b. **V_{SS} :** Ground reference.

c. **X_1 and X_2 :** A tuned circuit like LC, RC or crystal is connected at these two pins. The internal clock generator divides oscillator frequency by 2, therefore, to operate a system at 3 MHz, the crystal of tuned circuit must have a frequency of 6 MHz.

d. **CLK OUT :** This signal is used as a system clock for other devices. Its frequency is half the oscillator frequency.

2. **Data bus and address bus :**

a. **AD_0 to AD_7 :** The 8-bit data bus ($D_0 - D_7$) is multiplexed with the lower half ($A_0 - A_7$) of the 16-bit address bus. During first part of the machine cycle (T_1), lower 8-bits of memory address or I/O address appear on the bus. During remaining part of the machine cycle (T_2 and T_3) these lines are used as a bi-directional data bus.

b. **A_8 to A_{15} :** The upper half of the 16-bit address appears on the address lines A_8 to A_{15}. These lines are exclusively used for the most significant 8 bits of the 16-bit address lines.

3. **Control and status signals :**

a. **ALE (Address latch enable) :**

i. This is a positive going pulse generated every time when 8085 begins an operation (machine cycle). It indicates that the bits on $AD_7 - AD_0$ are address bits.

Scanned by TapScanner

ii. This signal is used primarily to latch the low order address from the multiplexed bus and generate a separate set of eight address line, A_7–A_0.

b. **$\overline{RD}$ and $\overline{WR}$:**

i. These signals are basically used to control the direction of the data flow between processor and memory or I/O device/port.

ii. A low on $\overline{RD}$ indicates that the data must be read from the selected memory location or I/O port via data bus.

iii. A low on $\overline{WR}$ indicates that the data must be written into the selected memory location or I/O port via data bus.

c. **IO/$\overline{M}$, S_0 and S_1 :** IO/$\overline{M}$ indicates whether I/O operation or memory operation is being carried out. S_1 and S_0 indicate the type of machine cycle in progress.

d. **READY :** It is used by the microprocessor to sense whether a peripheral is ready or not for data transfer. If not, the processor waits. It is thus used to synchronize slower peripherals to the microprocessor.

4. **Interrupt signals :**

a. The 8085 has five hardware interrupt signals : RST 5.5, RST 6.5, RST 7.5, TRAP and INTR.

b. The microprocessor recognizes interrupt requests on these lines at the end of the current instruction execution.

c. The $\overline{INTA}$ (Interrupt acknowledge) signal is used to indicate that the processor has acknowledged an INTR interrupt.

5. **Serial I/O signals :**

a. **SID (Serial I/P data) :** This input signal is used to accept serial data bit by bit from the external device.

b. **SOD (Serial O/P data) :** This is an output signal which enables the transmission of serial data bit by bit to the external device.

6. **DMA Signal :**

a. **HOLD :** This signal indicates that another peripheral device is requesting for the use of address bus, data bus and control bus.

b. **HLDA :** This active high signal is used to acknowledge HOLD request.

7. **Reset signals :**

a. **$\overline{RESET\ IN}$:** A low on this pin :

i. Sets the program counter to zero (0000H).

ii. Resets the interrupt enable and HLDA flip-flops.

iii. Tri-states the data bus, address bus and control bus.

iv. Affects the contents of processor's internal registers randomly.

b. **RESET OUT :** This active high signal indicates that processor is being reset. This signal is synchronized to the processor clock and it can be used to reset other devices connected in the system.

Que 1.10. **With the neat pin and block diagram, describe the internal architecture of 8085. State the function of each block.**

AKTU 2015-16, Marks 10

OR

Explain the architecture of 8085 microprocessor in brief with the help of neat diagram.

AKTU 2016-17, Marks 10

OR

With the help of neat block diagram, explain the architecture of 8085 microprocessor.

AKTU 2018-19, Marks 07

Answer

A. **Pin diagram of 8085 :** Refer Q. 1.9, Page 1–9C, Unit-1.

B. **Internal architecture or functional block diagram :** The 8085 is an 8-bit, general-purpose microprocessor that is ideally suited to many applications. Fig. 1.10.1 shows the architecture of 8085.

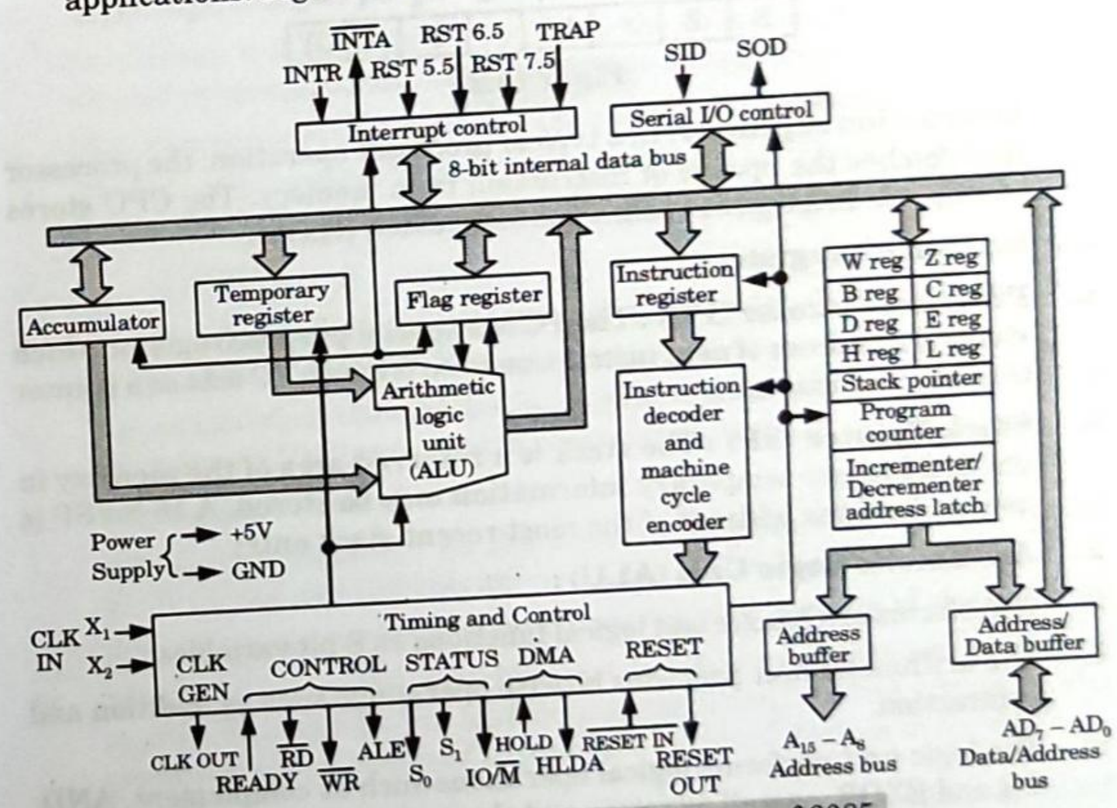


Fig. 1.10.1. Internal architecture of 8085.

C. **8085 consists of following functional blocks :**

1. **Registers :** The 8085 registers are classified as :

i. **General purpose register :** B, C, D, E, H and L are 8-bit general purpose registers, which can be used as a separate 8-bit registers or as 16-bit register pairs, BC, DE and HL.

Scanned by TapScanner

ii. **Temporary registers :**

a. **Temporary data register :** It provides input to the ALU. The programmer cannot access temporary data register. It is internally used for execution of most of the arithmetic and logical instructions.

b. **W and Z registers :** These registers are used to hold 8-bit data during execution of some instructions. These registers are not available for programmer, since 8085 uses them internally.

iii. **Special purpose registers :**

a. **Register A (Accumulator) :** It is an 8-bit register. It is used extensively in arithmetic, logic, load and store operations. Most of the times the result of arithmetic and logical operations is stored in the register A. Hence, it is also identified as accumulator.

b. **Flag register :** It is an 8-bit register, in which five of the bits carry significant information in the form of flags : S (Sign flag), Z (Zero flag), AC (Auxiliary carry flag), P (Parity flag) and CY (Carry flag) as shown in Fig. 1.10.2.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z	×	AC	×	P	×	CY

Fig. 1.10.2.

c. **Instruction registers :** In a typical processor operation, the processor first fetches the opcode of instruction from memory. The CPU stores this opcode in a register called the instruction register.

iv. **Sixteen bit registers :**

a. **Program Counter (PC) :** The PC is a special purpose register which stores the address of next instruction to be fetched. PC acts as a pointer to next instruction.

b. **Stack Pointer (SP) :** The stack is a reserved area of the memory in the RAM where temporary information may be stored. A 16-bit SP is used to hold the address of the most recent stack entry.

2. **Arithmetic Logic Unit (ALU) :**

i. It performs arithmetic and logical functions on 8-bit variables.

ii. The arithmetic unit performs bitwise operations such as addition and subtraction.

iii. The logic unit performs logical operations such as complement, AND, OR and EXOR, as well as rotate and clear.

3. **Instruction decoder :**

i. The opcode of instruction which is fetched by the processor is stored in the instruction register.

ii. Then it is sent to the instruction decoder.

iii. The instruction decoder, decodes it and accordingly gives the timing and control signals which controls the register, the data buffers, ALU and external peripheral signals $\overline{INTA}$.

4. **Address buffer :**

i. This is an 8-bit unidirectional buffer.

ii. It is used to drive external high order address bus ($A_{15} - A_8$).

iii. It is also used to tri-state the high order address bus under certain conditions such as reset, hold, halt and when address lines are not in use.

5. **Address/Data buffer :**

i. This is an 8-bit bi-directional buffer. It is used to drive multiplexed address/data bus, *i.e.*, low-order address bus ($A_7 - A_0$) and data bus ($D_7 - D_0$).

ii. The address and data buffers are used to drive external address and data buses respectively.

6. **Interrupt control :**

i. Sometimes it is necessary to execute the collection of special routine whenever special condition exists within a program.

ii. The occurrence of this special condition is referred as interrupt.

iii. The interrupt control block has five interrupt inputs RST 5.5, RST 6.5, RST 7.5, TRAP and INTR and one acknowledge signal $\overline{INTA}$.

7. **Serial I/O control :**

i. The 8085 serial input/output control provides two lines, SOD and SID for serial communication.

ii. The serial output data (SOD) line is used to send data serially and serial input data (SID) line is used to receive data serially.

8. **Timing and control circuitry :**

i. The control circuitry in the 8085 is responsible for control and synchronization of all operations.

ii. The operations in 8085 are synchronized with the help of clock signal.

iii. Along with the control of fetching and decoding operations and generating appropriate signals for instruction execution, control circuitry also generates signal required to interface external devices to the 8085.

Que 1.11. **What is flag register ? Explain each flag of 8085 microprocessor.**

Answer

A. **Flag register :** It is 8-bit register in which five bits carry significant information in form of flags as shown in Fig. 1.11.1.

Scanned by TapScanner

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z	×	AC	×	P	×	CY

Fig. 1.11.1. Flag register.

B. Types :

1. S (Sign flag) :

i. After the arithmetic or logical operations, if bit D_7 of the result is 1, the sign flag is set.

ii. In a given byte if D_7 is 1, the number will be viewed as negative number.

iii. If D_7 is 0, the number will be considered as positive.

2. Z (Zero flag) :

i. The zero flag sets, if the result of operation in ALU is zero and flag reset if the result is non-zero.

ii. The zero flag is also set if a certain register content becomes zero following an increment and decrement operation of that register.

3. AC (Auxiliary carry flag) :

i. This flag is set if there is overflow out of bit 3, *i.e.*, carry from lower nibble to higher nibble (D_3 to D_4 bit).

ii. This flag register is used for BCD operations and not available for programmer.

4. P (Parity flag) :

i. Parity is defined by the number of 1's present in the accumulator.

ii. After arithmetic or logic operation if the result has an even number of 1's, *i.e.*, even parity, the flag is set.

iii. If the parity is odd, flag is reset.

5. CY (Carry flag) :

i. The flag is set if there is an overflow out of bit 7.

ii. The carry flag also serves as a borrow flag for subtraction.

Que 1.12. Explain the flags of 8085 microprocessor. Give the flag status when following additions are performed.

i. 51H + A9H ii. 2EH + 5AH iii. 76H + A4H

AKTU 2016-17, Marks 10

Answer

A. Flags : Refer Q. 1.11, Page 1– 14C, Unit-1.

B. Numerical :

i.
```
   51 H = 01010001
 + A9 H = 10101001
 ----------------
 SUM = 11111010
```
Flag status : S = 1, Z = 0, CY = 0, P = 1, AC = 0

ii.
```
   2 E H = 00101110
 + 5A H = 01011010
 ----------------
 SUM = 10001000
```
Flag status : S = 1, Z = 0, CY = 0, P = 1, AC = 1

iii.
```
   76 H = 01110110
 + A 4 H = 10100100
 ----------------
 SUM = 000 11010
```
Flat status : S = 0, Z = 0, CY = 1, P = 0, AC = 0

Que 1.13. Name the different types of buses used in Microprocessor family. Explain the need to demultiplex the bus AD_7-AD_0 with neat diagram.

AKTU 2018-19, Marks 07

Answer

i. Types of Buses : Refer Q. 1.4, Page 1–4C, Unit-1.

ii. Need to demultiplex :

1. The signal lines $AD_0 - AD_7$ are bidirectional. They are used as the low-order address bus as well as the data bus.
2. In executing an instruction, during the earlier part of the cycle, these lines are used as the low order address bus and as the data bus during the later cycle.
3. However, the lower order address is lost after the first clock period. This address needs to be latched and used for identifying the memory address.
4. To achieve this, the data bus $AD_0 - AD_7$ is needed to be demultiplexed. Fig. 1.13.1 shows the use of latch and ALE signal to demultiplex the bus.

Scanned by TapScanner

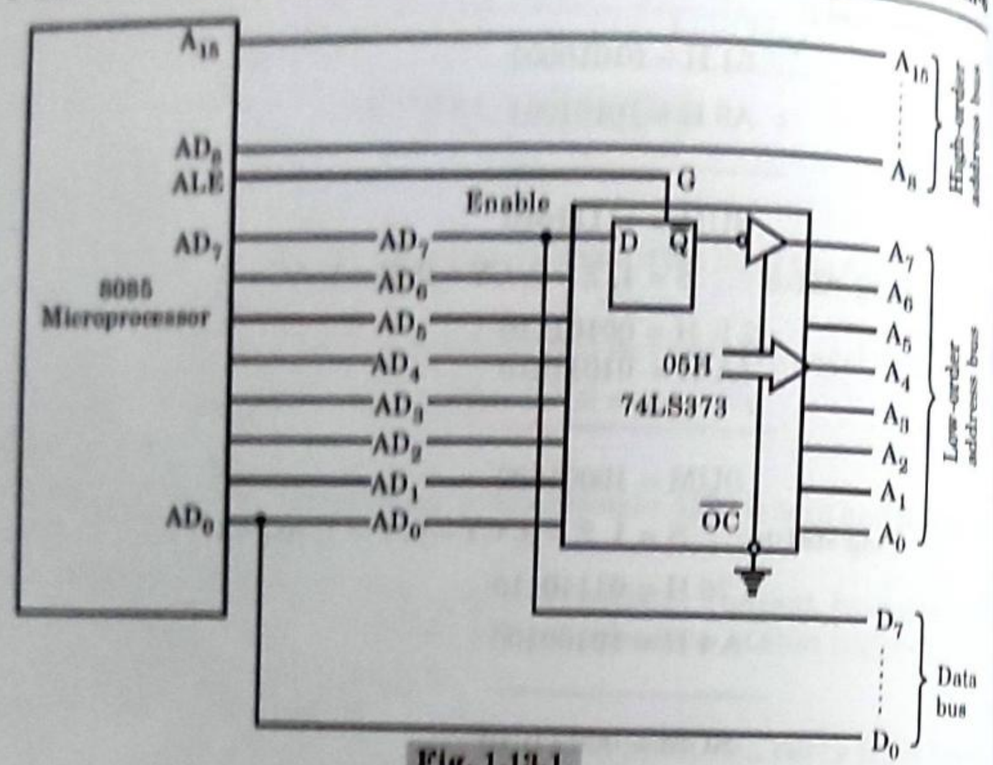


Fig. 1.13.1.

Que 1.14. **Explain the function of externally initiated signals of 8085.**

AKTU 2016-17, Marks 05

Answer

External devices can initiate the operation by activating Reset, Interrupt, READY or HOLD pins of the microprocessor. These operations are as follows :

1. After activation of Reset pin, suspend all internal operations and clear program counter so that it can fetch the next instruction from the predefined memory address.
2. After activation of any interrupt pin, execute specific sequence of instructions called interrupt service routine and after completion resume its interrupted operation.
3. After activation of READY pin, extend the machine cycle until the activation of READY pin to interface with the slower devices.
4. After activation of HOLD pin, relinquish the control of buses and allow the external controller to use them.

Que 1.15. **Discuss the following :**

i. Instruction cycle and machine cycle.
ii. T-states with typical timing diagram.

Answer

i. Instruction cycle and machine cycle :

1. Instruction cycle consists of opcode fetch followed by an execute cycle.
2. The execute cycle itself consists of zero or more fetch cycles which may be required to fetch the operand.
3. All these operations are performed in different time slots known as machine cycles.
4. An instruction cycle may consist of more than one machine cycles.
5. The first of these is always the opcode fetch cycle.
6. Some instruction, which require register to register transfer inside the microprocessor, may be executed in one machine cycle.
7. Whereas some which require data transfer between microprocessor and memory or input/output device may used more than one machine cycle.
8. Fig. 1.15.1 exhibits instruction cycle and machine cycle within it. It shows two machine cycles M_1 and M_2.
9. M_1 in which opcode is fetched and decoded consist of four state.
10. M_2 consists of three states, the data contained in the next byte of instruction is fetched and instruction executed.

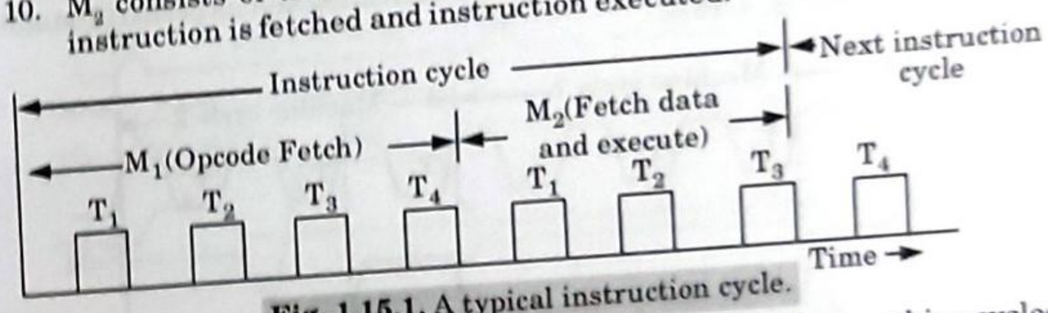


Fig. 1.15.1. A typical instruction cycle.

11. In general, one instruction cycle consists of one to five machine cycles and one machine cycle consists of three to six T-states, *i.e.*, three to six clock periods, as shown in Fig. 1.15.2.

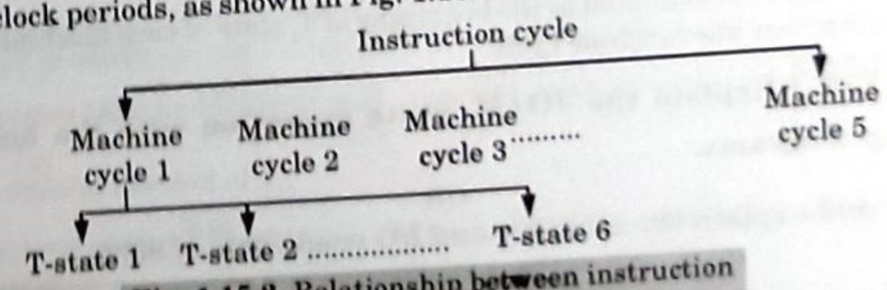


Fig. 1.15.2. Relationship between instruction cycle, machine cycle and T-state.

ii. T-state :

1. One machine cycle consist of several clock cycles each of these clock cycles are known as state.

Scanned by TapScanner

2. The state within machine cycles is generally referred as T-states (T_1, T_2 ...).
3. One machine cycle consists of 3 to 6 T-states, *i.e.*, 3 to 6 clock periods.
4. The opcode fetch cycle has 4 T-states.
5. The 8085 uses the first 3 states to fetch the code and T_4 to decode and execute the opcode.
6. In 8085 instruction set, some instructions have opcodes with 6 T-states.
7. Ideally the clock signal should be square wave with zero rise time and fall time but in practice, we do not get zero rise time and fall time as shown in Fig. 1.15.3.

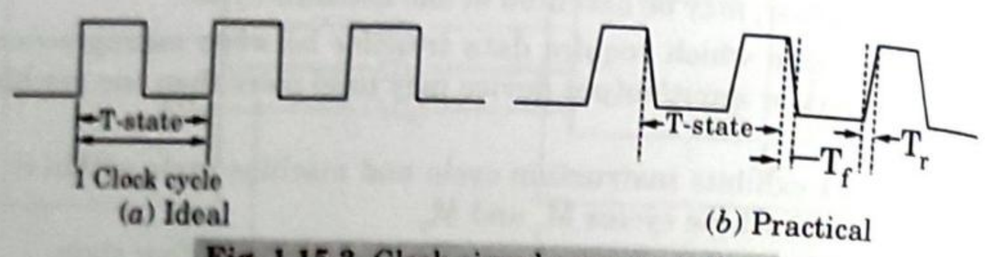


Fig. 1.15.3. Clock signal representation.

8. A typical timing diagram showing ALE activation and its period (T-state) is shown in Fig. 1.15.4.

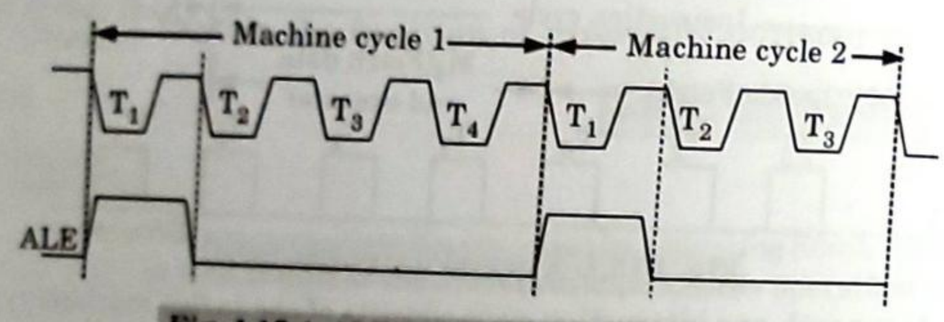


Fig. 1.15.4. ALE activation and its period.

9. ALE signal is activated in the beginning of T_1 state of each machine cycle except bus idle machine cycle.

Que 1.16. **Explain the IO/$\overline{M}$ write operation with the help of timing diagrams.**

OR

Draw and explain the memory and I/O read cycle of 8085.

AKTU 2016-17, 2017-18; Marks 10

OR

Draw and explain the timing diagram of memory read operation in 8085. Write different steps used in it. **AKTU 2015-16, Marks 10**

Answer

A. Read Cycle :

1. The high order address ($A_{15} \Leftrightarrow A_8$) and low order address ($AD_7 \Leftrightarrow AD_0$) are asserted on 1st low going transition of the clock pulse.
2. The timing diagram for IO/$\overline{M}$ read are shown in Fig. 1.16.1 and Fig. 1.16.2. $A_{15} \Leftrightarrow A_8$ remains valid in T_1, T_2 and T_3 *i.e.*, duration of the bus cycle, but $AD_7 \Leftrightarrow AD_0$ remains valid only in T_1.
3. Since it has to remain valid for the whole bus cycle, it must be saved for its use in T_2 and T_3.

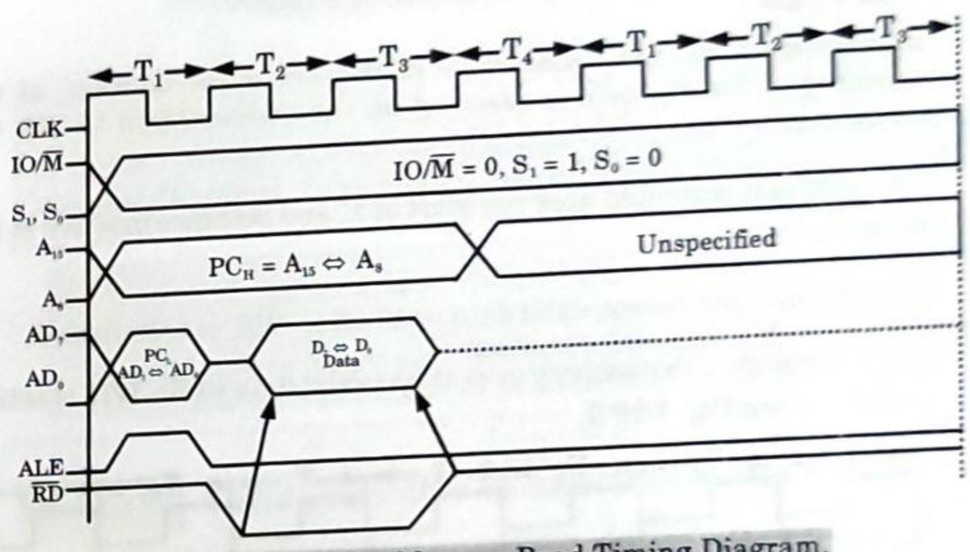


Fig. 1.16.1. Memory Read Timing Diagram.

4. ALE is asserted at the beginning of T_1 of each bus cycle.
5. ALE is active during T_1 only and is used as the clock pulse to latch the address ($AD_7 \Leftrightarrow AD_0$) during T_1. The $\overline{RD}$ is asserted near the beginning of T_2.
6. It ends at the end of T_3.
7. As soon as the $\overline{RD}$ becomes active, it forces the memory or I/O port to assert data.
8. $\overline{RD}$ becomes inactive towards the end of T_3, causing the port or memory to terminate the data.

Scanned by TapScanner

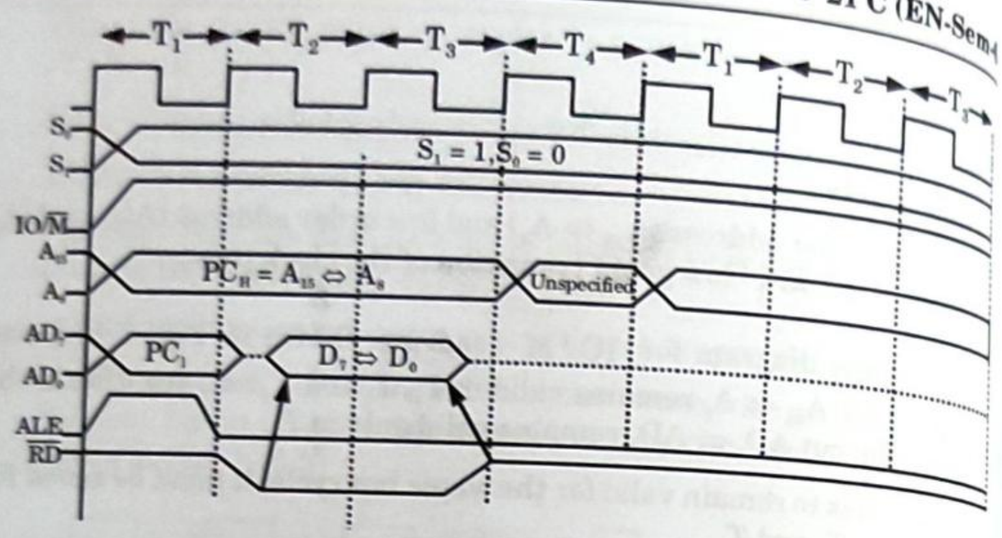


Fig. 1.16.2. I/O Read Timing Diagram.

B. Write Cycle :

1. Immediately after the termination of the low order address, at the beginning of the T_2, data is asserted on the address/data bus by the processor.
2. $\overline{WR}$ control is activated near the start of T_2 and becomes inactive at the end of T_3.
3. The processor maintains valid data until after $\overline{WR}$ is terminated.
4. This ensures that the memory or port has valid data while $\overline{WR}$ is active, it is clear from Fig. 1.16.3.

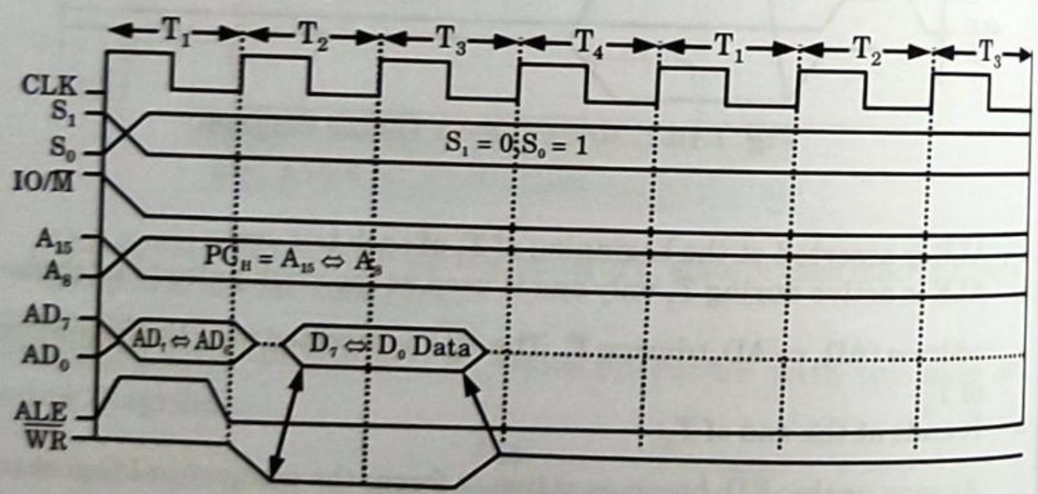


Fig. 1.16.3. Memory Write Timing Diagram.

5. For READ bus cycle, the data appears on the bus as a result of activating $\overline{RD}$ and for the $\overline{WR}$ bus cycle, valid data is appeared on the bus during the time when $\overline{WR}$ is active.

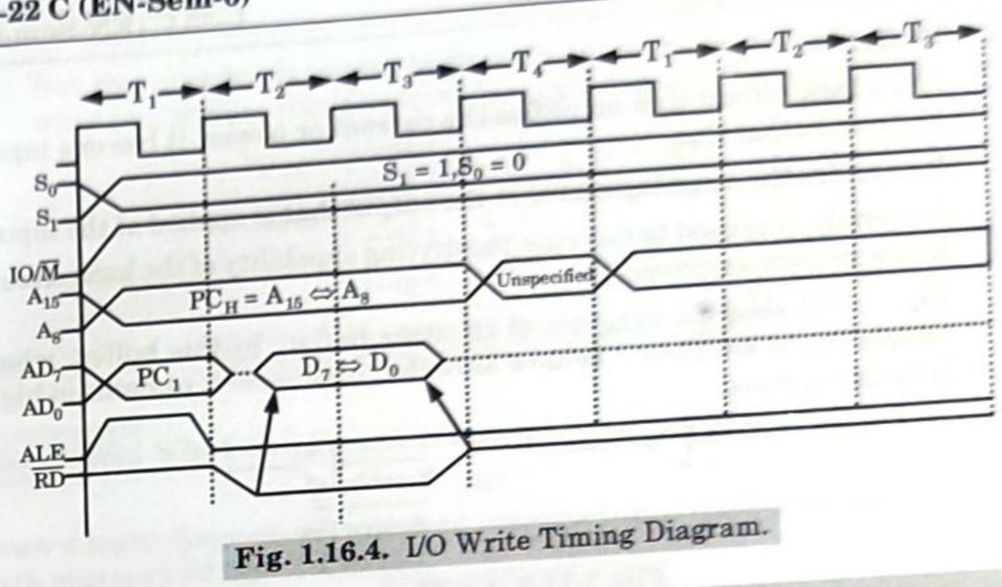


Fig. 1.16.4. I/O Write Timing Diagram.

PART-3

Logic Devices for Interfacing, Memory Interfacing, Interfacing Output Displays, Interfacing Input Devices Memory, Mapped I/O.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.17. **Write short note on logic devices for interfacing.**

Answer

Logic devices for interfacing :

i. Tri-state devices :

1. These are the devices which have three logic states, *i.e.*, logic 0, logic 1 and high input impedance.
2. These devices have enable line, when activated they work like ordinary logic devices, but when the enable line is disabled then the logic devices switch into high impedance state and behave as they are disconnected from the system.

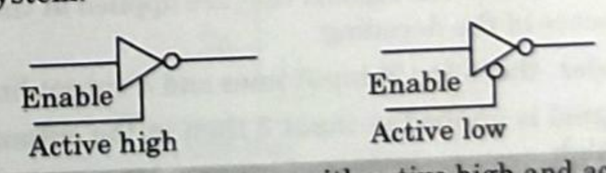


Fig. 1.17.1. Tri-state inverter with active high and active low.

Scanned by TapScanner

ii. Buffer :

1. It is a logic circuit that amplifies the current or power. It has one input and one output line.
2. It provides the same logic level at the output that is applied at the input.
3. Generally, it is used to increase the driving capability of the logic circuit hence known as driver.
4. Fig. 1.17.2 shows a diagram of tri-state buffer. In this buffer, when enable line is low then it behave as buffer; otherwise it remains in high impedance state.

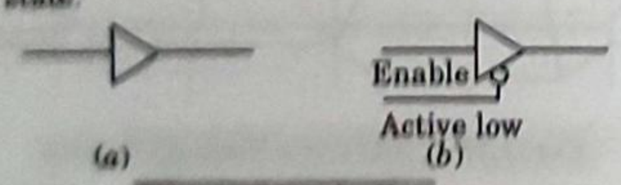


Fig. 1.17.2. A buffer.

iii. Decoder :

1. Decoder is a logic circuit that identifies combination of signals that are applied at the input.
2. In decoder if there are n input lines then at the output there are 2^n output lines.
3. Suppose there is 3 × 8 decoder, *i.e.*, there are 3 input lines then there are 2^3, *i.e.*, 8 possible combination which are identified by 0 to 7 output lines.
4. If 111 is applied at the input then the output at 7 will be 1 and all others will remain at logic zero.

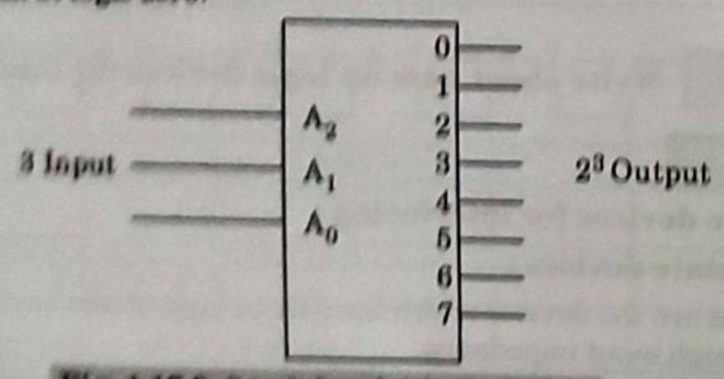


Fig. 1.17.3. 3 to 8 decoder logic symbol.

iv. Encoder :

1. Encoder is a logic circuit which is used to provide the appropriate code at the output for each of the signals that are applied at the input. It is the reverse process of the decoding.
2. In the encoder, there are 2^n input lines and n output lines.
3. Suppose signal is applied at input 3 then at the output both LSB and MSB goes high.
4. Similarly, if signal is applied at input 2 then MSB goes high and LSB goes low.

5. But this encoder is unable to provide appropriate code at the output when two or more than two lines are activated simultaneously at a time.

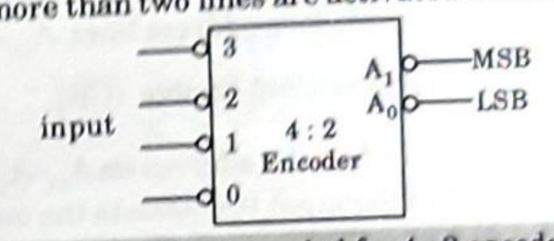


Fig. 1.17.4. Logic symbol for 4 : 2 encoder.

Que 1.18. **What do you understand by memory interfacing ?**

OR

Draw a logic diagram of complete interfacing of 8085 microprocessor with memory of size 4K bytes.

Answer

A. Memory interfacing :

1. The main function of memory interfacing is that microprocessor should be able to read from and write into the given register or a memory chip.
2. To perform these operations microprocessor should be able to do following functions :

i. Select the chip.

ii. Identify the register.

iii. Enable the appropriate buffer.

B. Interfacing circuit :

1. Fig. 1.18.1 shows an interfacing circuit using a 3-to-8 decoder to interface the 2732 EPROM memory chip.

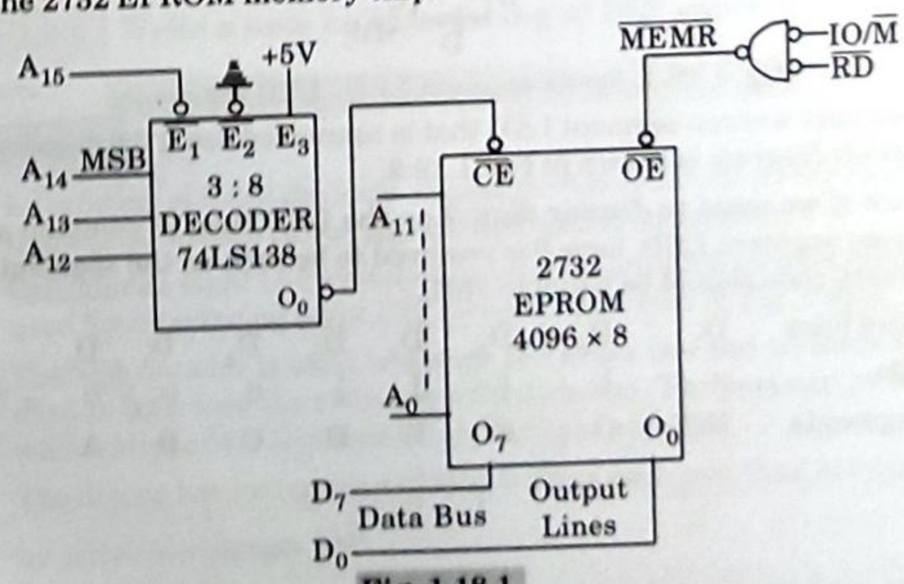


Fig. 1.18.1.

Scanned by TapScanner

2. The 8085 address lines A_{11}–A_0 are connected to pins A_{11}–A_0 of the memory chip to address 4096 register.
3. The decoder is used to decode four address lines A_{15}–A_{12}. The output O_0 of the decoder is connected to Chip Enable ($\overline{CE}$).
4. The $\overline{CE}$ is asserted only when the address on A_{15}–A_{12} is 0000, A_{15} (low) enables the decoder and the input 000 asserts the ouput O_0.

Que 1.19. Explain the interfacing of seven segment LED display as an output device.

Answer

1. A seven segment display is an output device, consists of one segment of decimal point and seven light emitting diode segment.
2. The physical arrangement of these LEDs is shown in Fig. 1.19.1.
3. The necessary segments are glowed by sending an appropriate signal for current flow through diodes to display a number.
4. Seven segment LED are categorized in two groups : common cathode and common anode.
5. The seven segment A to G are connected to data lines D_0 to D_6 respectively. DP is connected to data line D_7 if decimal point is used.
6. The binary code required to display a digit is determined by the type of seven segment LED, connection of data lines and logic required to light the segment.

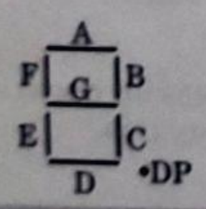


Fig. 1.19.1. Seven-segment LED : LED segments.

7. Consider a seven segment LED that is connected using 3:8 decoder. The circuit diagram is shown in Fig. 1.19.2.
8. Now if we want to display digit 7 on the LED then in common anode seven segment LED, logic 0 is required to be turn on the segment. The binary code should be equal to 78H.

Data lines	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
Bits	×	1	1	1	1	0	0	0	= 78H
Segments	NC	G	F	E	D	C	B	A	

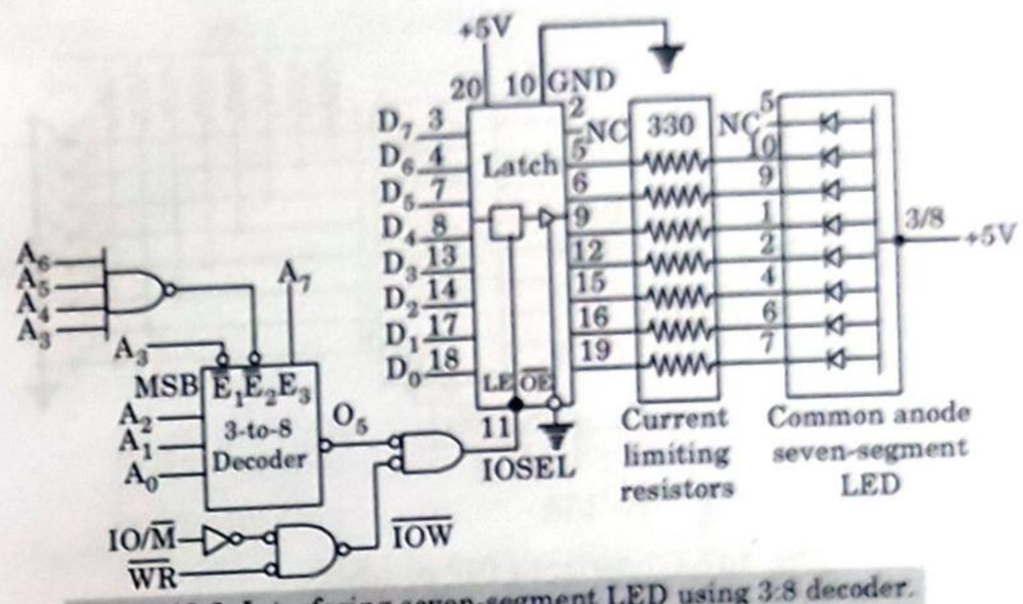


Fig. 1.19.2. Interfacing seven-segment LED using 3:8 decoder.

8. Now if we have to design an output port with port address F5H, then address line should have following logic.

A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
1	1	1	1	0	1	0	1	= F5H

9. This can be achieved using A_2, A_1 and A_0 as input lines to the decoder. A_7 can be connected to active low enable $\overline{E}_1$ and the remaining address lines can be connected to $\overline{E}_2$ through 4-input NAND gate. Fig. 1.19.2 shows the output port with address F5H.

Que 1.20. Write a note on interfacing of DIP switches as an input device.

Answer

1. Interfacing of input device is similar to that of output device except that there is difference in bus signals and circuit components.
2. Consider an eight DIP switches interfaced using 3:8 decoder. The circuit used for interfacing eight DIP switches is shown in Fig. 1.20.1.
3. The 3 : 8 decoder is used to decode low order bus and tri-state buffer is used to interface the switches with data bus. The port can be accessed with address 84H however it has multiple address.
4. The device has two groups of four buffers each and they are controlled by active low signals $\overline{OE}$.
5. When $\overline{OE}$ is low the input data shown up on the output lines and when $\overline{OE}$ signal is high, the output lines assumes high impedance state.

Scanned by TapScanner

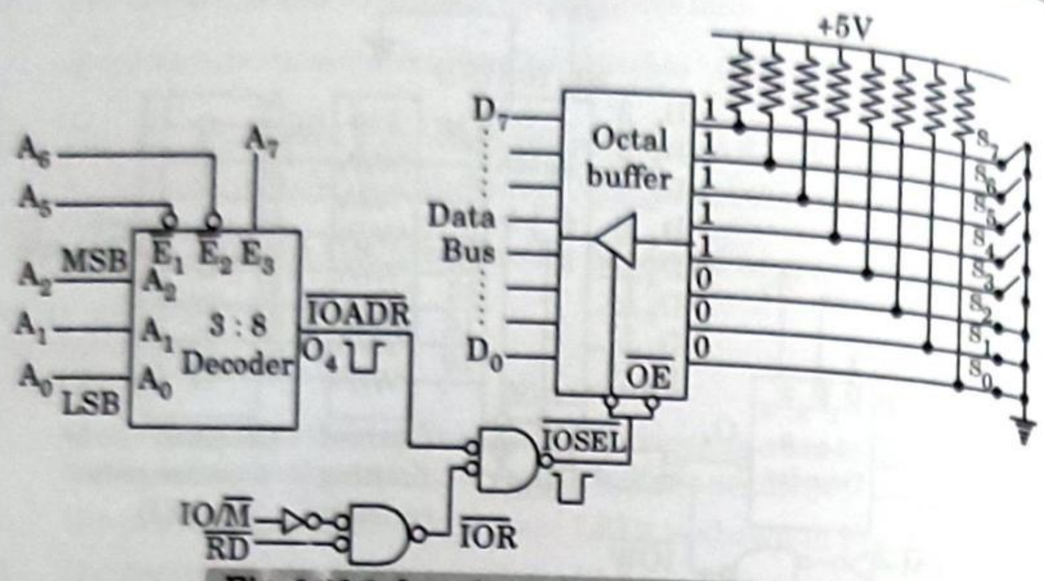


Fig. 1.20.1. Interfacing DIP switches.

6. The low order address bus, except the lines A_4 and A_3, is connected to the decoder. The address lines A_3 and A_4 are left in don't care state.
7. The output line O_4 of the decoder goes low when the address bus has following address

$$\underbrace{\overset{A_7}{1}\ \overset{A_6}{0}\ \overset{A_5}{0}}_{\text{Enable lines}}\ \underbrace{\overset{A_4}{0}\ \overset{A_3}{0}}_{\text{Don't care}}\ \underbrace{\overset{A_2}{1}\ \overset{A_0}{0}\ \overset{A_1}{0}}_{\text{input}} = 84H$$

8. The control signal $\overline{I/OR}$ is generated by ANDing the IO/$\overline{M}$ and $\overline{RD}$ in a negative NAND gate, and the I/O select pulse is generated by ANDing the output of decoder and control signal $\overline{IOR}$.
9. When the address is 84H and control signal $\overline{IOR}$ is asserted, the I/O select the pulse enable tri-state buffer and logic levels of the switches are placed on data bus. The 8085 then begins to read switch position and places the reading in accumulator.

Que 1.21. Explain the memory-mapped I/O with an example.

Answer

1. In memory-mapped I/O, the input and output devices are assigned and identified by 16-bit addresses.
2. To transfer data between the MPU and I/O devices, memory related instructions (such as LDA, STA, etc.) and memory control signals ($\overline{MEMR}$ and $\overline{MEMW}$) are used.

Memory address	Machine code	Mnemonics	Comments
2050	32	STA 8000H	Store the content of accumulator in memory location 8000H
2051	00		
2052	80		

3. For example, the following instruction will transfer the contents of the accumulator to the memory location 8000H.

   The STA is a 3-byte instruction; the first byte is the opcode, second and third bytes specify the memory address.

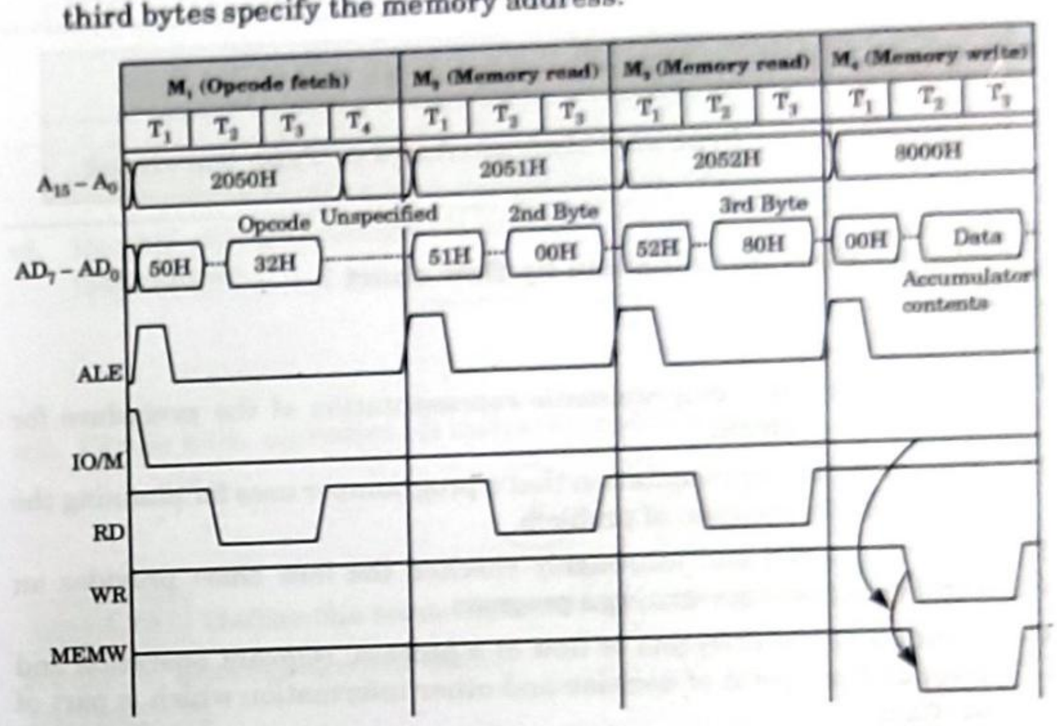


Fig. 1.21.1. Timing of execution of the instruction : STA 8000H.

4. However, the 16-bit address 8000H is entered in the reverse order; the low-order byte 00 is stored in location 2051H followed by the high order address 80H.
5. The instruction LDA (Load accumulator direct) transfers the data from a memory location to the accumulator. The instruction LDA is a 3-byte instruction.
6. To use memory-related instructions for data transfer the control signals Memory read ($\overline{MEMR}$) and Memory write ($\overline{MEMW}$) should be connected to I/O devices instead of IOR and IOW signals and the 16-bit address bus ($A_{15} - A_0$) should be decoded.
7. Device selection and data transfer in memory-mapped I/O require three steps :

   i. Decode the address bus to generate the device address pulse.

Scanned by TapScanner

ii. AND the control signal with the device address pulse to generate the device select (I/O select) pulse.

iii. Use the device select pulse to enable the I/O port.

PART-4

Basic Programming Concepts : Flow Chart Symbols, Data Transfer Operations, Arithmetic Operations, Logic Operations, Branch Operation.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.22. What do you mean by flow chart ?

Answer

1. A flow chart is a diagrammatic representation of the procedure for solving the problem.
2. It is a pictorial representation that a programmer uses for planning the procedure for solution of problem.
3. Once developed and thoroughly checked the flow chart provides an excellence guide for writing a program.
4. It indicates the direction of flow of a process, relevant operation and computation, point of decision and other information which is part of solution.
5. Step-by-step process of how to develop a flow chart :

i. Gather information of how the process flow : Using conservation, experience and product development codes.

ii. Trial process flow.

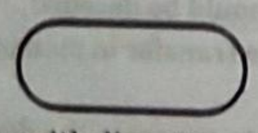
iii. Allow other more familiar personnel to check for accuracy.

iv. Make changes if necessary.

v. Compare final actual flow with best possible flow.

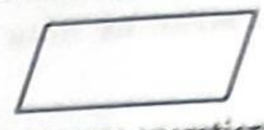
6. Some graphical symbols and their use are given as :

i. **Oval :** It indicates start or stop operation.

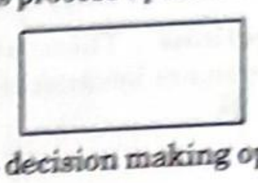


ii. **Arrow :** It indicates flow with direction.

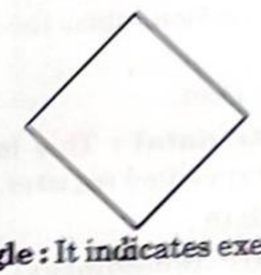
iii. **Parallelogram :** It indicates input/output operation.



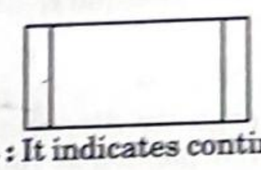
iv. **Rectangle :** It indicates process operation.



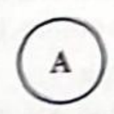
v. **Diamond :** It indicates decision making operation.



vi. **Double sided rectangle :** It indicates execution of pre-defined process (subroutine).



vii. **Circle with alphabet :** It indicates continuation.



Que 1.23. Define the term instruction and explain the instruction format of microprocessor 8085.

Answer

A. Instruction :

1. Instruction is a command given by the user to the microprocessor and needs to be get executed by the processor, the entire group of instruction are called instruction set.
2. Thus, instruction format of 8085 has two fundamental fields :

i. Opcode (Information of operation).

ii. Operand (Address of location).

B. Types of instruction :

1. **1-byte instructions :** In single byte instruction there is only one field, Example : INRC & ADDC.
2. **2-byte instructions :** The 2-byte instructions have 2-fields, namely, the opcode field and the data or address field. Example : SBI 02 H.

Scanned by TapScanner

3. **3-byte instructions :** The 3-byte instructions also have only 2-fields namely opcode and data/address field. Example : LXI D, 2060 H.

Que 1.24. **What do you mean by data transfer instructions ? Explain.**

Answer

A. **Data transfer instructions :** These are the instructions used to copy the data from one memory location or register to another memory location or register.

B. **Instructions :**

1. **MOV (Move) :** This instruction copies the content of source register to the destination register.

   Format : MOV Source, Dest.

2. **MVI (Move immediate data) :** This instruction is used to move immediate data into the specified register.

   Format : MVI R, 8-bit data

3. **LXI (Load register pair immediate) :** This instruction load 16-bit data in the register pair B-C, D-E and H-L with the data byte available immediately.

   Format : LXI R_p, 16-bit address

4. **LHLD (Load H and L register direct) :** This instruction copies the contents of memory location pointed by 16-bit address in register L and copies the contents of next memory location in register H.

   Format : LHLD 16-bit address

5. **LDA (Load accumulator direct) :** This instruction copies the data of given memory location in accumulator.

   Format : LDA 16-bit address

6. **LDAX (Load accumulator indirect) :** This instruction copies the contents of memory location whose address is specified by register pair into the accumulator.

   Format : LDAX R_p

7. **SHLD (Store H and L register direct) :** This instruction stores the contents of L register in the memory location given within the instruction and content of H register at address next to it.

   Format : SHLD 16-bit address

8. **STA (Store accumulator direct) :** The contents of accumulator are copied into memory location specified by operand.

   Format : STA 16-bit address

9. **STAX (Store accumulator indirect) :** The contents of accumulator are copied into memory location specified by register pair.

   Format : STAX R_p

Que 1.25. **What do you understand by arithmetic instruction ?**

Answer

A. **Arithmetic instruction :** These are the instructions used for arithmetic operation like addition, subtraction etc.

B. **Instructions :**

1. **ADD (Add register data to accumulator) :** This instruction add the contents of register or memory with the content of accumulator and result is stored in accumulator.

   Format : ADD R/M

2. **ACI (Add immediate data to accumulator with carry) :** This instruction add the 8-bit data and carry flag with the accumulator contents and result is stored in accumulator.

   Format : ACI 8-bit data

3. **ADC (Add register data to accumulator with carry) :** The contents of register or memory and carry flag are added to contents of accumulator and result is placed in the accumulator.

   Format : ADC R/M

4. **ADI (Add immediate data to accumulator) :** The 8-bit data are added to contents of accumulator and result is stored in accumulator.

   Format : ADI 8-bit data

5. **DAD (Add register pair to H and L register) :** This instruction is used to add the 16-bit contents of specified register with contents of register pair HL and sum is saved in HL register.

   Format : DAD R_p

6. **SUB (Subtract register or memory from accumulator) :** The contents of register or memory location specified by operand are subtracted from the contents of accumulator.

   Format : SUB R/M

7. **SUI (Subtract immediate data from accumulator) :** The 8-bit data are subtracted from contents of accumulator and result is stored in accumulator.

   Format : SUI 8-bit data

8. **SBB (Subtract source and borrow from accumulator) :** The contents of register or memory and the borrow flag are subtracted from contents of accumulator and result are placed in accumulator.

   Format : SBB R/M

9. **SBI (Subtract immediate with borrow) :** The 8-bit data and borrow are subtracted from the contents of accumulator.

   Format : SBI 8-bit data

Scanned by TapScanner

10. **DAA (Decimal adjust accumulator) :** The contents of accumulator are change from the binary value to two 4-bits binary coded decimal digit. This is the only instruction that uses auxiliary flag to perform binary to BCD conversion.

Format : DAA

11. **INR (Increment contents of register / memory by 1) :** This instruction increment the contents of register/memory by 1 and results are stored in same place.

Format : INR R/M

12. **INX (Increment register pair by 1) :** This instruction increments the contents of specified register pair by 1.

Format : INX R_P

13. **DCR (Decrement source by 1) :** The contents of designated register memory are decremented by 1 and store the result in same place.

Format : DCR R/M

14. **DCX (Decrement register pair by 1) :** This instruction decrements the contents of specified register pair by 1.

Format : DCX R_P

Que 1.26. **Discuss the logical instructions in 8085.**

Answer

A. **Logical instructions :** These are the instructions used to perform the logical operation like AND, OR, XOR etc.

B. **Instructions :**

1. **ANA (Logical AND with accumulator) :** The contents of accumulator are logically ANDed with the contents of register or memory location and result is stored in the accumulator.

Format : ANA R/M

2. **ANI (AND immediate with accumulator) :** This instruction ANDed the contents of accumulator with specified 8-bit data and result is placed in accumulator.

Format : ANI 8-bit data

3. **XRA (Exclusive OR with accumulator) :** The contents of specified register/memory location are exclusive ORed with contents of the accumulator and result are placed in accumulator.

Format : XRA R/M

4. **XRI (Exclusive OR immediate with accumulator) :** The 8-bit data are exclusive ORed with contents of accumulator and result is placed in the accumulator.

Format : XRI 8-bit data

5. **ORA (Logically OR with accumulator) :** This instruction logically ORed the contents of accumulator with contents of register on memory location.

Format : ORA R/M

6. **ORI (Logically OR immediate) :** The contents of accumulator are logically ORed with the 8-bit data in operand and the result is stored in accumulator.

Format : ORI 8-bit data

7. **CPI data (Logically compare immediate) :** Compare data with the contents of A of less than equal to or greater.

8. **CMP M (Logically compare) :** Compare the contents of memory with the contents of A for less than equal to or greater than.

9. **RAR :** This instruction rotates the content of accumulator right by one position. Bit B_0 is placed in CY and CY is placed in B_7.

Que 1.27. **What do you understand by branch operation ? List all the branching instruction and give their details.**

Answer

A. **Branching instruction :** This group of instruction alters the sequence of program execution either conditionally or unconditionally.

B. **Instructions :**

1. **JMP (Jump unconditionally) :** The jump instructions specify the memory location explicitly. The program sequence is transferred to memory location specified by the 16-bit address. The unconditionally jump instruction enables the programmer to set up continuous loop.

Format : JMP 16-bit address

Instruction	Description	Condition for JUMP
JC	Jump if carry	CY = 1
JNC	Jump if no carry	CY = 0
JP	Jump if positive	S = 0
JM	Jump if minus	S = 1
JPE	Jump if parity even	P = 1
JPO	Jump if parity odd	P = 0
JZ	Jump if zero	Z = 1
JNZ	Jump if no zero	Z = 0

2. **CALL (Unconditional subroutine call) :** The program sequence is transferred to address specified by operand. Before the transfer, the address of next instruction to CALL is pushed on the stack.

Scanned by TapScanner

Format : CALL 16-bit address

Instruction Code	Description	Flag status
CC	Call if carry	CY = 1
CNC	Call if no carry	CY = 0
CP	Call if positive	S = 0
CM	Call if minus	S = 1
CPE	Call if parity even	P = 1
CPO	Call if parity odd	P = 0
CZ	Call if zero	Z = 1
CNZ	Call if no zero	Z = 0

3. **RET (Return from subroutine unconditionally) :** This instruction pops the return address from the stack and loads program counter with this return address. Thus transfers program control to the instruction next to CALL in the main program.

Instruction code	Description	Flag status
RC	Return if carry	CY = 1
RNC	Return if no carry	CY = 0
RP	Return if positive	S = 0
RM	Return if minus	S = 1
RPE	Return if parity even	P = 1
RPO	Return if parity odd	P = 0
RZ	Return if zero	Z = 1
RNZ	Return if no zero	Z = 0

4. **PCHL (Load program counter with HL contents) :** The contents of register H and L are copied into program counter by the help of this instruction.

5. **RST (Restart) :** This instruction transfers the program control to the specific memory location. This instruction is like a fixed address CALL instruction. This fixed address is also called as vectored address.

Instruction code	Vector	Address
RST 0	0 × 8	= 0000H
RST 1	1 × 8	= 0008H
RST 2	2 × 8	= 0010H
RST 3	3 × 8	= 0018H
RST 4	4 × 8	= 0020H

RST 5	5 × 8	= 0028H
RST 6	6 × 8	= 0030H
RST 7	7 × 8	= 0038H

The processor multiplies the RST number by 8 to calculate these vector addresses. The RST instruction saves the current program counter contents on the stack like CALL instruction.

Que 1.28. **Explain machine control instructions.**

OR

Explain the following instructions of 8085 microprocessor

a. POP PSW **b. XTHL**

c. SPHL **d. PUSH PSW**

e. CMP M

AKTU 2017-18, Marks 10

Answer

A. **Machine control instruction :** These instructions control machine function such as Halt, Interrupt or do nothing.

B. **Various machine control instructions are :**

1. **DI (Disable Interrupts) :** This instruction resets the interrupt disable flip flop and all interrupt except TRAP are disabled.
2. **EI (Enable Interrupt) :** This instruction will set the interrupt enable flip flop and all the interrupts are enabled.
3. **HLT (Halt and Enter Wait State) :** This instruction Halt the processor. It can be restarted by a valid interrupt or by applying a RESET signal.
4. **NOP (No Operation) :** The instruction is fetched and decoded; however, no operation is executed.
5. **SIM (Set Interrupt Mask) :** This is a multipurpose instruction and is used to implement the 8085 interrupts. For this instruction command byte must be loaded in accumulator.
6. **RIM (Read Interrupt Mask) :** This instruction loads the status of interrupt mask, the pending interrupt and the contents of serial input data line into the accumulator.
7. **PUSH (Push Register Pair onto Stack/PSW) :** This instruction copies the contents of register pair on the stack in the following sequence :

i. The stack pointer is decremented and the contents of higher order register (B, D, H, A) are copied on that location.

ii. The stack pointer is decremented again and the contents of low order register (C, E, L) are copied to that location.

8. **XCHG :** This instruction exchanges the contents of H & L with D & E registers.
9. **POP (Pop off Stack to Register Pair/PSW) :**

i. This instruction copies the contents of memory location pointed by the stack pointer into the lower byte of specified register pair and increment the stack pointer by one.

Scanned by TapScanner

ii. It then copies the contents of memory location pointed by stack pointer into higher bytes of specified register pair and increment the stack pointer again by one.

iii. Only higher order register is to be specified in the instruction.

10. SPHL (Copy H and L register to the stack pointer) :

i. This instruction copies the contents of HL register pair into the stack pointer.

ii. The contents of H register are copied to higher order byte of stack pointer and contents of L register are copied to the lower byte of stack pointer.

11. XTHL (Exchange H and L with top stack) :

i. This instruction exchanges the contents of memory location pointed by stack pointer with contents of L-register and contents of next memory location with the contents of H register.

ii. This instruction does not modify the stack pointer.

12. IN (Input Data to Accumulator from the Port with 8 bit address) : This instruction copies the data at the port whose address is specified in the instruction into the accumulator.

13. OUT (Output Data from Accumulator to a Port with 8-bit Address) : This instruction copies the contents of the accumulator into the output port specified in the instruction.

14. CMP M : Refer Q. 1.26, Page 1–33C, Unit-1.

PART-5

Writing Assembly Language Programs, Programming Techniques : Looping, Counting and Indexing, Additional Data Transfer and 16-bit Arithmetic Instruction, Logic Operation : Rotate, Compare.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.29. Explain the various steps involved in writing the assembly language programs.

Answer

A. Program :

1. A program is a set of instructions arranged in the specific sequence to do the specific task.
2. It tells the microprocessor what it has to do.
3. The process of writing the set of instructions which tells the microprocessor what to do is called programming.

B. Steps involved in programming :

i. Specifying the problem :

1. The first step in the programming is to find out which task is to be performed. This is called specifying the problem.
2. If the programmer does not understand what is to be done, the programming process cannot begin.

ii. Designing the problem-solution : During this process, the exact step by step process that is to be followed, is developed and written down.

iii. Coding :

1. Once the program is specified and designed, it can be implemented.
2. Implementation begins with the process of coding the program.
3. Coding the program means to tell the processor the exact step by step process in its language. Each processor has a set of instructions.
4. Programmer has to choose appropriate instructions from the instruction set to build the program.

iv. Debugging :

1. Debugging is the process of testing the code to see if it does the given task.
2. If program is not working properly, debugging process helps in finding and correcting errors.

Que 1.30. Write the programming techniques used in microprocessor.

Answer

Programming techniques :

Looping : It is a programming technique which is used to instruct the microprocessor to repeat tasks.

Counting : This technique allows programmer to count how many times the instruction/set of instructions are executed.

Indexing : This programming technique allows programmer to point or refer the data stored in sequential memory locations one by one.

Que 1.31. Explain various data transfer instruction related to microprocessor register and I/O.

Answer

16-bit data transfer to register pair (LXI) :

1. **LXI R_p, 16-bit :** This instruction loads 16-bit data in the register pair. Register pair can be BC, DE and HL. It is a 3-byte instruction.
2. **LXI SP, 16-bit :** This instruction loads 16-bit data in the stack pointer register.

Data transfer from memory to the microprocessor :

1. **MOV R, M :** This instruction copies the data byte from memory location into a register. It is a 1-byte instruction. R can be A, B, C, D, E, H and L and M is specified by the content of HL register.

Scanned by TapScanner

2. **LDAX B/D :** It is a 1-byte instruction which copies the data byte from memory location into the accumulator. Memory location can be specified by the content of registers BC or DE.
3. **LDA 16-bit :** It is a 3-byte instruction which copies the data of given memory location into accumulator.

Data transfer from microprocessor to memory or directly into memory :

1. **MOV M, R :** This is a 1-byte instruction that copies data from a register R, into the memory location specified by the content of HL registers.
2. **STAX B/D :** The contents of accumulator are copied into memory location specified by the register pairs. It is a 1-byte instruction.
3. **STA 16-bit :** It is a 3-byte instruction that copies data from accumulator into the memory location specified by the 16-byte operand.
4. **MVI M, 8-bit :** It is a 2-byte instruction that copies 8-bit data into memory location specified by HL register pair from the accumulator.

Que 1.32. Explain various 16-bit arithmetic instructions.

Answer

These are the instructions which perform arithmetic task related to memory such as ADD M, INR M etc.

i. **ADD M (Add memory data to the accumulator) :** This instruction add the contents of memory location specified by the HL register pair with accumulator and the result are stored in accumulator.

ii. **SUB M (Subtract memory content from accumulator) :** This instruction subtracts the contents of memory location specified by register pair HL and stores the result into accumulator.

iii. **INR M (Increment contents of memory by 1) :** This instruction increment the contents of memory location by 1. The memory location is specified by register pair HL. After execution of this instructions all flags are affected except carry flag.

iv. **DCR M (Decrement contents of memory by 1) :** This instruction decrease the contents of memory location specified by register pair HL by 1.

v. **INX R_p :** It is a 1-byte instruction. It treats the contents of two register as one 16-bit number and increases the contents by 1. R_p can be BC, DE, HL and stack pointer.

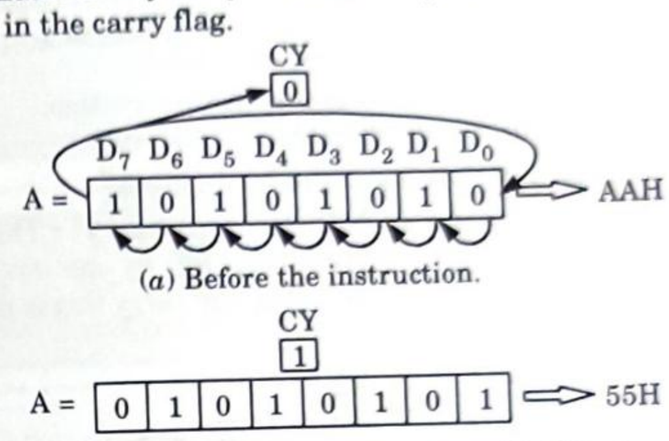
vi. **DCX R_p :** It is a 1-byte instruction. It decreases the 16-bit contents of a register pair by 1. R_p can be B, D, H and SP (stack pointer).

Que 1.33. Explain the various logical instructions related to rotating the accumulator bits.

Answer

These operations include instructions such as RLC, RRC, RAR and RAL.

i. **RLC (Rotate accumulator left) :** This instruction rotates the contents of accumulator left by one position. Bit D_7 is placed in the position of D_0 as well as in the carry flag.

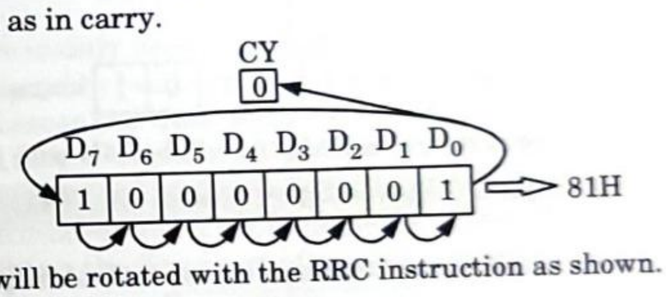


(*a*) Before the instruction.

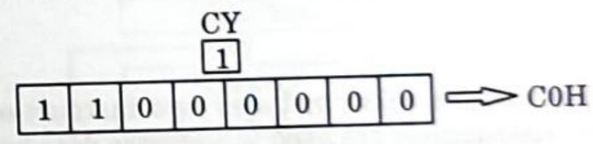
(*b*) After the RLC, (A) will be 55H with CY set.

Fig. 1.33.1. Accumulator contents after RLC.

ii. **RRC (Rotates accumulator right) :** This instruction rotates the contents of accumulator right by one position. Bit D_0 is placed in position of D_7 as well as in carry.



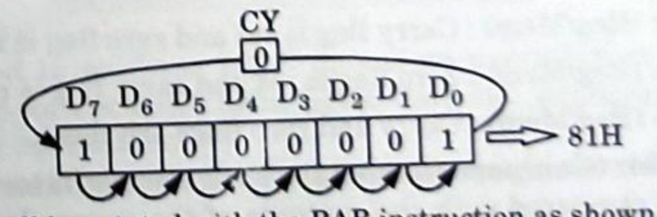
(*a*) A will be rotated with the RRC instruction as shown.



(*b*) After the execution of the instruction RRC, A will be C0H with the CY flag set.

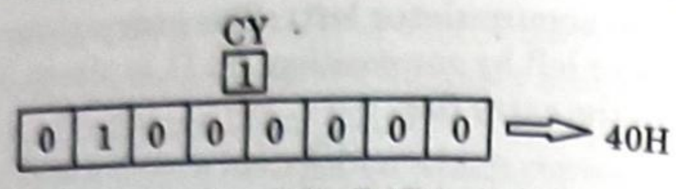
Fig. 1.33.2. Rotate right instructions.

iii. **RAR (Rotate accumulator right through carry) :** This instruction rotates the contents of the accumulator right by one position. Bit D_0 is placed in carry flag and the bit in carry flag is placed in most significant position D_7.



(*a*) A will be rotated with the RAR instruction as shown.

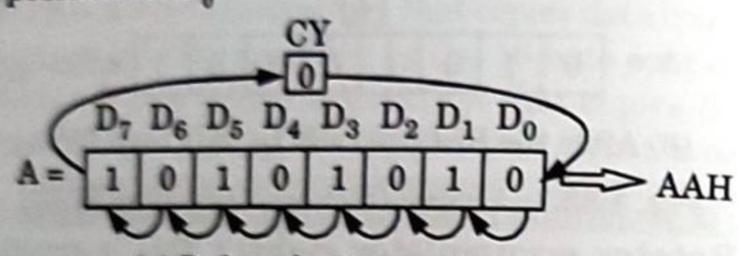
Scanned by TapScanner



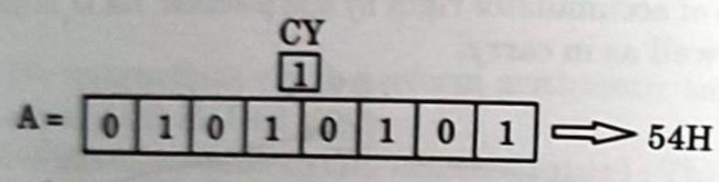
(b) After the execution of the RAR instruction, A will be 40H with the CY flag set.

Fig. 1.33.3. Rotate right instructions.

iv. **RAL (Rotate accumulator left through carry) :** This instruction rotates the contents of the accumulator left by one position through carry flag. Bit D_7 is placed in carry flag and carry flag is placed in least significant position bit D_0.



(a) Before the instruction.



(b) After the instruction RAL, (A) will be 54H with CY set.

Fig. 1.33.4. Accumulator contents after RLC.

Que 1.34. **Explain the compare operations of microprocessor 8085.**

Answer

The microprocessor 8085 includes two instructions CMP and CPI. These two instructions are used to compare data byte with contents of accumulator by subtracting the data byte or content of memory from accumulator and indicating the result by modifying the flags. However contents are not modified.

i. **CMP R/M (Compare with accumulator) :** This instruction compares the contents of specified register or memory location with the contents of accumulator and both the contents are preserved and comparison is shown by setting the flags.

If (A) < (Reg/Mem) : Carry flag is set and zero flag is reset.

If (A) = (Reg/Mem) : Zero flag is set and carry flag is reset.

If (A) > (Reg/Mem) : Carry and zero flags are reset.

CPI 8-bit (Compare immediate with accumulator) : The specified byte is compared with the contents of the accumulator. The values being compared remain unchanged and results of comparison are indicated by setting the flag as follows :

If (A) < Data : Carry flag is set and zero flag is reset.

If (A) = Data : Zero flag is set and carry flags is reset.

If (A) > Data : Carry and zero flags are reset.

PART-6

Counter and Time Delays, 8085 Interrupts.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.35. **Write a note on counter and time delay operation in microprocessor.**

Answer

Counters :

1. Counters are mainly used to keep track of events.
2. It can be designed by loading an appropriate number into any one register of microprocessor and then using INR or DCR instructions.
3. Whenever a loop is established using a register, end count is updated then every time count is checked to determine whether it has reached the required number and if not then loop is repeated again.
4. Fig. 1.35.1 shows the flow chart of counter operation.

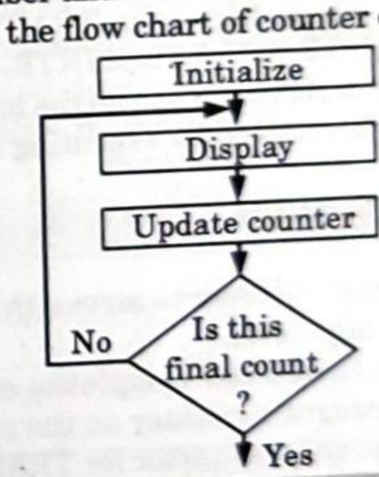


Fig. 1.35.1. Flow chart of a counter.

Time delay :

1. This is used to perform 'NOP' means no operation as real time applications such as traffic light, digital clock etc., requires some delay during their application.
2. Time delay can be generated using execution of instruction several times.
3. The flowchart in Fig. 1.35.2 shows the generation of time delay.

Scanned by TapScanner

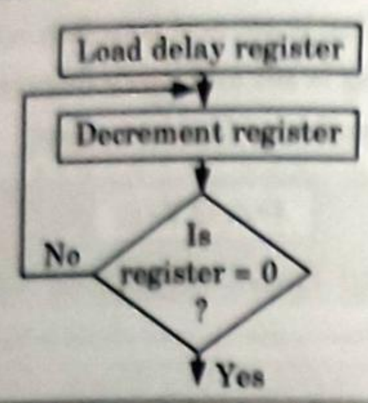


Fig. 1.35.2. Time delay loop flow diagram.

Que 1.36. **Define interrupt.**

OR

Explain the role of interrupts in programming. Explain the interrupts used in 8085. List out all the vectored interrupts of 8085 and give their vector address. **AKTU 2015-16, Marks 10**

Answer

A. **Interrupt :** Interrupts are the signals generated by external devices to request the microprocessor to perform a task.

B. **Role and types of interrupt :**

a. **Hardware interrupts :**

1. 8085 microprocessor have five interrupt pin namely TRAP, RST 7.5, RST 6.5, RST 5.5 and INTR.
2. If the signals on these pin go high then TRAP will be serviced first followed by RST 7.5, RST 6.5, RST 5.5 and INTR.
3. When all the interrupts are serviced then all the interrupts except TRAP are disabled. They can be enabled by executing enable interrupt (EI) instruction.

4. **Types of hardware interrupts :**

i. **TRAP :**

1. It is non-maskable interrupt. In order to service this interrupt the signal on TRAP pin must have high level.
2. If this condition occur then 8085 completes execution of current instruction, pushes the program counter on the stack and branches to location 0024H (Interrupt address vector for TRAP).

ii. **RST 7.5 :**

1. It is maskable interrupt. It can be enabled and disabled using SIM (Set interrupt mask) instruction.
2. In order to service RST 7.5, 8085 complete current execution of instruction, pushes the program counter onto the stack and branches to 003CH.

iii. **RST 6.5 :**

1. This can be enabled and disabled using SIM instruction.

2. RST 6.5 is high level sensitive, microprocessor 8085 execute current instruction, save the program counter onto the stack and branches to location 0034H.

iv. **RST 5.5 :**

1. It is maskable interrupt and can be enabled and disabled by SIM instruction. It is high level sensitive.
2. In order to service this interrupt 8085 completes the execution of current instruction, save the program counter onto the stack and branches to location 002CH.

v. **INTR :**

1. It is non-maskable interrupt. Whenever INTR signal is high then 8085 completes its current instruction and send active low interrupt acknowledge signal $\overline{INTA}$.
2. If interrupt is enabled in response to $\overline{INTA}$ signal, external logic places an instruction opcode on data bus on receiving the instruction.
3. 8085 save the address of next instruction onto stack and execute received instruction.

b. **Software interrupts :**

1. In software interrupts, the cause of the interrupt is an execution of the instruction. These are special instructions supported by the microprocessor.
2. After execution of these instructions, microprocessor completes the execution of the instruction it is currently executing and transfers the program control to the subroutine program.
3. After completion of the execution of the subroutine program, program control returns to the main program.
4. The 8085 has eight software interrupts from RST 0 to RST 7.
5. The vector address can be calculated as shown in Table 1.36.1 :

Interrupt number × 8 = Vector address

Table 1.36.1.

Instruction	Hex code	Vector address
RST 0	C7	0000H
RST 1	CF	0008H
RST 2	D7	0010H
RST 3	DF	0018H
RST 4	E7	0020H
RST 5	EF	0028H
RST 6	F7	0030H
RST 7	FF	0038H

Scanned by TapScanner

C. Vectored interrupts :

1. When the internal control circuit of processor produces a CALL to predetermined memory location which is the starting address of interrupt service routine, the address is called vector address and such interrupts are called vectored interrupts.

   Example : TRAP, RST 7.5, RST 6.5, RST 5.5

Interrupt type	Trigger	Priority	Maskable	Vector address
TRAP	Edge and level	1st (Highest)	No	0024H
RST 7.5	Edge	2nd	Yes	003CH
RST 6.5	Level	3rd	Yes	0034H
RST 6.5	Level	4th	Yes	002CH

2. The TRAP has the highest priority, followed by RST 7.5, 6.5, 5.5, and INTR, in that order; however, the TRAP has a lower priority than the HOLD signal used for DMA.

D. Non-vectored interrupts : When address of subroutine is received from the external device, is called non-vectored interrupt.
Example : INTR.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. Explain the microprocessor architecture and operation of its components.
Ans. Refer Q. 1.3.

Q. 2. What do you understand by memory ? Also give its classification.
Ans. Refer Q. 1.6.

Q. 3. With the neat pin and block diagram, describe the internal architecture of 8085. State the function of each block.
Ans. Refer Q. 1.10.

Q. 4. Explain the flags of 8085 microprocessor. Give the flag status when following additions are performed.
   i. 51H + A9H
   ii. 2EH + 5AH
   iii. 76H + A4H

Ans. Refer Q. 1.12.

Q. 5. Draw and explain the memory and I/O read cycle of 8085.
Ans. Refer Q. 1.16.

Q. 6. Write short note on logic devices for interfacing.
Ans. Refer Q. 1.17.

Q. 7. What do you mean by flow chart ?
Ans. Refer Q. 1.22.

Q. 8. Explain machine control instructions.
Ans. Refer Q. 1.28.

Q. 9. Write the programming techniques used in microprocessor.
Ans. Refer Q. 1.30.

Q. 7. Explain the role of interrupts in programming. Explain the interrupts used in 8085. List out all the vectored interrupts of 8085 and give their vector address.
Ans. Refer Q. 1.36.

☺☺☺

Scanned by TapScanner

2 UNIT

Intel 8086 Microprocessor

CONTENTS

Part-1 : Internal Architecture (Bus Interface Unit, Execution Unit, Pipelining, Register Organization), Pin Diagram ..... **2–2C to 2–17C**

Part-2 : Memory Addressing, Physical Memory Organization, Interrupts (Hardware and Software Interrupts) ..... **2–17C to 2–21C**

PART-1

Internal Architecture (Bus Interface Unit, Execution Unit, Pipelining, Register Organization), Pin Diagram.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.1. **Give the features of 8086.**

OR

Write the difference between 8085 and 8086 microprocessor.

Answer

A. Features :

1. The 8086 is a 16-bit microprocessor. The arithmetic logic unit, internal registers and most of its instruction are designed to work with 16-bit binary words.
2. The 8086 has a 16-bit data bus, so it can read data from or write data to memory and ports either 16-bits or 8-bits at a time.
3. The 8086 has a 20-bit address bus, so it can directly access 2^{20} or 1 MB memory locations.
4. The 8086 has multiplexed address and data bus which reduces the number of pins needed, but it slows down the transfer of data.
5. The 8086 requires one phase clock with a 33 % duty cycle to provide optimized internal timing.

B. Difference between 8085 & 8086 :

S. No.	8085 μP	8086 μP
1.	8085 is an 8-bit microprocessor.	8086 is a 16-bit microprocessor.
2.	8085 has 16 address lines.	8086 has 20 address lines.
3.	8085 has only five flags.	8086 has nine flags.
4.	It does not support pipelining.	It supports pipelining.
5.	8085 operates on clock cycle with 50 % duty cycle.	8086 operates on clock cycle with 33 % duty cycle.

Scanned by TapScanner

Que 2.2. **Draw the internal architecture of 8086 and discuss block diagram.**

Answer

A. The internal block diagram of 8086 :

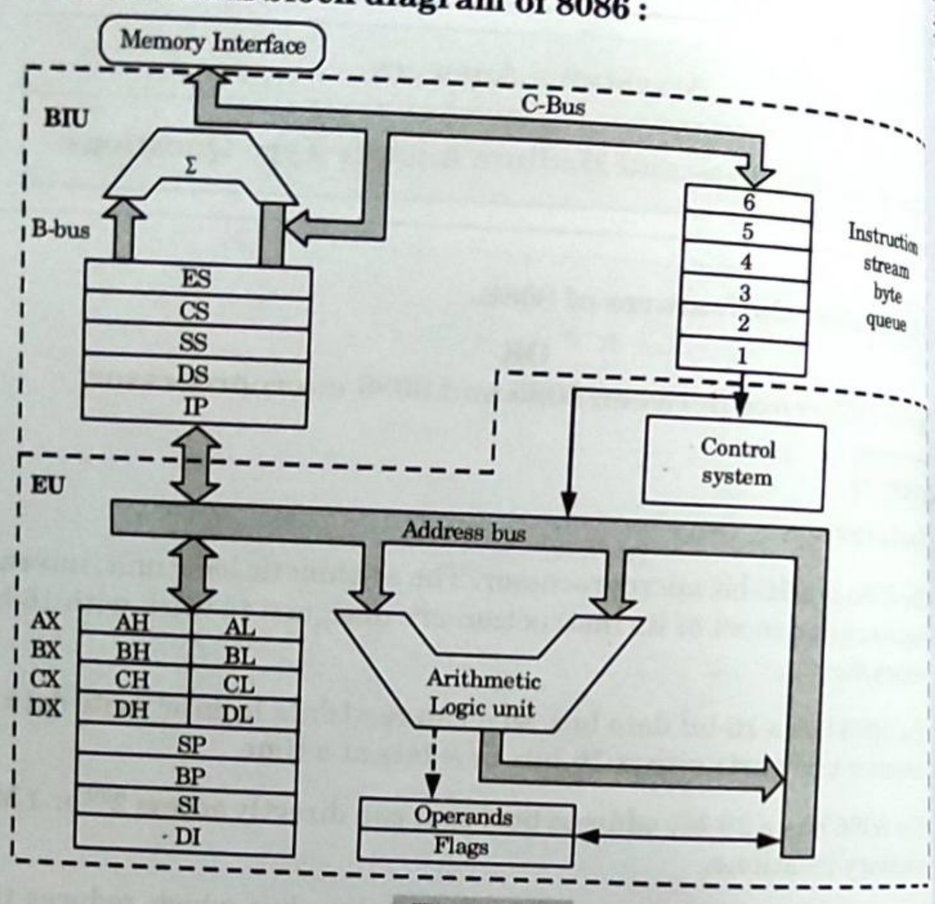


Fig. 2.2.1.

B. It is internally divided into two separate functional units:

1. Bus Interface Unit (BIU) :

a. The bus interface unit is the 8086's interface to the outside world. It provides a full 16-bit bi-directional data bus and 20-bit address bus.

b. The bus interface unit is responsible for performing all following external bus operations :

i. It sends address of the memory or I/O.
ii. It fetches instruction from memory.
iii. It reads data from port/memory.
iv. It writes data into port/memory.
v. It supports instruction queuing.
vi. It provides the address relocation facility.

c. To implement these functions, the BIU contains the instruction queue, segment registers, instruction pointer, address summer and bus control logic.

2. Execution Unit (EU) : The execution unit of 8086 tells the BIU from where to fetch instructions or data, decodes instructions and executes instructions. It contains :

i. Control circuitry
ii. Instruction decoder
iii. Arithmetic logic unit (ALU)
iv. Registers organization :
a. Flag registers
b. General purpose registers
c. Pointers and index register.

3. The control circuitry in the EU directs the internal operations. A decoder in the EU translates the instructions fetched from memory into a series of actions which the EU performs.

4. ALU is 16-bit. It can add, subtract, AND, OR, XOR, increment, decrements, complement and shift binary numbers.

Que 2.3. **Give the pin diagram for 8086 microprocessor and explain the pins.**

Answer

A. Pin diagram : The functional signal description of pin diagram of microprocessor 8086 is given in Fig. 2.3.1.

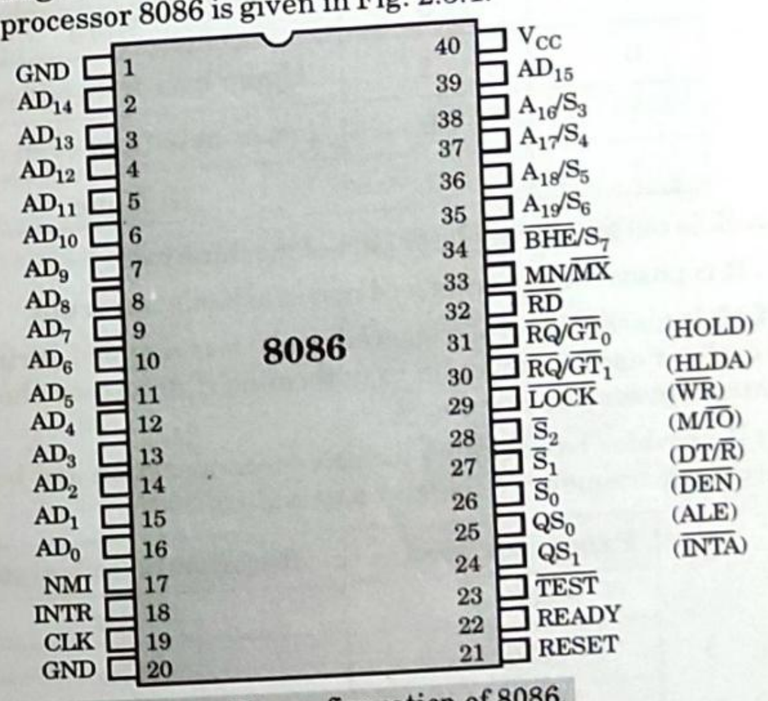


Fig. 2.3.1. Pin configuration of 8086.

B. Pins description :

1. **$AD_{15} - AD_0$:** These are time multiplexed memory I/O address and data lines which act as address but during first part of machine cycle and data bus for remaining part of the machine cycle.

Scanned by TapScanner

2. $A_{19}/S_6 - A_{16}/S_3$: These are used to output upper 4-bits of addr during first part of machine cycle. During remaining part of the mach cycle these are used to output status, which indicates the type operation to be performed.

S_3 and S_4 : S_3 and S_4 indicates the segment register being used follows :

S_4	S_3	Register
0	0	ES
0	1	SS
1	0	CS or none
1	1	DS

3. $\overline{BHE}/S_7$ (Bus High Enable/Status) : Low signal on this pin durin first part of machine cycle indicates that at least one of the curre transfer is to be made on higher order byte $AD_{15} - AD_8$, otherwi transfer is made on low order byte $AD_7 - AD_0$.

$\overline{BHE}$	A_0	Data accesses
0	0	Word
0	1	Upper byte from odd address
1	0	Lower byte from even address
1	1	None

Status S_7 is output during later part of machine cycle.

4. NMI : It is positive edge triggered non-maskable interrupt.
5. INTR : It is maskable level triggered input. It is sampled during the las clock cycle of each instruction to determine if processor should enter into interrupt service routine.
6. CLK : It provides basic timing for processor operation and bus contro activity clock frequency depends on version of 8086.

Processor 8086	Required clock signal
8086	5 MHz
8086-2	8 MHz
8086-1	10 MHz

7. RESET : It causes the processor to terminate the current activity and start execution from FFFF0H. This is active high signal and must be active for at least four clock cycles.

8. READY : This signal is used primarily to synchronize slower peripherals with the microprocessor.
9. $\overline{TEST}$: This input is examined by a WAIT instruction. The 8086 enters into a wait state after execution of WAIT instruction until a low signal is applied on $\overline{TEST}$ pin.
10. $\overline{RD}$: Low on $\overline{RD}$ pin indicates 8086 is reading data from memory or an I/O device.
11. MN/$\overline{MX}$: This pin decides whether 8086 is to be operated in minimum mode or maximum mode.

Que 2.4. **Draw the register organization of 8086.**

OR

Explain pipelining architecture of 8086 microprocessor. Show bit-wise flag register of 8086 and explain each of them.

AKTU 2018-19, Marks 07

Answer

A. General data registers :

1. The registers AX, BX, CX and DX are the general purpose 16-bit registers.
2. Each 16-bit register can be split into two 8-bit registers. Letter L and H specify the lower and higher bytes.
3. These registers are either used for holding data, variables, and intermediate results temporarily, or as counters.

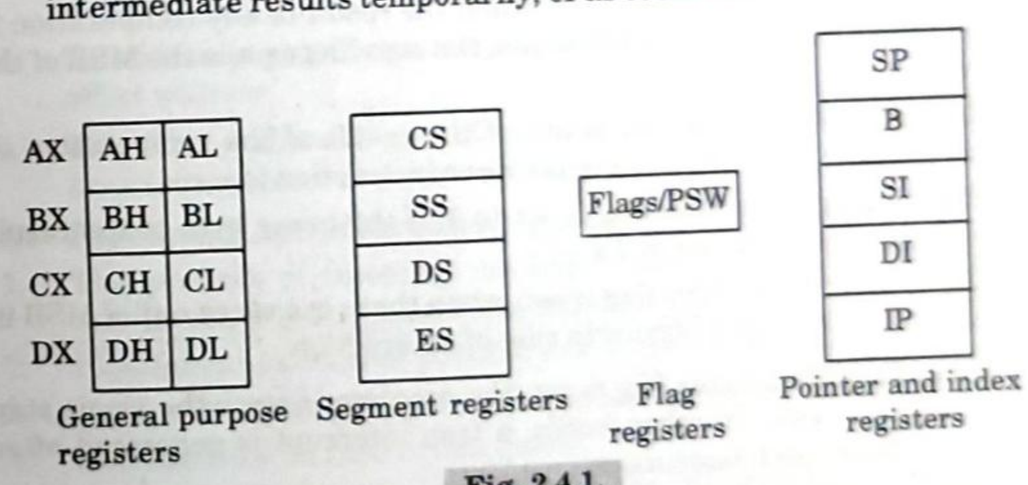


Fig. 2.4.1.

B. Segment registers : There are four types of segment registers :

1. CS : The code segment register is used for addressing a memory location in code segment of the memory, where the executable program is stored.

Scanned by TapScanner

2. **DS :** The data segment register points to the data segment of the memory, where the data is resided.
3. **ES :** The extra segment refers to a segment which essentially is another data segment of the memory. Thus, the extra segment also contains data.
4. **SS :** The stack segment register is used for addressing stack segment of memory. The stack segment is that segment of memory which is used to store stack data.

C. Pointer and Index registers :

1. The pointers contain offset within the particular segments.
2. The pointers IP, BP and SP usually contain offsets within the code, data and stack segments respectively.
3. The index registers are used as general purpose registers as well as for offset storage.
4. The register SI is generally used to store the offset of source data in data segment while the register DI is used to store the offset of destination in data or extra segment.

D. Flag registers :

i. The 8086 flag register contents indicate the result of computations in the ALU. It also contains some flag bits to control the CPU operations.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
×	×	×	×	O	D	I	T	S	Z	×	AC	×	P	×	CY

ii. **The description of each flag bit is as follows :**

1. **Sign flag (S) :** This flag is set, when the result of any computation is negative. For signed computations, the sign flag equals the MSB of the result.
2. **Zero flag (Z) :** This flag is set, if the result of the computation or comparison performed by the previous instruction is zero.
3. **Parity flag (P) :** This flag is set to 1, if the lower byte of the result contains even number of 1's.
4. **Carry flag (CY) :** This flag is set, when there is a carry out of MSB in case of addition or a borrow in case of subtraction.
5. **Trap flag (T) :** If this flag is set, the processor enters the single step execution mode. In other words, a trap interrupt is generated after execution of each instruction.
6. **Interrupt flag (I) :** If this flag is set, the maskable interrupts are recognized by the CPU, otherwise they are ignored.
7. **Direction flag (D) :** This is used by string manipulation instructions. If this flag bit is '0', the string is processed beginning from the lowest address to the highest address, means auto-incrementing mode.

Otherwise, the string is processed from the highest address towards the lowest address means auto-decrementing mode.

8. **Auxiliary carry flag (AC) :** This is set, if there is a carry from the lowest nibble means bit three.
9. **Overflow flag (O) :** This flag is set, if an overflow occurs.

E. Architecture of 8086 : Refer Q. 2.2, Page 2–3C, Unit-2.

Que 2.5. **Draw architecture of 8086 and explain its different unit. What do you mean by pipelining and explain the concept of memory segmentation.** **AKTU 2017-18, Marks 10**

OR

With the neat block diagram describe the internal architecture of 8086. State the function of each block shown. Explain the use of instruction queue. **AKTU 2015-16, Marks 10**

Answer

A. Architecture : Refer Q. 2.2, Page 2–3C, Unit-2.

B. Memory Segmentation :

1. In the segmented addressing, on the other hand, the available memory space is divided into "chunks" called segments. Such a memory is known as segmented memory.
2. In 8086 system the available memory space is 1MB. This memory is divided into number of logical segments.
3. Each segment is 64 kB in size and addressed by one of the segment registers.
4. To address a specific memory location within a segment we need an offset address.
5. The offset address is also 16-bit wide and it is provided by one of the associated pointer or index register.

C. Pipelining and instruction queue :

1. The process of fetching the next instruction when the present instruction is being executed is called as pipelining.
2. Pipelining has become possible due to the use of 6 byte queue.
3. Instruction queue is a process to speed up the processor.
4. BIU contains an instruction queue.
5. When the EU executes instructions, the BIU gets up to 6 bytes of the next instruction and stores them in the instruction queue.
6. The fetching of opcodes well in advance, prior to their need for execution increases the overall efficiency of the processor boosting its speed.
7. The processor no longer has to wait for the memory access operations for the subsequent instruction opcode to complete.

Scanned by TapScanner

Que 2.6. **Explain the function of pins for minimum mode operation of 8086 μP.**

Answer

1. **INTA (Interrupt acknowledge output) :** This indicates recognition of an interrupt request. Whenever this goes low, it means that 8086 has accepted the interrupt.
2. **ALE (Address latch enable output) :** This signal is provided by 8086 to demultiplex $AD_0 - AD_{15}$ into $A_0 - A_{15}$ and $D_0 - D_{15}$ using external latch.
3. **$\overline{DEN}$ (Data enable output) :** This signal informs the transceiver that the CPU is ready to send or receive data.
4. **DT/$\overline{R}$ (Data transmit/receive) output :** This is used to decide the direction of data flow through the transceivers. When the processor sends out the data this signal is high and when the processor is receiving data this signal is low.
5. **M/$\overline{IO}$:** It is used to distinguish memory data transfer, (M/$\overline{IO}$ = High) and I/O data transfer (M/$\overline{IO}$ = Low).
6. **$\overline{WR}$:** This signal is low when 8086 is writing into the memory or I/O device.
7. **HOLD Input, HLDA Output :**
   i. A high on HOLD pin indicates that another master is requesting to take over the system bus.
   ii. On receiving HOLD signal, processor output HLDA signal high as acknowledgement. At the same time processor tri-states the system bus.
   iii. A low on HOLD gives the system bus control back to the processor. Then output is low signal on HLDA.

Que 2.7. **Explain function of pins for maximum mode operation of 8086 μP.**

Answer

1. **QS_1, QS_0 (Output) :** These two output signals reflect the status of the instruction queue.

QS_1	QS_0	Status
0	0	No operation (queue is idle)
0	1	First byte of an opcode
1	0	Queue is empty
1	1	Subsequent byte of an opcode

2. **$\overline{S_2}, \overline{S_1}, \overline{S_0}$ (Output) :** These three status signals indicates the type of transfer to be take place during the current bus cycle.

$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	Machine cycle
0	0	0	Interrupt acknowledge
0	0	1	I/O Read
0	1	0	I/O Write
0	1	1	Halt
1	0	0	Instruction fetch
1	0	1	Memory read
1	1	0	Memory write
1	1	1	Inactive-passive

3. **$\overline{LOCK}$:** This signal indicates that an instruction with $\overline{LOCK}$ prefix being executed and the bus is not to be used by another processor.
4. **$\overline{RQ}/\overline{GT_1}$ and $\overline{RQ}/\overline{GT_0}$:** In maximum mode, HOLD and HLDA pins are replaced by $\overline{RQ}$ (bus request) / $\overline{GT_0}$ (bus grant) and $\overline{RQ}/\overline{GT_1}$ signal. These lines are bi-directional and are used to both request and grant a DMA operation.

Que 2.8. **Explain minimum mode operation of 8086 microprocessor with block diagram.** **AKTU 2017-18, Marks 10**

Answer

1. In a minimum mode 8086 system, the microprocessor 8086 is operated in minimum mode by strapping its MN/ $\overline{MX}$ pin to logic 1.
2. In this mode, all the control signals are given out by the microprocessor chip itself.
3. There is a single microprocessor in the minimum mode system.
4. The remaining components in the system are latches, transceivers, clock generator, memory and I/O devices.
5. The latches are generally buffered output D-type flip-flops, like, 74LS373 or 8282.
6. They are used for separating the valid address from the multiplexed address/data signals and are controlled by the ALE signal generated by 8086.

Scanned by TapScanner

7. Transceivers are the bidirectional buffers and sometimes they are called data amplifiers.
8. They are required to separate the valid data from the time multiplexed address/data signal.
9. They are controlled by two signals, namely, $\overline{DEN}$ and DT/R.
10. The $\overline{DEN}$ signal indicates that the valid data is available on the data bus, while DT/R indicates the direction of data, *i.e.*, from / to the processor.
11. The system contains memory for the monitor and users program storage.
12. Usually, EPROMS are used for monitor storage, while RAMs for users program storage.
13. A system may contain I/O devices for communication with the processor as well as some special purpose I/O devices.

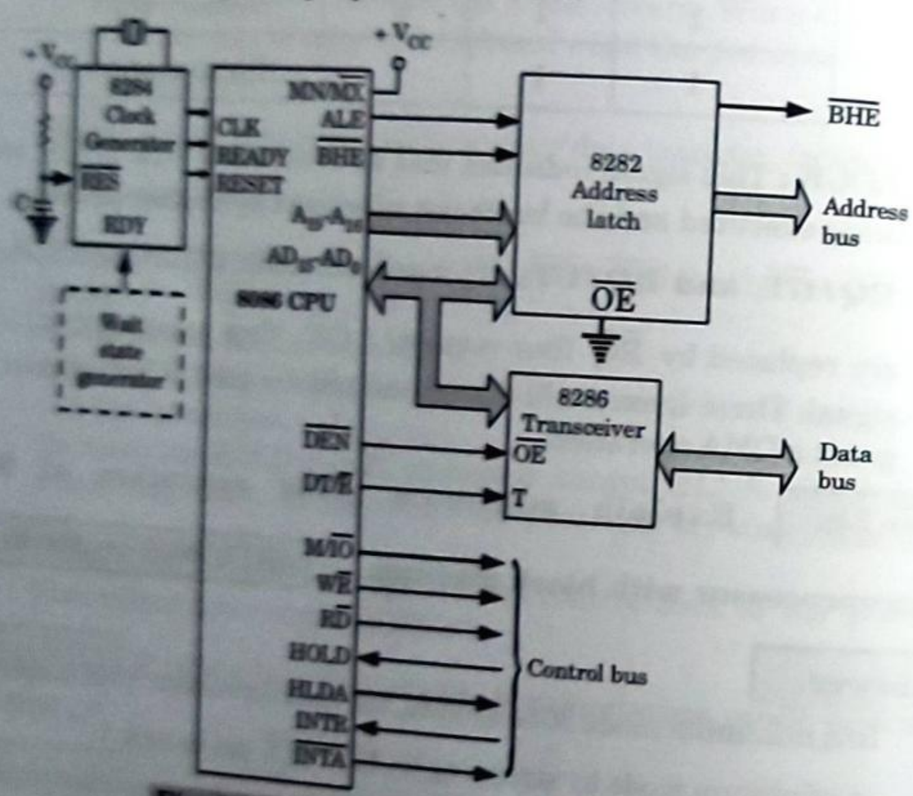


Fig. 2.8.1. Minimum mode 8086 system.

14. The clock generator (IC8284) generates the clock from the crystal oscillator and then shapes it to make it more precise so that it can be used as an accurate timing reference for the system.
15. The clock generator also synchronizes some external signals with the system clock.

Que 2.9. Explain the minimum mode read and write bus cycle with proper timing diagram.

Answer

A. Read cycle :

1. The read cycle begins the T_1 with the assertion of the address latch enable (ALE) signal and also M/$\overline{IO}$ signal.
2. During the negative going edge of this signal, the valid address is latched on the local bus.
3. The $\overline{BHE}$ and A_0 signals addresses low, high or both bytes.
4. From T_1 to T_4, the M/$\overline{IO}$ signal indicate a memory or I/O operation.
5. At T_2, the address is removed from the local bus and is sent to the output.
6. The bus is then tristated.
7. The read $\overline{RD}$ control signal is also activated in T_2.
8. The read $\overline{RD}$ signal causes the addressed device to enable its data bus drivers.
9. After $\overline{RD}$ goes low, the valid data is available on the data bus.
10. The addressed device will drive the READY line high.
11. When the processor returns the read signal to high level, the addressed device will again tristate its bus drivers.

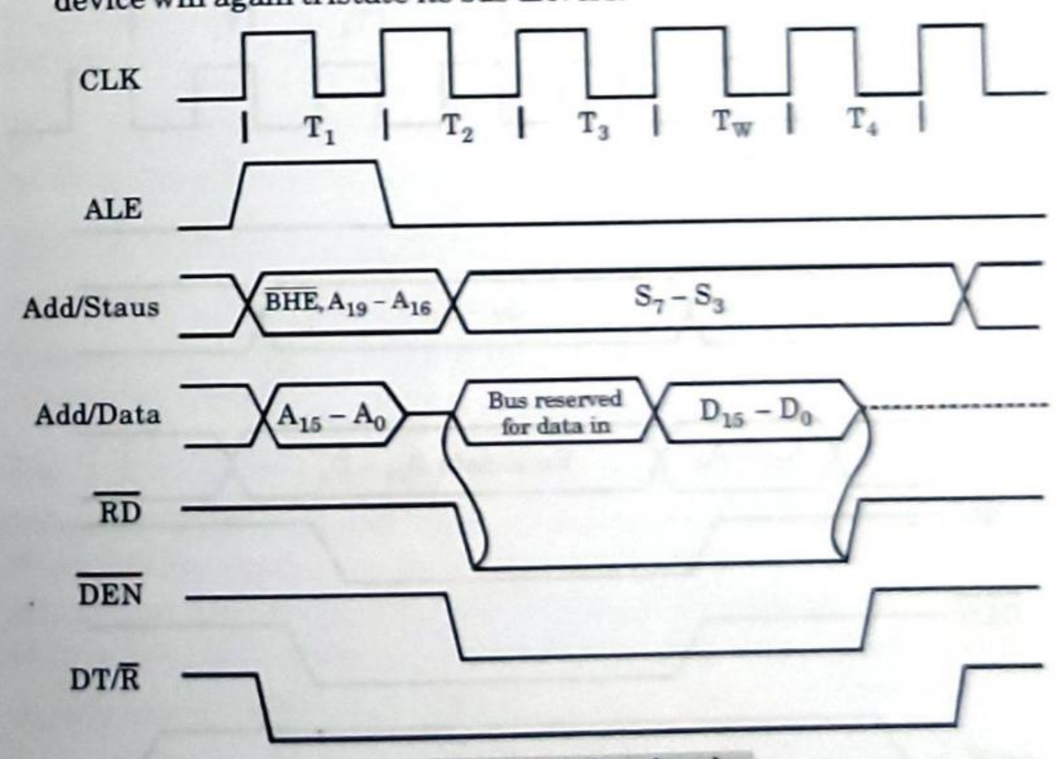


Fig. 2.9.1. Read cycle.

B. Write cycle :

1. A write cycle also begins with the assertion of ALE and the emission of the address.

Scanned by TapScanner

2. The M/$\overline{IO}$ signal is again asserted to indicate a memory or I/O operation.
3. In T_2, after sending the address in T_1, the processor sends the data to be written to the addressed location.
4. The data remains on the bus until middle of T_4 state.
5. The $\overline{WR}$ becomes active at the beginning of T_2.
6. The $\overline{BHE}$ and A_0 signals are used to select the proper byte or bytes of memory or I/O word to be read or written.
7. The M/$\overline{IO}$, $\overline{RD}$ and $\overline{WR}$ signals indicate the types of data transfer as described in Table. 2.9.1.

Table 2.9.1.

M/$\overline{IO}$	$\overline{RD}$	$\overline{WR}$	Transfer type
0	0	1	I/O read
0	1	0	I/O write
1	0	1	memory read
1	1	0	memory write

8. The timing diagram of write cycle is shown in Fig. 2.9.2.

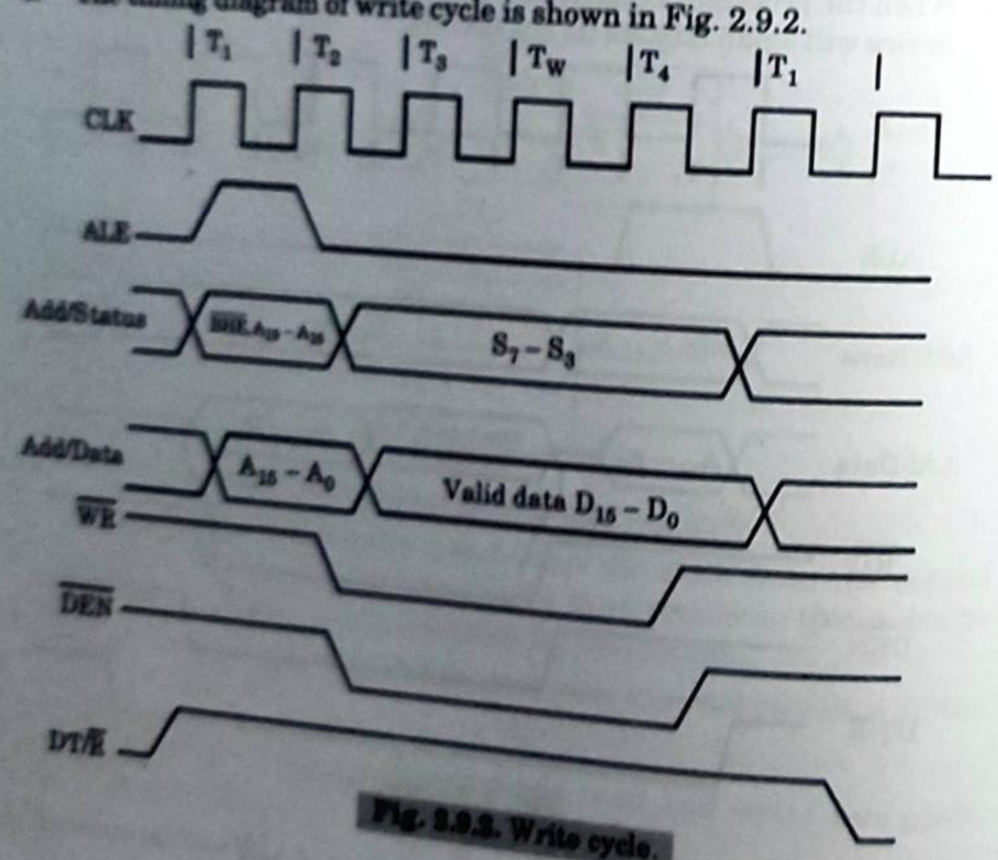


Fig. 2.9.2. Write cycle.

Que 2.10. **Explain maximum mode operation of 8086 microprocessor with block diagram.**

Answer

1. In the maximum mode, the 8086 is operated by strapping the MN / $\overline{MX}$ pin to ground. In this mode, the processor derives the status signals $\overline{S}_2, \overline{S}_1$ and $\overline{S}_0$.
2. Another chip called bus controller derives the control signals using this status information.
3. The basic functions of the bus controller chip IC8288, is to derive control signals like $\overline{RD}$ and $\overline{WR}$ (for memory and I/O devices), $\overline{DEN}$, DT / $\overline{R}$ ALE, etc., using the information made available by the processor on the status lines.
4. The bus controller chip has input lines $\overline{S}_2, \overline{S}_1$ and $\overline{S}_0$ and CLK, which are driven by the CPU.
5. $\overline{AEN}$ and IOB are generally grounded. CEN pin is usually tied to + 5V.
6. The significance of the MCE/PDEN output depends upon the status of the IOB pin.
7. $\overline{INTA}$ pin is used to issue two interrupt acknowledge pulses to the interrupt controller or to an interrupting device.
8. $\overline{IORC}$, $\overline{IOWC}$ are I/O read command and I/O write command signals respectively.
9. These signals enable an IO interface to read or write the data from or to the addressed port.
10. The $\overline{MRDC}$, $\overline{MWTC}$ are memory read command and memory write command signals respectively and may be used as memory read and write signals.
11. All these command signals instruct the memory to accept or send data to or from the bus. For both of these write command signals, the advanced signals namely $\overline{AIOW}$ and $\overline{AMWTC}$ are available.
12. They also serve the same purpose, but are activated one clock cycle earlier than the $\overline{IOWC}$ and $\overline{MWTC}$ signals, respectively.

Scanned by TapScanner

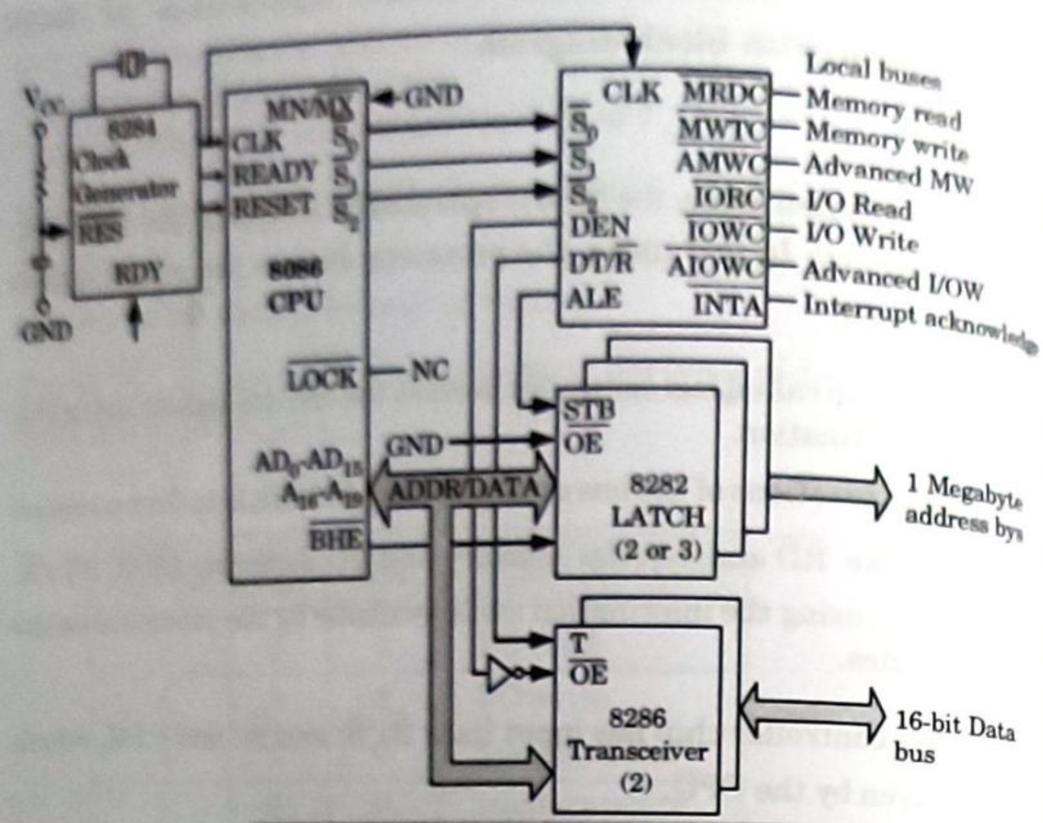


Fig. 2.10.1. Maximum mode 8086 system.

Que 2.11. **Explain minimum and maximum operating modes of 8086 with timing diagram.** **AKTU 2017-18, Marks 10**

Answer

A. Minimum operating modes : Refer Q. 2.9, Page 2–11C, Unit-2.

B. Maximum operating modes :

1. S_0, $\overline{S}_1$, $\overline{S}_2$ are set at the beginning of bus cycle.
2. On detecting the change on passive state $S_0 = \overline{S}_1 = \overline{S}_2 = 1$, the 8288 bus controller will output a pulse on its ALE and apply a required signal to its DT/$\overline{R}$ pin during T_1.
3. In T_2, 8288 will set DEN = 1 thus enabling transceiver. For an input, 8288 it will activates $\overline{MRDC}$ or $\overline{IORC}$.

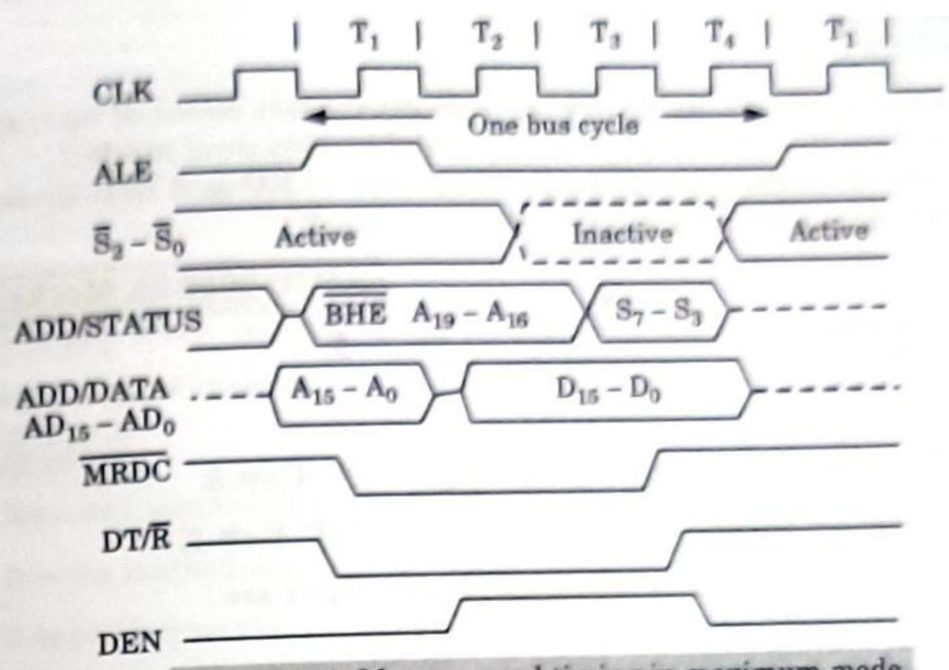


Fig. 2.11.1. Memory read timing in maximum mode.

4. These signals are activated until T_4. For an output, the $\overline{AMWC}$ or $\overline{IOWC}$ is activated from T_3 to T_4.
5. The status bits $\overline{S}_2$ to $\overline{S}_0$ remain active until T_3, and become passive during T_3 and T_4.
6. If ready input is not activated before T_3, wait state will be inserted between T_3 and T_4.

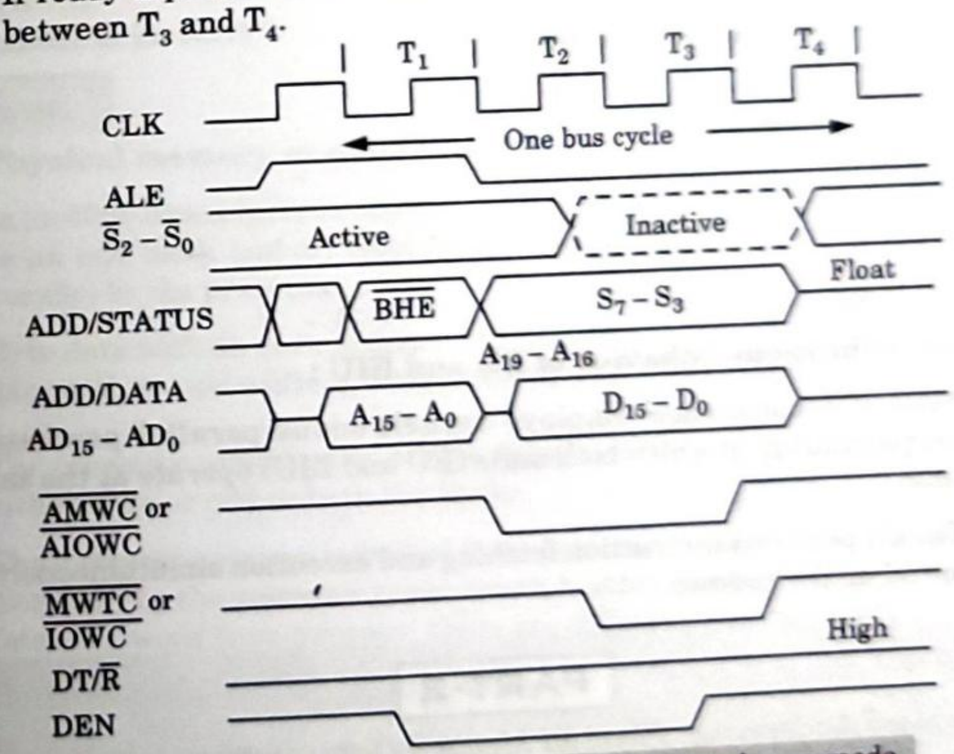


Fig. 2.11.2. Memory write timing in maximum mode.

Scanned by TapScanner

Que 2.12.

i. What is meant by minimum and maximum mode of operation in 8086 IC ? Which pins are affected by this dual mode ?

ii. How does the asynchronous behavior of EU and BIU increase the throughput of 8086 microprocessor ?

AKTU 2018-19, Marks 07

Answer

i.

a. Minimum mode : Refer Q. 2.8, Page 2–10C, Unit-2.

b. Maximum mode : Refer Q. 2.10, Page 2–14C, Unit-2.

c. Pins are affected by this dual mode : There are :

1. AD_{15}-AD_0
2. A_{19}/S_6, A_{18}/S_5, A_{17}/S_4, A_{16}/S_3
3. BHE/S_7-Bus high enable/status
4. $\overline{RD}$ -Read
5. Ready
6. INTR-Interrupt request
7. $\overline{TEST}$
8. NMI-Non maskable interrupt
9. RESET
10. CLK- Clock input
11. V_{cc}
12. GND
13. MN/$\overline{MX}$.

ii. Asynchronous behavior of EU and BIU :

1. 8086 microprocessor employs asynchronous parallel processing (or pipelining) in which both units (EU and BIU) operate at the same time.
2. Thus it performs instruction fetching and execution simultaneously to speed up the process.

PART-2

Memory Addressing, Physical Memory Organization, Interrupts (Hardware and Software Interrupts).

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.13. What do you mean by memory addressing ? Explain memory segmentation.

Answer

A. Memory addressing :

1. It is the method of specifying the memory location of an operand.
2. It explains how the memory space is available to the processor.

B. Types :

i. **Linear addressing :** In linear addressing the entire memory space is available to the processor in one linear array.

ii. **Segmented addressing or Memory segmentation :** Refer Q. 2.5, Page 2–8C, Unit-2.

Que 2.14. What do you understand by physical memory organization in 8086 ?

Answer

Physical memory organization :

1. In an 8086 based system, the 1 Mbytes memory is physically organized as an odd bank and an even bank, each of 512 kbytes, addressed in parallel by the processor.
2. Byte data with an even address is transferred on D_7–D_0, while the byte data with an odd address is transferred on $D_{15} - D_8$ bus lines.
3. The processor provides two enable signals, $\overline{BHE}$ and A_0 for selection of either even or odd or both the banks.
4. The instruction stream is fetched from memory as words and is addressed internally by the processor as necessary. In other words, if microprocessor fetches a word from memory, there are different possibilities :

i. Both the bytes may be data operands.

ii. Both the bytes may be containing opcode bits

iii. One of the bytes may be opcode while the other may be data.

Scanned by TapScanner

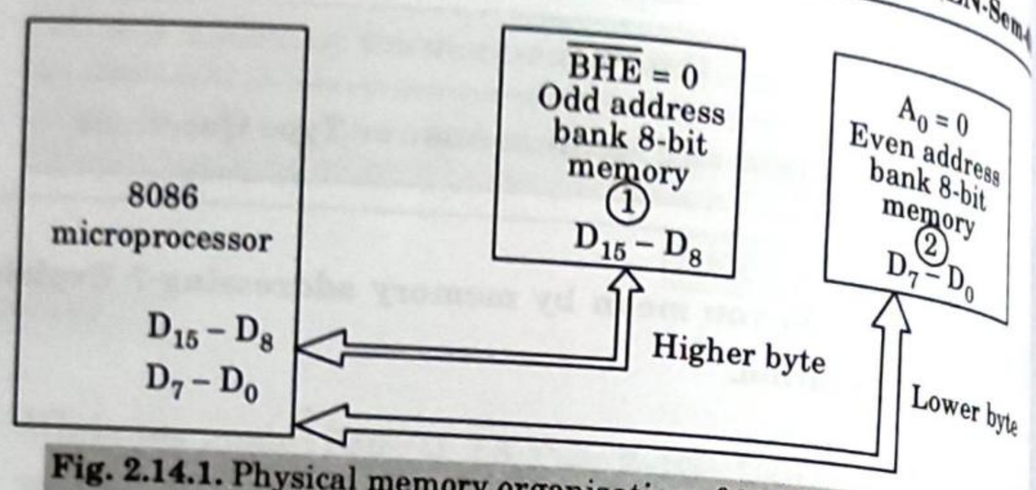


Fig. 2.14.1. Physical memory organization of 8086 microprocessor.

5. 8086 microprocessor is a 16-bit microprocessor and hence can access two bytes of data in one memory or I/O read or write operation. But the commercially available memory chips are only byte size, *i.e.*, they can store only one byte in a memory location.
6. To store 16-bit data, two successive memory locations are used and the lower byte of 16-bit data can be stored in the first memory location while the second byte is stored in the next location.
7. In a sixteen bit read or write operation both of these bytes will be read or written in a single machine cycle.
8. A map of an 8086 memory system starts at (00000) H and ends at (FFFFFH) H being a 16-bit processor is expected to access 16-bit data to / from 8-bit commercially available memory chips in parallel, as shown in Fig. 2.14.1.

Table 2.14.1. Selection of physical memory

$\overline{BHE}$	A_0	Function
0	0	Whole word
0	1	Upper byte from to odd address
1	0	Lower byte from to even Address
1	1	None

Que 2.15. **Explain the interrupts sequence and types of interrupts in 8086.**

AKTU 2016-17, Marks 10

AKTU 2017-18, Marks 10

Answer

A. Interrupt :

1. Whenever a microprocessor is interrupted, it stops executing its current program and calls a special routine which services the interrupt. The event that causes the interruption is called interrupt and the special routine executed to service the interrupt is called interrupt service routine/procedure.
2. Normally program can be interrupted in three ways :

i. By external signal.

ii. By special instruction in program.

iii. By occurrence of some condition.

B. Interrupt sequence in 8086 :

The Interrupt sequence in an 8086 system is described as follows :

1. One or more IR lines are raised high that set corresponding IRR bits.
2. 8086 resolves priority and sends an INT signal to CPU.
3. The CPU acknowledge with INTA pulse.
4. Upon receiving an INTA signal from the CPU, the highest priority ISR bit is set and the corresponding IRR bit is reset. The 8086 does not drive data during this period.
5. The 8086 will initiate a second INTA pulse. During this period 8086 releases an 8-bit pointer on to a data bus from where it is read by the CPU.
6. This completes the interrupt cycle.
7. The ISR bit is reset at the end of the second INTA pulse if automatic end of interrupt (AEOI) mode is programmed.
8. Otherwise ISR bit remains set until an appropriate EOI command is issued at the end of interrupt subroutine.

B. Types :

1. **Hardware Interrupt :** Microprocessor 8086 can get interrupt from external signal applied to non maskable interrupt (NMI) input pin or interrupt (INTR) input pin.
2. **Divide by Zero Interrupt (Type 0) :** When the quotient from either a DIV or IDIV instruction is too large to fit in result register, 8086 will automatically execute type 0 interrupt.
3. **Single Step Interrupt (Type 1) :**

i. Type 1 interrupt is a single step TRAP.

Scanned by TapScanner

ii. In the single step mode, system will execute one instruction and wait for further direction from user.

iii. Then user can examine the contents of register and memory locations, they are correct, user can tell the system to execute next instruction.

iv. This feature is useful for debugging.

v. An 8086 system is used in single step mode by setting trap flag and then 8086 will automatically execute type 1 interrupt.

vi. To reset the TRAP flag we have to reset bits 8, this can be done using AND operation.

4. Software Interrupts (Type 0–255) :

i. The 8086 INT instruction can be used to cause the 8086 to do one of the 256 possible interrupt types.

ii. The desired interrupt types are specified as a part of the instruction.

iii. For example the instruction INT 32 will cause the 8086 to do a type 32 interrupt response.

iv. The 8086 will push the flag register on the stack, reset TF and IF and push the CS and IP values for the start of interrupt service procedure from interrupt pointer table in memory.

v. The IP value for any interrupt type is always at an address of 4 times the interrupt type and the CS value is at the location two address higher.

5. Non Maskable Interrupt (Type 2) :

i. As the name suggest interrupt can't be disabled by any software instruction.

ii. This interrupt is activated by low to high transition on 8086 NMI input pin.

iii. In response 8086 will do type 2 interrupt.

6. Maskable Interrupt (INTR) :

i. The 8086 INTR input can be used to interrupt a program execution.

ii. This interrupt can be implemented using two pins INTR and $\overline{INTA}$.

iv. This interrupt can be enabled and disabled using STI (IF = 1) or CLI (IF = 0).

4. When 8086 is reset, interrupt flag is automatically cleared (IF = 0) so, after reset INTR is disabled, user have to execute STI function to enable INTR interrupt.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. Give the pin diagram for 8086 microprocessor and explain the pins.

Ans. Refer Q. 2.3.

Q. 2. Explain pipelining architecture of 8086 microprocessor. Show bit-wise flag register of 8086 and explain each of them.

Ans. Refer Q. 2.4.

Q. 3. Draw architecture of 8086 and explain its different unit. What do you mean by pipelining and explain the concept of memory segmentation.

Ans. Refer Q. 2.5.

Q. 4. Explain minimum mode operation of 8086 microprocessor with block diagram.

Ans. Refer Q. 2.8.

Q. 5.

i. What is meant by minimum and maximum mode of operation in 8086 IC ? Which pins are affected by this dual mode ?

ii. How does the asynchronous behavior of EU and BIU increase the throughput of 8086 microprocessor ?

Ans. Refer Q. 2.12.

Q. 6. What do you understand by physical memory organization in 8086 ?

Ans. Refer Q. 2.14.

Q. 7. Explain the interrupts sequence and types of interrupts in 8086.

Ans. Refer Q. 2.15.

☺☺☺

Scanned by TapScanner

3 UNIT

Peripheral Devices

CONTENTS

Part-1 : 8237 DMA Controller, 8255 Programmable Peripheral Interface, 8253/8254 Programmable Timer/Counter ..... **3–2C to 3–20C**

Part-2 : 8259 Programmable Interrupt Controller, 8251 USART and RS232C ..... **3–20C to 3–31C**

Part-3 : Fundamental of Programming : Program Structure and Programming Techniques for Microprocessors, 8085 Addressing Modes, 8085 Instruction Set ..... **3–32C to 3–35C**

Part-4 : Assembly Language Programming of 8085 Microprocessor with Examples (Arithmetic Operations on 8-bit Numbers-add, Subtract, Multiply, Divide, Square and Square Root etc, Largest/Smallest Number, Ascending/Descending Order) ..... **3–35C to 3–51C**

PART-1

8237 DMA Controller, 8255 Programmable Peripheral Interface, 8253/8254 Programmable Timer/Counter.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.1. **What do you understand by DMA ? With the help of block diagram explain the working of 8237/8257.**

AKTU 2015-16, Marks 15

OR

Give the features and functional block diagram of 8237 DMA controller.

AKTU 2017-18, Marks 10

Answer

A. DMA :

1. To transfer large block of data at high speed, a special control unit may be provided to allow transfer of a block of data directly between an external device and the main memory, without continuous intervention by the processor. This approach is called direct memory access or DMA.
2. DMA transfers are performed by a control circuit associated with the I/O device and this circuit is referred as DMA controller.

B. Features :

1. Intel 8237, direct memory access (DMA) controller, enables data transfer between memory and the I/O with reduced load on the system's main processor by providing the memory with control signals and memory address information during the DMA transfer.
2. The 8237 is a four-channel device that can be expanded to include any number of DMA channel inputs.
3. The 8237 is capable of DMA transfers at rates of up to 1.6 megabyte per second.

C. Architecture of 8237 :

i. The internal block diagram is shown in Fig. 3.1.1.

Scanned by TapScanner

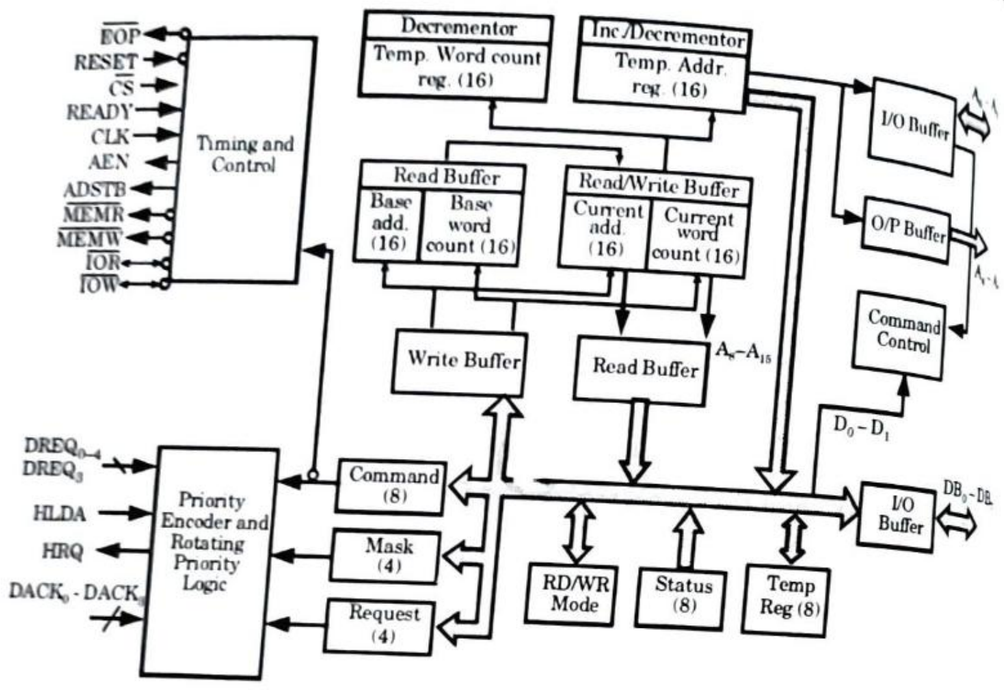


Fig. 3.1.1. Block diagram of 8237.

ii. 8237A contain three basic block of control logic :

a. **Timing and control block :** It generates internal timing and external control signal to the 8237A.

b. **Program command control block :** It decodes various commands given to the 8237 by the microprocessor before servicing a DMA request. It also decodes the mode control word, which is used to select the type of DMA during the servicing.

c. **Priority encoder block :** It resolves the priority between DMA channels requesting the service.

d. **Internal registers :** 8237 contain 344 bits of internal memory in the form of register. Various types of register are :

1. **Current address register :** Each of 4 DMA channels of 8237 has a 10-bit current address register that hold the current memory address, being accessed during DMA transfer.
2. **Current word register :** Each channel has 16-bit current word register that holds the number of data byte transfers to be carried out.
3. **Base address and base word count register :** Each channel has a pair of these registers. They maintain an original copy of respective initial current address register and current word register respectively.
4. **Command register :** This 8-bit register controls the complete operation of 8237. This can be programmed operation.

5. **Mode register :**

i. Each DMA channel has 8 bit mode registers. This is written by CPU in program mode. Bits 0 and 1 of mode register determine which of the 4 channel mode register is to be written.

ii. The bits 2 and 3 indicate type of DMA transfer. Bit 4 of mode register indicates whether auto initialization is selected or not.

iii. While bit 5 indicates whether address increment or decrement mode is selected.

6. **Request Register :**

i. Each channel has a request associated with it, in the request register.

ii. These are non-maskable and subject to prioritization by the priority resolving network of 8237.

7. **Mask Register :** Each of four channel of DMA has mask bit which can be set under program control to disable the incoming DREQ requests at the specific channel.

8. **Temporary register :** The temporary register holds data during memory to memory data transfers.

9. **Status Register :** This register keeps the track of all the DMA channel pending request and the status of their terminal counts.

Que 3.2. **Describe the pin configuration of 8237 DMA controller.**

Answer

Pin diagram : Pin diagram of 8237 DMA controller is shown in Fig. 3.2.1. The functional signal description in brief as follows :

1. **V_{CC} :** This is + 5 V supply pin required for operation of circuit.
2. **GND :** This is return line for the supply (ground pin of IC).
3. **CLK :** This is the internally required clock signal for deriving the internal timings required for the circuit operation.
4. **$\overline{CS}$:** This is an active-low chip select the input of IC.
5. **RESET :** A high on this input line clears the command, status, request and temporary register. It also clear internal first/last flip-flop and set the master register.
6. **READY :** This active high input is used to match the read or write speed of 8237 with slow memories or input/output devices.
7. **Address bus (A_0 – A_3 and A_4 – A_7) :** The four least significant lines A_0 – A_3 are bi-directional three state signal. In idle cycle, they are input and used by CPU to address the control register to be read or written. The four most significant address lines (A_4 – A_7) are activated only during DMA services to generate the respective address bits.

Scanned by TapScanner

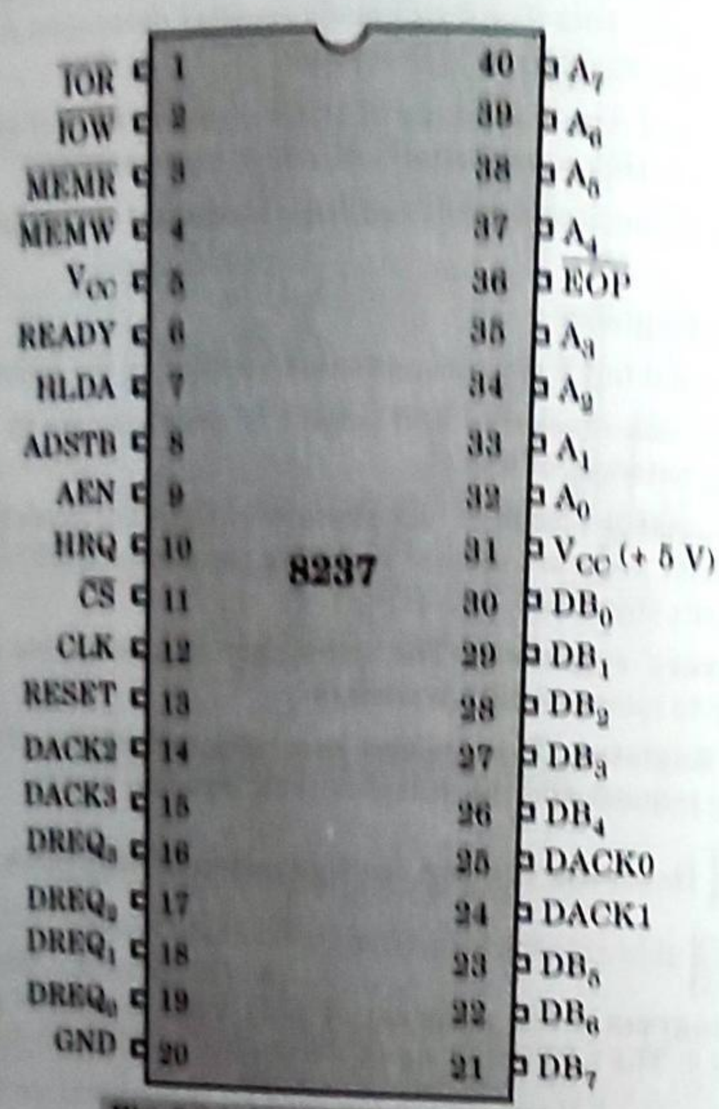


Fig. 3.2.1. Pin diagram of 8237.

8. **HRQ (Hold request) :** The HRQ is an output pin used to request the control of the system bus from the CPU. If the corresponding mask bit is not set, every valid DREQ to 8237 will issue HRQ signal to the CPU.
9. **DB_0 – DB_7 (Data bus) :** These are bi-directional lines used to transfer data to/from memory or input/output.
10. **IOR and IOW (I/O read and I/O write) :** These are active low bi-directional symbol. In idle cycle, these are input control signals used by CPU to read/write the control registers. In the active cycle IOR signal is used to access the data from a peripheral and IOW signal is used to load data to the peripheral.
11. **$DACK_0$ – $DACK_3$ (DMA acknowledge) :** These are used to indicate peripheral devices that the DMA request is granted.

12. **AEN (Address enable) :** This active-high output enables the 8-bit latch that drives the upper 8-bit address bus. The AEN pin is used to disable other bus drivers during DMA transfers.
13. **ADSTB (Address strobe) :** This output line is used to strobe the upper address byte generated by 8237, in master mode into an external latch.
14. **$DREQ_0$ – $DREQ_3$:** These are used to indicate peripheral devices that the DMA request is granted.
15. **MEMR :** This active-low output is used to access data from the selected memory location, during DMA read or a memory to memory transfer.
16. **MEMW :** This active low output signal is used to write data to the selected memory location during DMA write or memory to memory transfer.
17. **EOP (End of Process) :** This is an active low bi-directional (input or output) pin, used to indicate the completion of DMA operation.

Que 3.3. **What is meant by direct memory access ? With the help of neat schematic, explain the interaction between 8086 MPU and DMA controller for providing direct memory access to a peripheral device.**

AKTU 2018-19, Marks 07

Answer

A. Direct Memory Access : Refer Q. 3.1, Page 3–2C, Unit-3.

B. Interfacing between 8086 and DMA controller :

1. The 8237 works in two modes, *i.e.*, master and slave modes. In slave mode, the 8237 functions as an input/output device.
2. In this mode the system buses are controlled by microprocessor and hence the microprocessor is connected to the system bus.
3. In master mode 8237 becomes the bus master and hence the microprocessor is isolated from the system bus. This isolation is done by AEN Signal.
4. In minimum configuration, 8237 DMA controller is used to transfer the data. The peripheral chips are interface as normal IO ports.
5. Fig. 3.3.1 shows the interfacing of DMA controller with 8086.
6. In minimum mode the HOLD and HLDA signals are used to bus arbitration and in maximum mode configuration $\overline{RQ}_0/GT_0$ and $\overline{RQ}_1/GT_1$.

Scanned by TapScanner

7. The 8237 outputs only 16-bit memory address but not the complete 20-bit address of 8086.
8. To store the remaining four address bits $A_{16} - A_{19}$ a separate page latch is required.
9. The 8237 is not compatible with 8086 in its maximum mode configuration. This is because *RQ /GT* is not compatible with *HRQ* and *HLDA* of 8237.

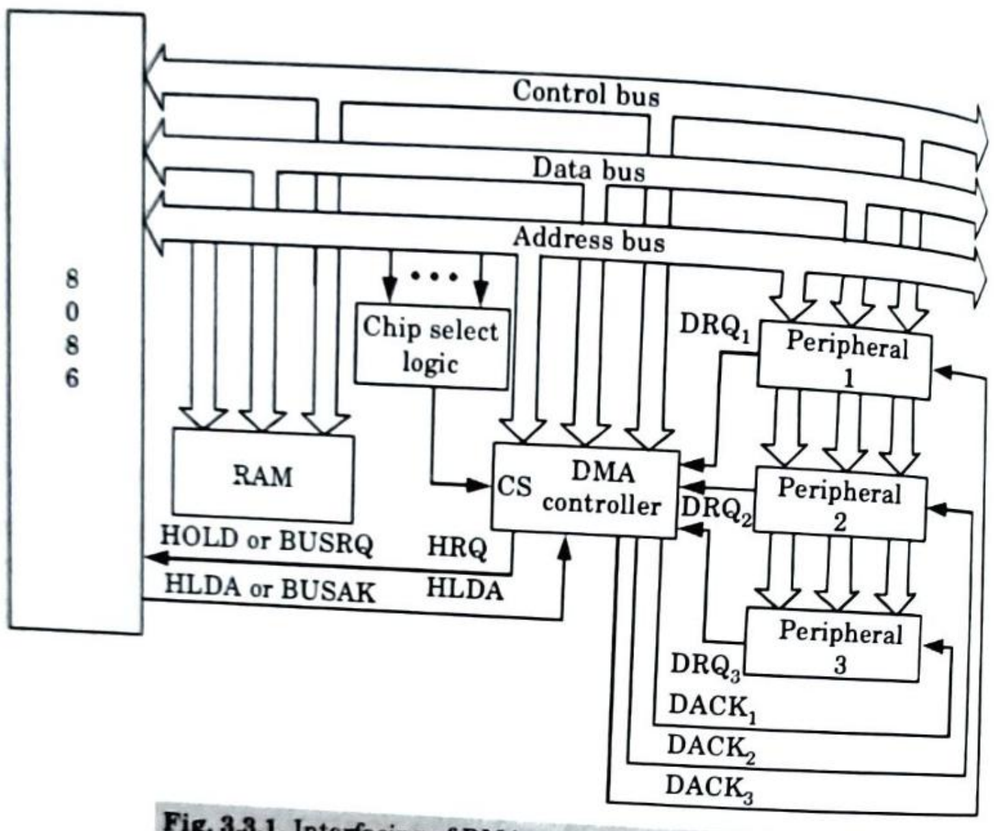


Fig. 3.3.1. Interfacing of DMA controller with processor.

Que 3.4. **With a neat diagram discuss internal architecture of 8255.**

AKTU 2015-16, Marks 05

OR

Explain the internal architecture of 8255.

AKTU 2017-18, Marks 10

Answer

A. Internal architecture of 8255 :

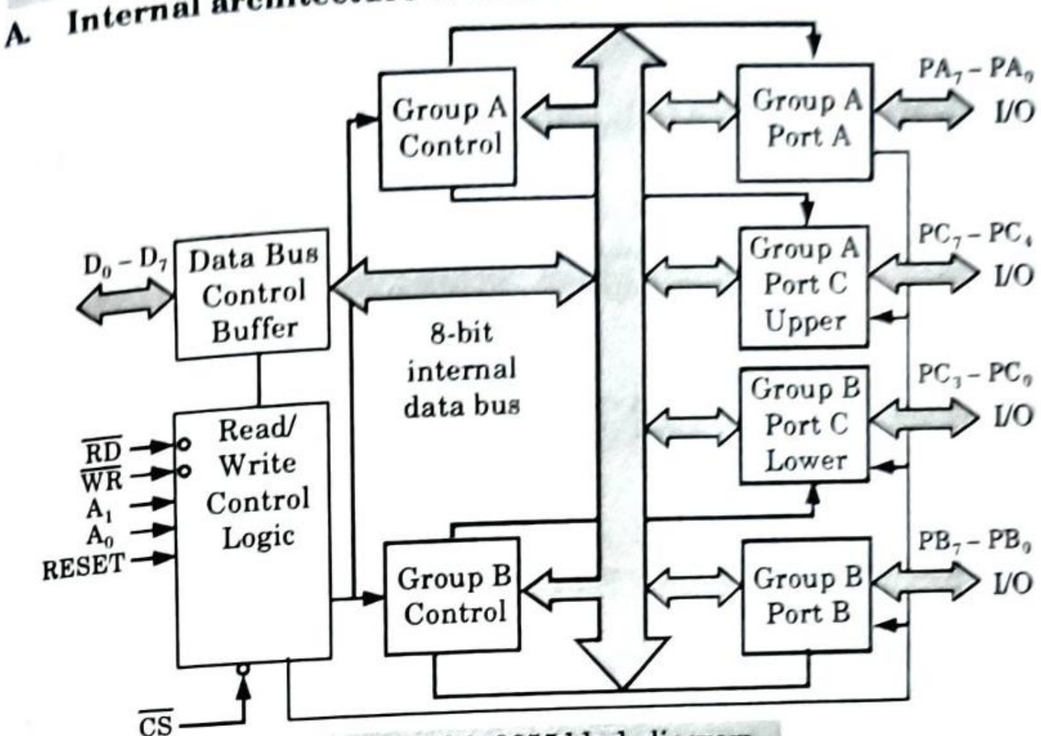


Fig. 3.4.1. 8255 block diagram.

B. Functional blocks :

a. Data bus control buffer :

1. This tri-stated bi-directional buffer is used to interface the 8255 to the system data bus.
2. IN or OUT instructions executed by the CPU either read data from, or write data into the buffer.
3. Output data from the CPU to the ports or control register, and input data to the CPU from the ports or status register are all passed through the buffer.

b. Read/Write control logic :

1. The control logic block accepts control bus signals as well as inputs from the address bus, and issues commands to the individual group control blocks.
2. It issues appropriate enabling signals to access the required data/control words or status word. $\overline{RD}$, $\overline{WR}$, RESET, and $\overline{CS}$ are 4-control signals generated by control section.

c. Group A and Group B controls :

1. Each of the group A and group B control blocks receives control words from the CPU through the data bus buffer and the internal data bus accepts commands from the control logic block and issues appropriate commands to the ports associated with it.

Scanned by TapScanner

2. The group A control block controls Port A and $PC_7 - PC_4$, while the group B control block controls Port B and $PC_3 - PC_0$.

d. Ports :

1. **Port A :** This has an 8-bit latched and buffered output and 8-bit input latch.
2. **Port B :** This has an 8-bit input/output latch/buffer and an 8-bit data input buffer.
3. **Port C :** This has one 8-bit unlatched input buffer and an 8-bit output latch/buffer. Port C can be splitted into two parts and each can be used for control signal outputs/inputs for Port A and Port B in the handshake mode.

Que 3.5. Describe the pin configuration of 8255.

Answer

1. **$PA_7 - PA_0$:** These are eight port A lines that act as either latched output or buffered input lines depending upon the control word loaded into the control word register.
2. **$PC_7 - PC_4$:** Upper nibble of port C lines. They may act as either output latches or input buffer lines. This port can also be used for generation of handshake lines in mode 1 or mode 2.
3. **$PC_3 - PC_0$:** These are the lower port C lines.
4. **$PB_0 - PB_7$:** These are the eight port B lines which are used as latched output lines or buffered input lines in the same way as port A.
5. **$\overline{RD}$:** This is the input line driven by the microprocessor and should be low to indicate read operation to 8255.
6. **$\overline{WR}$:** This is an input line driven by the microprocessor. A low on this line indicates write operation.
7. **$\overline{CS}$:** This is a chip select line. If this line goes low, it enables the 8255 to respond $\overline{RD}$ and $\overline{WR}$ signals, otherwise $\overline{RD}$ and $\overline{WR}$ signals are neglected.
8. **$A_1 - A_0$:** These are the address input lines and are driven by microprocessor.
9. **$D_0 - D_7$:** These are the data bus lines those carry data or control word to/from the microprocessor.
10. **RESET :** A logic high on this line clears the control word register of 8255. All ports are set as input ports by default after reset.

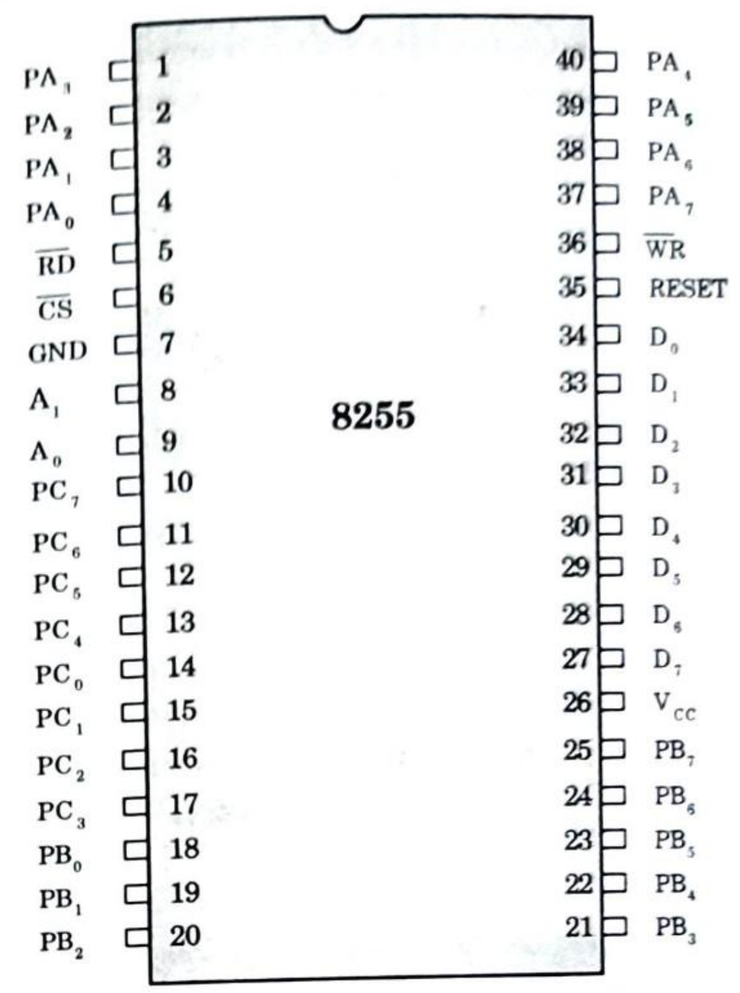


Fig. 3.5.1. 8255 Pin configuration.

Que 3.6. Explain the modes of operation of 8255.

Answer

The 8255 has following modes of operation :

i. Mode 0 (Basic input/output) :

1. In this mode, in addition to Port A and Port B, $PC_7 - PC_4$ and $PC_3 - PC_0$ of Port C can be considered as two individual 4-bit ports.
2. Therefore, four ports, each of which can be configured either as an input or an output port are available.
3. Here the ports are simple input or output ports; data is written to or read from the specified port without handshaking.
4. The data send out to the output ports are latched, whereas inputs are not latched.

ii. Mode 1 (Strobe input/output) :

1. In this mode, input or output data transfer is affected by strobe (handshaking signals).

Scanned by TapScanner

2. The two groups, Group A and Group B, can be configured separately, with each group consisting of an 8-bit port and a 4-bit port.
3. The 8-bit port can be programmed for input or output. The 4-bit port is used for handshaking.

iii. Mode 2 (Strobe bi-directional bus I/O) :

1. This mode allows bi-directional data transfer over a single 8-bit data bus using handshaking signals.
2. This feature is available only in Group A with port A as the 8-bit bi-directional data bus, and $PC_4 - PC_7$ are used for handshaking purposes.
3. In this mode, both input and output are latched.

iv. Bit Set-Reset (BSR) Mode :

1. In this mode, any of the 8-bits of port C can be set or reset by sending an OUT instruction to the control register.
2. When port C is used for control/status operation, this feature can be used to set or reset individual bits as if they were output ports.

Que 3.7. **Draw and explain the internal architecture of 8255 parallel I/O peripheral device. Also describe the bits of control word.**

AKTU 2016-17, Marks 10

Answer

A. **Internal architecture 8255 :** Refer Q. 3.4, Page 3–7C, Unit-3.

B. **Bits of control word :**

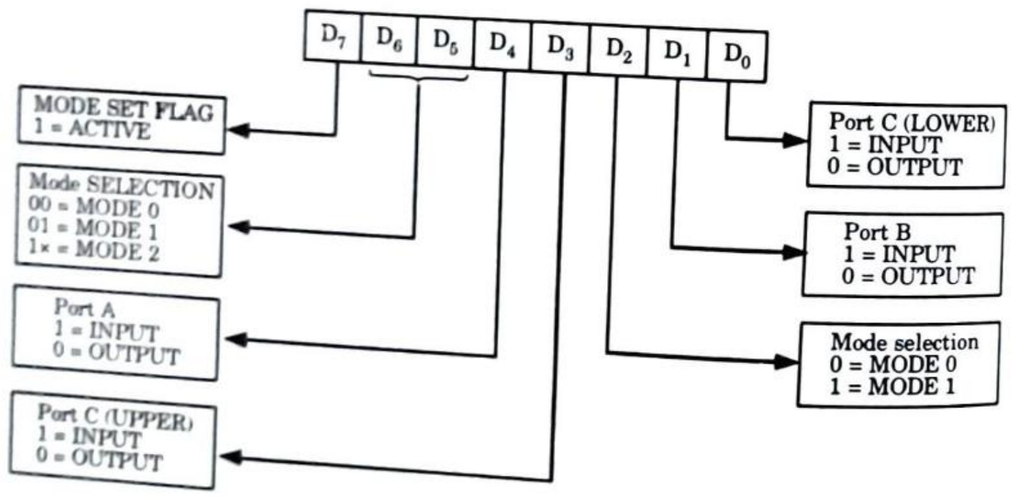


Fig. 3.7.1.

Que 3.8. **Form a control word to set the D_3 bit of port-C of the 8255. Write a program that drives 8 numbers of LEDs sequentially using any one of the ports of PPI.**

AKTU 2018-19, Marks 07

Answer

A. **Control word :** Refer Q. 3.7, Page 3–11C, Unit-3.

B. **Program :**

```
        MVI A, 80H    ; Load accumulator with control word 80H.
        OUT 03H       ; Shift ((A)) to CWR whose address is 03H.
        MVI A, 01H    ; Load accumulator with 01H to make its
                      ; 0th bit high.
MOON :  OUT 02H       ; Shift ((A)) to Port-C.
        CALL DELAY    ; Call delay sub-routine.
        RLC           ; Move 1 from right to left starting from
                      ; 0th bit.
        JMP MOON      ; Jump to label MOON to repeat.
```

Que 3.9. **Write a program to initialize 8255 as follows :**

Port A : Simple input port

Port B : Simple output port

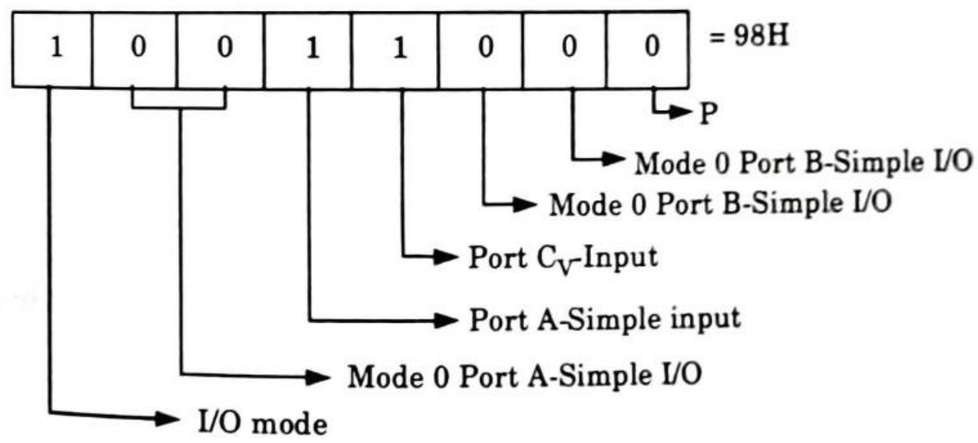
Port C_L : Output port

Port C_v : Input port

Assume the address of control register is 03H.

AKTU 2015-16, Marks 05

Answer



Program :

```
MVI A, 98H
OUT 03H
```

Scanned by TapScanner

Que 3.10. **Interface an 8255 with 8086 to work as an I/O port. Initialize port A as output port, port B as input port and port C as output port. The address of port A, port B, port C and CWR is 0760, 0762, 0764 and 0766 respectively. Write a program to sense switch positions (10110101) connected at port B. The sensed pattern is to be displayed on port A, to which 8 LEDs are connected, while port C lower displays number of on switches out of the total eight switches.**

AKTU 2016-17, Marks 10

Answer

i. The control word is given as follows :

B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	Control word
1	0	0	0	0	0	1	0	= 82H
I/O mode	Port A in mode 0		Port A, O/P	Port C, O/P	Port B, mode 0	Port B, I/P	Port C, O/P	

82H is the control word.

ii. The control word format for BSR mode is given as follows :

8255 ports	I/O Address lines																Hex port address
	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
Port A	0	0	0	0	0	1	1	1	0	1	1	0	0	0	0	0	0760H
Port B	0	0	0	0	0	1	1	1	0	1	1	0	0	0	1	0	0762H
Port C	0	0	0	0	0	1	1	1	0	1	1	0	0	1	0	0	0764H
CWR	0	0	0	0	0	1	1	1	0	1	1	0	0	1	1	0	0766H

iii. The assembly language program :

```
          MOV DX, 0766H  :  Initialise CWR
          MOV AL, 82H    :  Control word 82H
          OUT DX, AL
          SUB DX, 04     :  Get address of port B in DX
          IN AL, DX      :  Read port B for switch
          SUB DX, 22     :  Positions into AL and get port A address
          MOV BL, 00H    :  Initialise BL for switch count
          MOV CH, 08H    :  Initialise CH for total switch number
Label 1 :  ROL AL        :  Rotate AL through carry to check
          JNC label 2    :  Jump to label 2 if carry is not zero
          INC BL         :  Increment BL
Label 2 :  DEC CH        :  Decrement CH
          JNZ label 1    :  Jump to label 1 if CH is not zero
```



```
          MOV AL, BL
          ADD DX, OH
          OUT DX, AL
          HLT
```

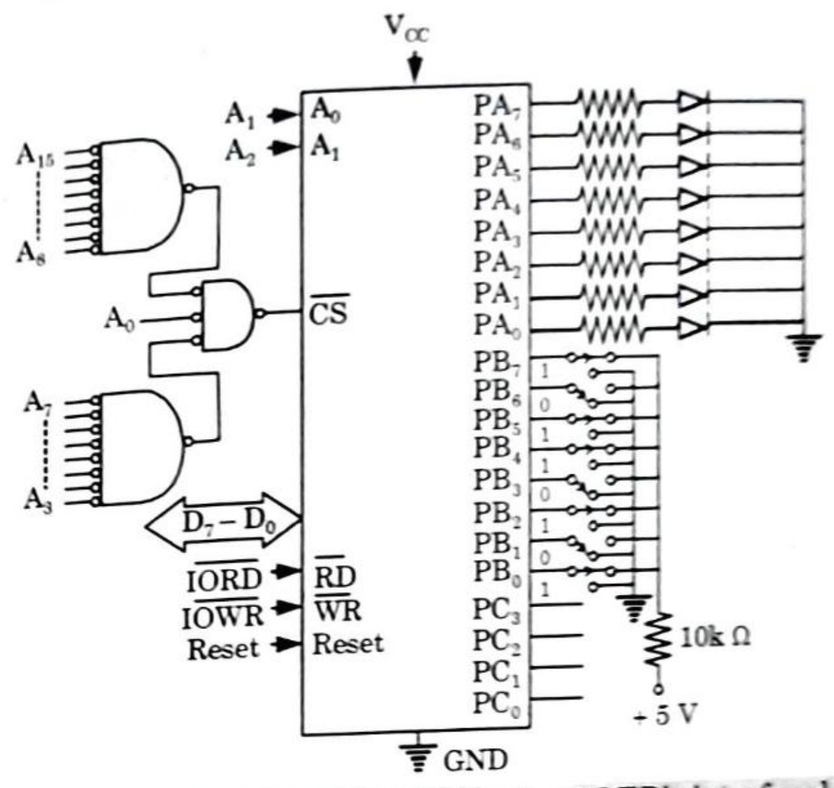


Fig. 3.10.1. Interfacing circuit showing 8-LED's interfaced.

1. The 8255 is to be interfaced with lower order data bus, *i.e.*, D_0-D_7.
2. The A_0 and A_1 pins of 8255 are connected to A_1 and A_2 pins of the microprocessor respectively.
3. For deriving the device address pulse, out of $A_0 - A_{15}$ lines, two address lines A_2 and A_1 are directly required by 8255 for three port and CWR address decoding. Hence only A_3 to A_{15} are used for decoding addresses.

Que 3.11. **Explain 8253/8254 programmable timer/counter.**

Answer

8253/8254 programmable timer/counter :

1. Delay routines cannot give time delay precisely. 8253 facilitates the generation of accurate time delays.
2. Also when 8253 is used as a timing and delay generation peripheral, the microprocessor becomes free from the tasks related to counting and can execute the programs in memory. This minimizes the software overhead on the microprocessor.
3. The 8254 is an upgraded version of the 8253, and they are pin compatible. Features of the two devices are almost identical except that

Scanned by TapScanner

i. The 8254 can operate on higher frequency than the 8253.

ii. The 8254 includes a status read-back command that can latch the count and the status of the counter.

Architecture and signal descriptions :

1. The internal block diagram of 8253 is shown in Fig. 3.11.1. The programmable timer device 8253 contains three independent 16-bit counters. It is thus possible to generate three independent delays or three independent counters simultaneously.
2. The three counters are controlled by programming the three internal command word registers.
3. The 8-bit, bi-directional data buffer interfaces internal circuit of 8253 to microprocessor systems bus.
4. Data is transmitted or received by the buffer upon the execution of IN or OUT instruction. The IN instruction reads data while OUT instruction writes data to a peripheral.

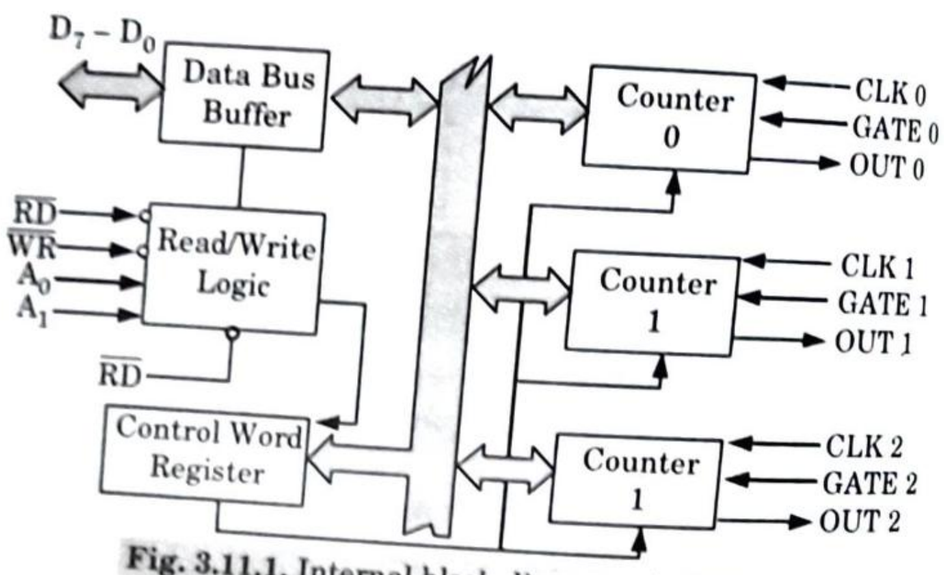


Fig. 3.11.1. Internal block diagram of 8253.

5. The three counters available in 8253 are independent of each other in operation, but they are identical to each other in organization.
6. The specialty of the 8253 counters is that they can be easily read on line without disturbing the clock input to the counter.
7. A_0, A_1 pins are address input pins and are required internally for addressing the control word registers and the three counter registers.
8. A low on $\overline{CS}$ line enables the 8253. No operation will be performed by 8253 till it is enabled.

B. Various operations for control inputs of 8253 :

Table 3.11.1.

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A_1	A_0	Selected Operation
0	1	0	0	0	Write Counter 0
0	1	0	0	1	Write Counter 1
0	1	0	1	0	Write Counter 2
0	1	0	1	1	Write Control Word
0	0	1	0	0	Read Counter 0
0	0	1	0	1	Read Counter 1
0	0	1	1	0	Read Counter 2
0	0	1	1	1	No Operation (tri-stated)
0	1	1	×	×	No Operation (tri-stated)
1	×	×	×	×	Disabled (tri-stated)

A control word register accepts the 8-bit control word written by the microprocessor and stores it for controlling the complete operation of the specific counter.

Que 3.12. **What is 8253/8254 programmable interval timer, draw and explain its internal architecture.** **AKTU 2015-16, Marks 7.5**

OR

Draw and explain block diagram and pin configuration of IC-8253. **AKTU 2017-18, Marks 10**

Answer

A. **8253/8254 programmable interval timer :** Refer Q. 3.11, Page 3–14C, Unit-3.

B. **Block diagram of 8253/8254 :** Refer Q. 3.11, Page 3–14C, Unit-3.

C. **Pin configuration of IC 8253 :**

1. **V_{cc} and GND :** Power supply and ground pins. 8253 uses + 5 V power supply.
2. **D_{7-0}:** Eight bi-directional data pins for communication with the processor.
3. **$\overline{RD}$:** Active low input pin that is activated by the processor to read counter information from the 8253.
4. **$\overline{WR}$:** Active low input pin that is activated by the processor to write control information to the 8253.

Scanned by TapScanner

5. $\overline{CS}$: Active low input pin used for selecting the chip.

6. A_1, A_0: Address input pins. They are used along with $\overline{RD}, \overline{WR}$, and $\overline{CS}$ to select one of the counters or the control port. The control port can be written only from the processor. The processor cannot read it. The counters can be read or written.

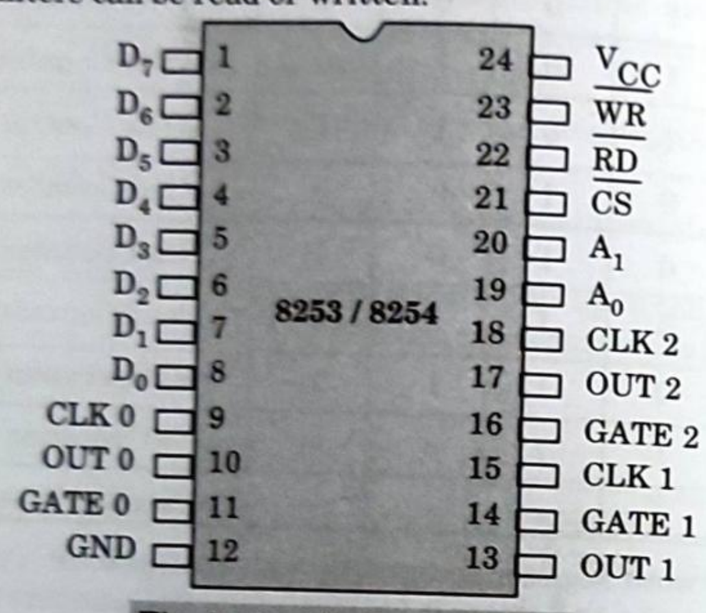


Fig. 3.12.1. Pin Configuration.

Table 3.12.1. Selection of a counter port or control port

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	Operation
0	0	0	1	0	Read Counter 0
0	1	0	1	0	Read Counter 1
1	0	0	1	0	Read Counter2
1	1	0	1	0	No operation
0	0	1	0	0	Write to Counter 0
0	1	1	0	0	Write to Counter 1
1	0	1	0	0	Write to Counter 2
1	1	1	0	0	Write to control port
X	X	X	X	1	No operation
X	X	1	1	0	No operation

7. CLK_0: Provides clock input for Counter 0.
8. CLK_1: Provides clock input for Counter 1.
9. CLK_2: Provides clock input for Counter 2. Maximum frequency allowed for any of these clocks is 2 MHz.
10. $Gate_0$: It is an input pin that controls the function of Counter 0.

11. $Gate_1$: It is an input pin that controls the function of Counter 1.
12. $Gate_2$: It is an input pin that controls the function of Counter 2. The gate input has different effects in different modes on the functioning of the corresponding counters.
13. Out_0: It is an output pin on which Counter 1 sends its output.
14. Out_1: It is an output pin on which Counter 1 sends its output.
15. Out_2: It is an output pin on which Counter 2 sends its output. The output generated by the timer depends on the mode of operation.

Que 3.13. **Discuss the mode of operation of 8253 program, internal times with its control format.** **AKTU 2016-17, Marks 10**

OR

Explain how 8253/8254 can be used as a square wave generator. **AKTU 2015-16, Marks 7.5**

Answer

1. **Mode 0 (Interrupt on terminal count) :**

a. In this mode, initially the OUT is low.

b. Once a count is loaded in the register, the counter is decremented every cycle and when the count reaches zero, the OUT goes high.

c. This can be used as an interrupt. The OUT remains high until a new count or a command word is loaded.

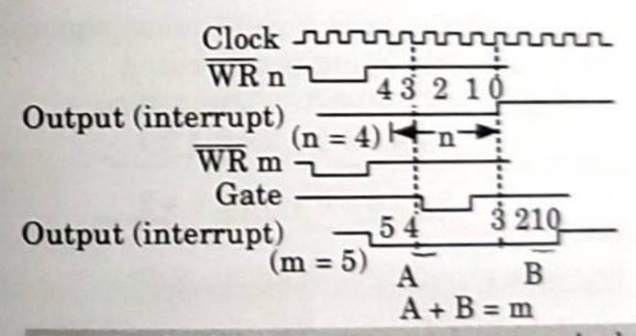


Fig. 3.13.1. Mode 0 : Interrupt on terminal count.

2. **Mode 1 (Hardware-retriggerable one-shot) :**

a. In this mode, the OUT is initially high.

b. When the Gate is triggered, the OUT goes low, and at the end of the count, the OUT goes high again, thus generating a one-shot pulse.

Scanned by TapScanner

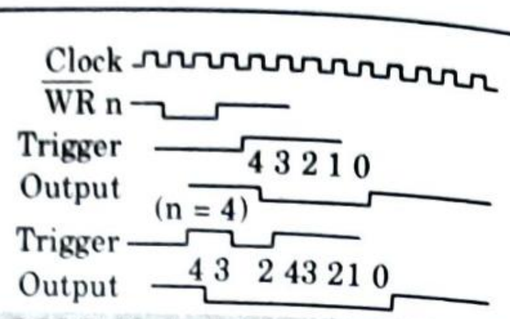


Fig. 3.13.2. Mode 1 : Programmable one-shot.

3. Mode 2 (Rate generator) :

a. This mode is used to generate a pulse equal to the clock period at a given interval.

b. When a count is loaded, the OUT stays high until the count reaches 1, and then the OUT goes low for one clock period.

c. The count is reloaded automatically and the pulse is generated continuously.

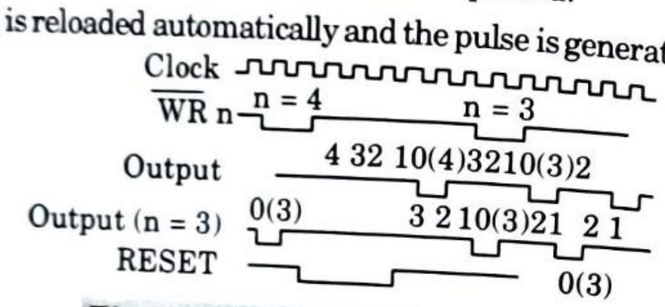


Fig. 3.13.3. Mode 2 : Generator clock.

4. Mode 3 (Square wave generator) :

a. In this mode, when a count is loaded, the OUT is high.

b. The count is decremented by two at every clock cycle and when it reaches zero, the OUT goes low and the count is reloaded again.

c. This is repeated continuously; thus a continuous square wave with period equal to the period of the count is generated.

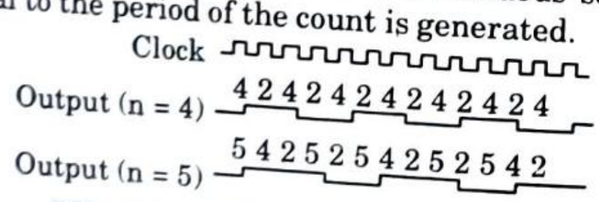


Fig. 3.13.4. Square wave generator.

5. Mode 4 (Software-triggered strobe) :

a. In this mode, the OUT is initially high; it goes low for one clock period at the end of the count.

b. The count must be reloaded for subsequent outputs.

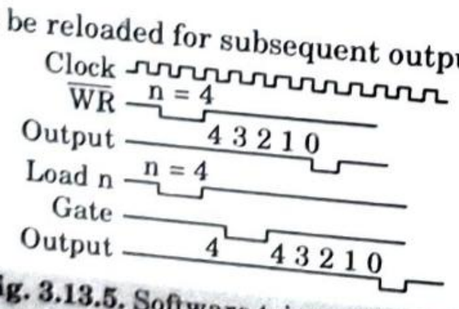


Fig. 3.13.5. Software triggered strobe.

6. Mode 5 (Hardware-triggered strobe) :

a. This mode is similar to Mode 4, except that it is triggered by the rising pulse at the gate.

b. Initially, the OUT is low, and when the Gate pulse is triggered from low to high the count begins.

c. At the end of the count, the OUT goes low for one clock period.

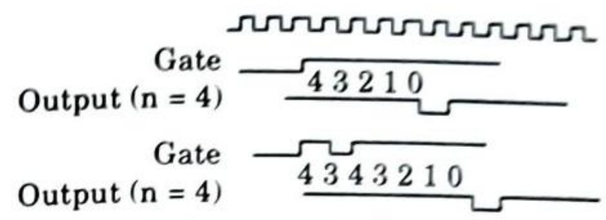


Fig. 3.13.6. Hardware triggered strobe.

PART-2

8259 Programmable Interrupt Controller, 8251 USART and RS232C.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.14. **Explain the 8259 Programmable interrupt controller.**

AKTU 2017-18, Marks 05

Answer

8259 :

1. The 8259A is packed in 28-pin plastic DIP using nMOS technology and requires only one power supply of +5 V.
2. It does not require any clock input.
3. It provides the facility of 8-vectored priority interrupts for the μP on a single chip.
4. The 8259 A has several modes of operation.
5. The Intel's 8259 A is a Programmable Priority Interrupt Controller (PIC).
6. It accepts interrupt requests from 8-I/O devices, determines its priority, and priority, and issues interrupt to the μP.
7. After receiving the acknowledgement from the μP, the 8259 places a CALL instruction along with the associated address of the current interrupt on the data bus.

Scanned by TapScanner

8. Without any additional circuitry nine numbers of the 8259A can be cascaded to handle up to 64-vectored priority interrupts.
9. It accept requests from the peripheral devices, determines the highest priority (importance) of the device.

Que 3.15. Explain pin diagram of 8259 A (PIC).

Answer

1. **$\overline{WR}$:** This pin is an active low write enable input to 8259A. This enables it to accept command words from CPU.
2. **$\overline{RD}$:** This is an active-low read enable input to 8259A. A low on this line enables 8259A to release status onto the data bus of CPU.
3. **D_7-D_0 :** These pins form a bidirectional data bus that carries 8-bit data either to control word or from status word registers. It also carries interrupt vector information.
4. **INT :** This pin goes high whenever a valid interrupt request is asserted. This is used to interrupt the CPU and is connected to the interrupt input of CPU.
5. **IR_0-IR_7 (Interrupt requests) :** These pins act as inputs to accept interrupt requests to the CPU. In the edge triggered mode, an interrupt service is requested by raising an IR pin from a low to a high state. It is held high until it is acknowledged, and just by latching it to high level, if used in the level triggered mode.
6. **$\overline{INTA}$ (Interrupt acknowledge) :** This pin is an input used to strobe-in 8259A interrupt vector data on to the data bus. In conjunction with CS, WR, and RD pins, this selects the different operations like, writing command words, reading status word, etc.
7. **CAS_2–CAS_0 (I/O Cascade Lines) :** The CAS lines form a private 8259A bus to control a multiple 8259A structure. These pins are outputs for a master 82C59A and inputs for a slave 82C59A.
8. **$\overline{SP}$ / $\overline{EN}$ (I/O Slave Program/Enable Buffer) :** This is a dual function pin. In the buffered mode, it can be used as an output to control buffer transceivers ($\overline{EN}$). When not in the buffered mode, it is used as an input to designate a master ($\overline{SP}$ = 1) or slave ($\overline{SP}$ = 0). A HIGH on the ($\overline{SP}$ = $\overline{EN}$ = 1) pin designates the 8259 as the master, a LOW on ($\overline{SP}$ = $\overline{EN}$ = 0) designates it as slave.

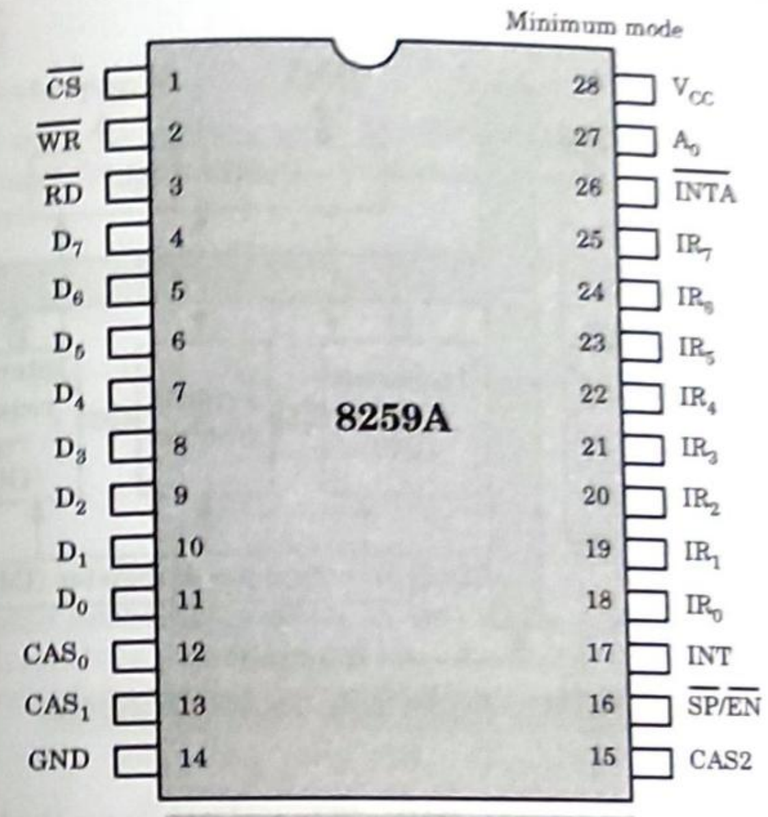


Fig. 3.15.1. 8259A pin diagram.

Que 3.16. Explain the architecture of 8259A.

Answer

The architectural block diagram of 8259A is shown in Fig. 3.16.1. The functional explanation of each block is given as follows :

i. **Interrupt request register (IRR) :** The interrupts at IRQ input lines are handled by interrupt request register internally. IRR stores all the interrupt requests in it in order to serve them one by one on the priority basis.

ii. **In-service register (ISR) :** This store all the interrupt requests those are being served, *i.e.*, ISR keeps a track of the request being served.

iii. **Priority resolver :**

1. This unit determines the priorities of the interrupt request appearing simultaneously.
2. The highest priority is selected and stored into the corresponding bit of ISR during $\overline{INTA}$ pulse.
3. The IR_0 has the highest priority while the IR_7 has the lowest one, normally in fixed priority mode.

Scanned by TapScanner

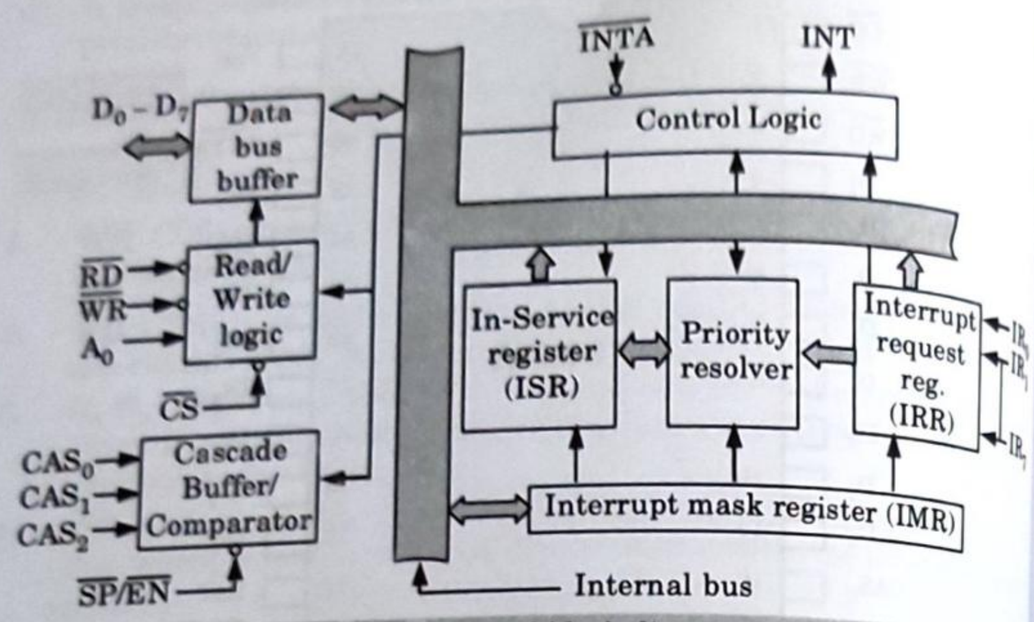


Fig. 3.16.1. Functional block diagram of 8259A.

iv. **Interrupt mask register (IMR) :** This register stores the bit required to mask the interrupt inputs. IMR operates on IRR at the direction of the priority resolver.

v. **Interrupt control logic :**

1. This block manages the interrupt and interrupt acknowledge signals to be sent to the CPU for serving one of the eight interrupt requests.
2. This also accepts the interrupt acknowledge (INTA) signal from CPU that causes the 8259A to release vector address on the data bus.

vi. **Data bus buffer :**

1. This tri-state bi-directional buffer interfaces internal 8259A bus to the microprocessor system data bus.
2. Control words, status and vector information pass through data buffer during read or write operations.

vii. **Read/write control logic :** This circuit accepts and decodes commands from the CPU. This block also allows the status of the 8259A to be transferred on the data bus.

viii. **Cascade buffer/comparator :** This block stores and compares the ID's of all the 8259A's used in the system.

Que 3.17. **Explain different modes of operation of 8259 A.**

AKTU 2017-18, Marks 10

Answer

i. **Fully nested mode :**

1. This is a general-purpose mode in which all IRs (interrupt requests) are arranged from highest to lowest, with IR_0 as the highest and IR_7 as the lowest.
2. In the example below, IR_4 has the highest priority and IR_3 has the lowest priority :

IR_0	IR_1	IR_2	IR_3	IR_4	IR_5	IR_6	IR_7
4	5	6	7	0	1	2	3
			Lowest priority	Highest priority			

ii. **Automatic rotation mode :** In this mode, a device, after being serviced, receives the lowest priority. Assuming that the IR_2 has just been serviced, it will receive the seventh priority, as shown below :

IR_0	IR_1	IR_2	IR_3	IR_4	IR_5	IR_6	IR_7
1	6	7	0	1	2	3	4

iii. **Specific rotation mode :** This mode is similar to the automatic rotation mode, except that the user can select any IR for the lowest priority, thus fixing all other priorities.

Que 3.18. **Draw and explain the internal architecture of 8259 interrupt controller. Also describe its initialization command words.**

AKTU 2016-17, Marks 10

Answer

A. **Internal architecture of 8259 (8259 A) :** Refer Q. 3.16, Page 3–22C, Unit-3.

B. **Initialization Command Words (ICWs) :**

1. It is used to program the basic operation of 8259A.
2. Before it starts functioning, the 8259A must be initialized by writing two to four command words into the respective command word registers. These are called as initialization command words (ICWs).
3. **Types :**

i. **Initialization Command Word 1 (ICW 1) :**

1. A write command issue to the 8259A with $A_0 = 0$ and $D_4 = 1$, is recognized as ICW 1.
2. During ICW 1, the following steps occurs :

Scanned by TapScanner

a. It reset edge sense circuit. This implies that after initialization, an interrupt must make a low to high transition.

b. The interrupt mask register is cleared.

c. Lowest priority is assigned to IR_7.

d. The slave mode address is set to 7 (1, 1, 1).

e. It clears special mode and sets the status read to IRR.

f. If bit D_0 in ICW 1 is set to '0', then it clears all functions associated with ICW 4.

ii. Initialization Command Word 2 (ICW 2) :

1. A write command issue after ICW 1 with $A_0 = 1$, is recognized as ICW 2.
2. The ICW 2 format is shown in Fig. 3.18.1.

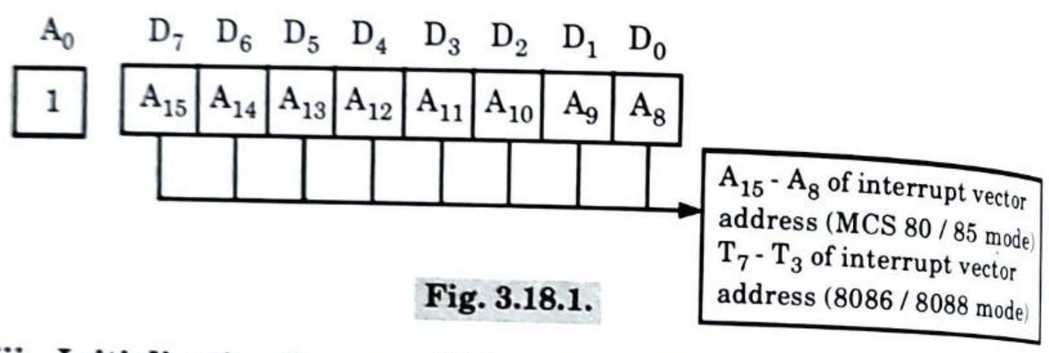


Fig. 3.18.1.

iii. Initialization Command Word 3 (ICW 3) :

1. If there is more than one 8259A in the system and if they are cascaded then ICW 3 is required.
2. There are two types of ICW 3 : Master ICW 3 and Slave ICW 3.
3. The master ICW 3 is used to specify whether it has a slave 8259 connected to its interrupt request input. The slave ICW 3 is used to assign a slave identification number.

iv. Initialization Command Word 4 (ICW 4) :

1. The ICW 4 is loaded only if the D_0 bit of ICW 1 is set.
2. The ICW 4 is used to initialize the 8259 in the following modes :
   a. Special Fully Nested Mode (SFNM)
   b. Buffered Mode (BUF)
   c. Auto EOI Mode (AEOI)
   d. 8086/8088 Mode

Que 3.19. **How many 8259 can be interconnected in cascade mode? Show their cascading structure.** **AKTU 2018-19, Marks 07**

Answer

1. Upto eight 8259s may be cascaded together to act as slave unit to a master 8259.
2. Thus, a total of 64 I/O devices may be connected to the μP via the 8259 to transfer data in interrupt driven mode.
3. Fig. 3.19.1 exhibits a system having two 8259s – one master and one salve used for interfacing 15 I/O devices to 8085.

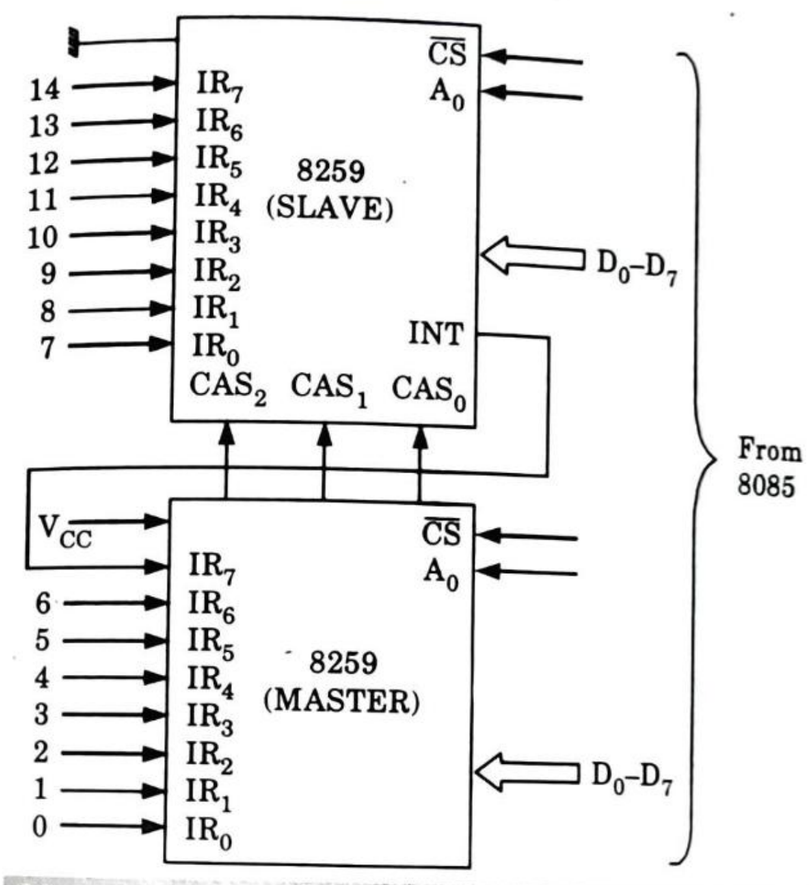


Fig. 3.19.1. Using the 8259 PICs to interface 15 I/O device to 8085 each having a unique ISS address.

4. The INT output of the slave is connected to the IR_7 input of the master.
5. If any of the seven devices 0-6 interrupts, the sequence of actions taken by the master.
6. When any of devices 7-14 interrupts, the slave will issue an interrupt to the μP if no other higher priority interrupt is pending.
7. On receipt of the INTA pulse, the master will send out the first byte of the CALL instruction. It will enable the slave by issuing a slave address code on the CAS lines.
8. The slave corresponding to this code will send the next two bytes of the CALL instruction directly to the μP in the next tow INTA pulses. This address would obviously correspond to the interrupting device.

Scanned by TapScanner

Que 3.20. **Explain the internal architecture of 8251.**

Answer

Architecture and signal descriptions of 8251 :

1. The architectural block diagram of 8251 A is shown in Fig. 3.20.1, followed by the functional description of each block.
2. The data buffer interfaces the internal bus of the circuit with the system bus.
3. The read write control logic controls the operation of the peripheral depending upon the operations initiated by the CPU. This unit also selects one of the two internal addresses those are control address and data address at the behest of the $C/\overline{D}$ signal.
4. The modem control unit handles the modem handshake signals to coordinate the communication between the modem and the USART. The transmit control unit transmits the data byte received by the data buffer from the CPU for further serial communication.
5. This decides the transmission rate which is controlled by the $\overline{TXC}$ input frequency. This unit also derives two transmitter status signals namely TXRDY and TXEMPTY. These may be used by the CPU for handshaking.

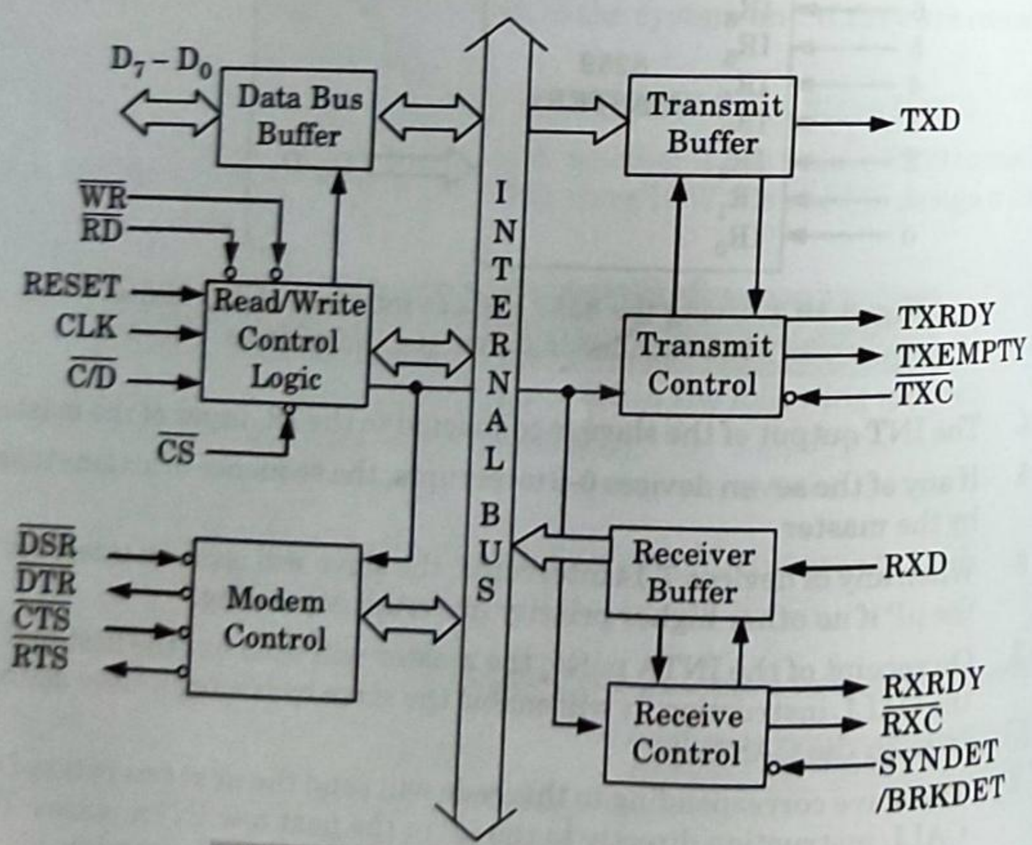


Fig. 3.20.1. 8251A Internal architecture.

6. The transmit buffer is a parallel to serial converter that receives a parallel byte for conversion into a serial signal and further transmission onto the communication channel.
7. The receive control unit decides the receiver frequency as controlled by the $\overline{RXC}$ input frequency. This unit generates a receiver ready (RXRDY) signal that may be used by the CPU for handshaking.
8. This unit also detects a break in the data string while the 8251 is in asynchronous mode. In synchronous mode, the 8251 detect SYNC characters using SYNDET/BD pin.

Que 3.21. **Describe the pin configuration of USART.**

Answer

1. **D_0-D_7 :** This is an 8-bit data bus used to read or write status, command word or data from or to the 8251A.
2. ***$C/\overline{D}$* Control Word/Data :** This input pin, together with $\overline{RD}$ and $\overline{WR}$ inputs, informs the 8251A that the word on the data bus is either a data or control word/status information. If this pin is 1, control/status is on the bus, otherwise data is on the bus.
3. ***$\overline{RD}$* :** This active-low input to 8251A is used to inform it that the CPU is reading either data or status information from its internal registers.
4. ***$\overline{WR}$* :** This active-low input to 8251A is used to inform it that the CPU is writing data or control word to 8251A.
5. ***$\overline{CS}$* :** This is an active-low chip select input of 8251A. If it is high, no read or write operation can be carried out on 8251. The data bus is tristated if this pin is high.
6. **CLK :** This input is used to generate internal device timings and is normally connected to clock generator output.
7. **RESET :** A high on this input forces the 8251A into an idle state. The device will remain idle till this input signal again goes low and a new set of control word is written into it.
8. ***$\overline{TXC}$* -Transmitter Clock Input :** This transmitter clock input controls the rate at which the character is to be transmitted.
9. **TXD-Transmitted Data Output :** This output pin carries serial stream of the transmitted data bits along with other information like start bit, stop bits and parity bit, etc.
10. ***$\overline{RXC}$* -Receiver Clock Input :** This receiver clock input pin controls the rate at which the character to be received.

Scanned by TapScanner

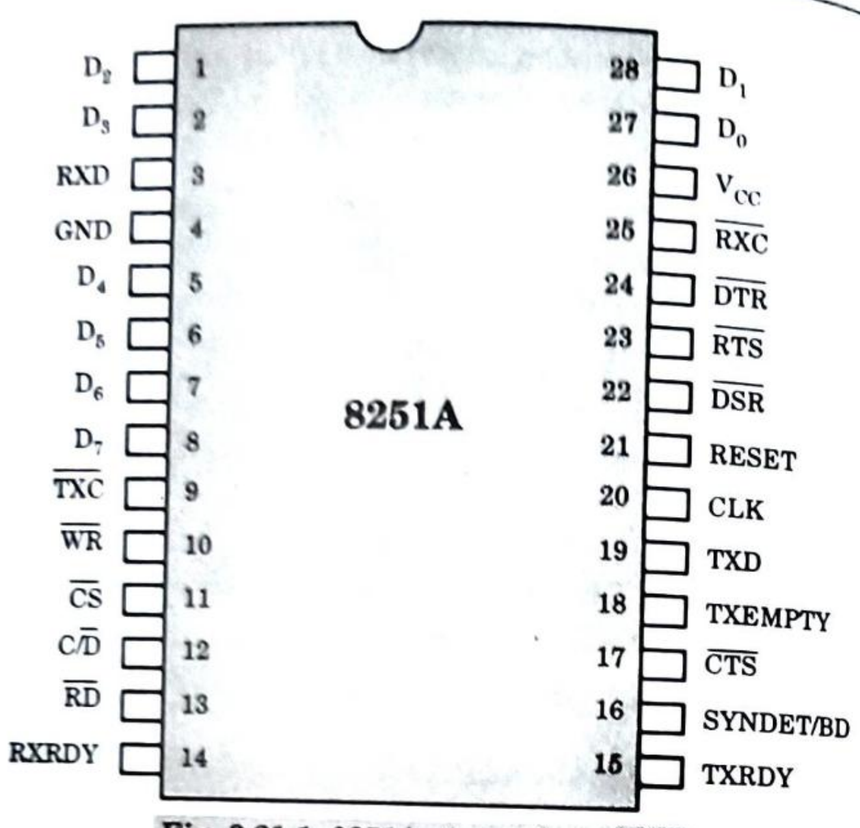


Fig. 3.21.1. 8251A pin configuration.

11. **RXD-Receive Data Input :** This input pin of 8251A receives a composite stream of the data to be received by 8251A.

12. **RXRDY-Receiver Ready Output :** This output indicates that the 8251A contains a character to be read by the CPU. The RXRDY signal may be used either to interrupt the CPU or may be polled by the CPU.

13. **TXRDY-Transmitter Ready :** This output signal indicates to the CPU that the internal circuit of the transmitter is ready to accept a new character for transmission from the CPU.

14. ***$\overline{DSR}$* -Data Set Ready :** This input may be used as a general purpose one bit inverting input port. Its status can be checked by the CPU using a status read operation. This is normally used to check if the data set is ready when communicating with a modem.

15. ***$\overline{DTR}$* -Data Terminal Ready :** This output may be used as a general purpose one bit inverting output port. This can be programmed low using the command word. This is used to indicate that the device is ready to accept data when the 8251A is communicating with a modem.

16. ***$\overline{RTS}$* -Request to Send Data :** This output also may be used as a general purpose one bit inverting output port that can be programmed low to indicate the modem that the receiver is ready to receive a data byte from the modem. This signal is used to communicate with a modem.

17. ***$\overline{CTS}$* -Clear to Send :** If the clear to send input line is low, the 8251A is enabled to transmit the serial data, provided the enable bit in the command byte is set to '1'.

18. **TXE-Transmitter Empty :** If the 8251A, while transmitting, has no characters to transmit, the TXE output goes high and it automatically goes low when a character is received from the CPU, for further transmission.

19. **SYNDET/BD-Synch Detect/Break Detect :** This pin is used in the synchronous mode for detecting SYNC characters (SYNDET) and may be used as either input or output.

Que 3.22. **Explain RS-232C.**

Answer

A. **The standards :** There are several standards that specify protocol for data transfer between the devices and the electrical characteristics of line drivers and receivers. RS-232C, RS-422A, RS-423A and RS-485 are the few standards. RS-232C is one of the most commonly used standards.

B. **The RS-232C :**

1. This standard was mainly designed to connect Data Terminal Equipment (DTE) that are sending and receiving serial data (such as a computer) and Data Communication Equipment (DCE) that are used to send data over long distances (such as a modem).

2. The driver for RS-232C converts TTL logic low into a signal of + 3 to + 25 V and TTL logic high into a signal of -3 to 25 V. The receiver converts the signals of +3 to -25 V to TTL logic low and -3 to -25 V to TTL logic high.

3. The ICs MC1488 and MC1489A are used as line drivers and receivers for RS-232C standard.

4. The MC1488 contains four drivers and each driver converts a TTL level signal to RS-232C level signal. The MC1489A contains four receivers that convert RS-232C level signals into TTL levels.

C. **Pins and signals :**

1. The standard prescribes the 25-pin connector for connecting the DTE and DCE. The rate of data transmission is restricted to a maximum of 20 k band and to a maximum distance of 50 feet.

2. Fig. 3.22.1 shows the pins and signals of RS-232C and the connection between DTE and DCE using line drivers and receivers.

i. **TXD and RXD :** The Transmit Data and Receive Data on the DTE are the serial data lines. These lines have opposite functions on a DCE.

Scanned by TapScanner

ii. $\overline{RTS}$ and $\overline{CTS}$: The Request To Send is activated by the transmitter when it wishes to send data over the line. This line is active till the end of communication. The Clear To Send is activated by the receiver to inform the transmitter whether it is ready or not ready to accept the data. It is also held active throughout the transmission.

iii. $\overline{DTR}$ and $\overline{DSR}$: The DTE informs the DCE through the Data Terminal Ready line that it is in on-line mode and communication is possible. The Data Set Ready signal is to indicate that DCE is ready for communication.

iv. $\overline{DCD}$: The Data Carrier Detect is activated by the DCE to indicate that it has established a contact with DTE. It is usually activated when RTS is granted.

v. RI : The Ring Indicator is activated by DCE when it detects an incoming call on the telephone line.

3. The following sequence of signal handshaking occurs when a computer sends a data to a modem. The computer activates the $\overline{RTS}$ signal to modem and the modem in turn activates the $\overline{DCD}$ and then $\overline{CTS}$.

4. The computer then sends data on TXD. After the data transmission is complete the computer deactivates the $\overline{RTS}$ which causes the modem to deactivate $\overline{CTS}$.

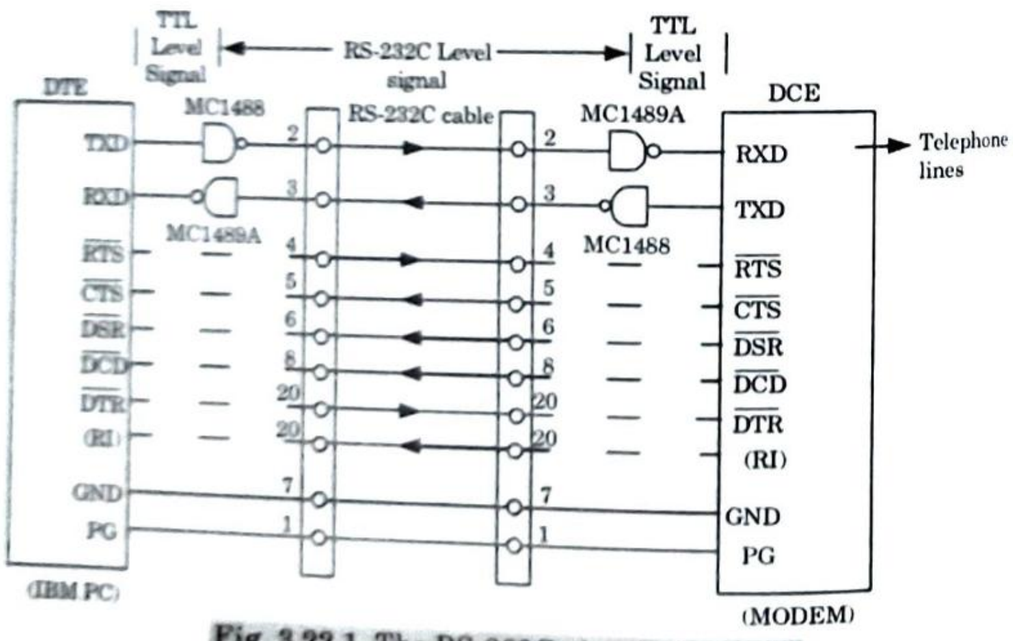


Fig. 3.22.1. The RS-232C pins and signals.

PART-3

Fundamental of Programming : Program Structure and Programming Techniques for Microprocessors, 8085 Addressing Modes, 8085 Instruction Set.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.23. **What do you mean by flow chart and algorithm ? Write an algorithm for finding out largest number from an unordered array of sixteen 8-bit numbers stored sequentially in memory.**

OR

Explain the various steps involved in writing the assembly language programs.

Answer

A. **Flowchart :** Refer Q. 1.22, Page 1–29C, Unit-1.

B. **Algorithm :**

1. An algorithm is a way of representing the procedure in steps for solving the program.
2. The representation of procedure in steps is used by the programmer for the solution of the problem.
3. Once it is developed and thoroughly checked the algorithm provides an excellence guide for writing a program.
4. **Algorithm :**

Step 1 : Initialize the counter and data pointer.

Step 2 : Move the first number in register AL.

Step 3 : Compare the next number with first number.

Step 4 : Check the condition :

if next number > first number, move to step 5

else move to step 6 and then return to step 4.

Step 5 : Replace the first number by next number.

Step 6 : Update the pointer and loop back counter time.

Step 7 : End of procedure.

C. **Steps involved :** Refer Q. 1.29, Page 1–37C, Unit-1.

Scanned by TapScanner

Que 3.24. **Write the programming techniques used in microprocessor.**

OR

Explain following :
Development tools : Editor, Library builder, Linker, Emulator.

AKTU 2017-18, Marks 05

Answer

A. **Programming techniques :** Refer Q. 1.30, Page 1–38C, Unit-1.

B. **Development tools :**

Editor :

1. Editor allows the user to enter the source code (either assembly language or some high level language) into the development system.

Library builder :

1. The library builder is used to create library files which are collections of procedures of frequently used functions.
2. A library file is a collection of assembled object files.
3. While developing software for a particular application, the programmers can link the library file in their programs.

Linker :

1. Linker is a program which is used for joining many object files into one large object file.
2. Linker does not assign absolute addresses. It only assigns relative addresses to the program starting from zero.
3. It is relocatable. The linkers with MASM or TASM produce link files with .EXE extension.

Emulator :

1. The emulator is used to test and debug the hardware and software of an external system such as the microprocessor based system.
2. Emulator is a combination of hardware and software.
3. An emulator consists of multi-wire cable that connects the host system of the external system.

Que 3.25. **Explain the addressing capability of 8085 μP.**

Answer

1. In 8085 microprocessor there is 16 bit address line.
2. Microprocessor 8085 communicates via its address bus of 16 bit wide.
3. The lower byte is AD_0 - AD_7 and upper byte is AD_8 - AD_{15}.

4. Thus it can address a maximum of 2^{16} different address locations. Again each address (memory location) can hold 1 byte of data/instruction.
5. Hence the maximum address capability of 8085 is

$$= 2^{16} \times 1 \text{ Byte} = 65,536 \times 1 \text{ Byte}$$
$$= 64 \text{ kB (where 1 k = 1024 Bytes)}$$

Que 3.26. **Explain the data addressing modes of 8085 with example.**

AKTU 2016-17, Marks 10

OR

What do you understand by instruction set in 8085 ?

Answer

A. **Addressing modes :** The different ways that a microprocessor can access data are referred to as addressing modes.

B. **Types :**

1. **Immediate addressing mode :**

i. In immediate addressing mode, the operand is specified within the instruction itself.

ii. One or two bytes within the instruction are used for specifying the data itself.

iii. **Importance :** In the immediate addressing mode, the data is stored along with the instruction and the data is directly transferred to the register.

iv. **Example :** MVI A, 05H

2. **Register addressing mode :**

i. The register addressing mode specifies the source operand, destination operand, or both to be contained in an 8085 registers.

ii. This results in faster execution, since it is not necessary to access memory locations for operands.

iii. **Importance :** In the register addressing mode, the data is copied from one register and stored in another register (or register pair).

iv. **Example :** MOV A, B

3. **Direct addressing mode :**

i. The direct addressing mode specifies the 16-bit address of the operand within the instruction itself.

ii. The second and third bytes of instruction contain this 16-bit address.

iii. **Example :** LDA 2000H

4. **Indirect addressing mode :**

i. In indirect addressing mode, the memory address where the operand located is specified by the contents of a register pair.

Scanned by TapScanner

ii. **Example** : LDAX B

A. **Implied or implicit addressing mode :**

i. In implied addressing mode, opcode specifies the address of the operands.

ii. **Importance** : In the register addressing mode, no operand (register or memory location or data) is specified.

iii. **Example** : CMA

C. **Instruction set** : Refer Q. 1.23, Page 1–30C, Unit-1.

PART-4

Assembly Language Programming of 8085 Microprocessor with Examples (Arithmetic Operations on 8-bit Numbers-add, Subtract, Multiply, Divide, Square and Square Root etc, Largest/Smallest Number, Ascending/Descending Order).

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.27. **Explain assembler level programming and draw the flowchart of assembler level programming.**

AKTU 2017-18, Marks 10

Answer

A. **Assembler level programming :**

1. A program is a set of instructions arranged in the specific sequence to do the specific task. It tells the microprocessor what it has to do.
2. Though the instructions can be written in hexadecimal code, it is difficult to understand a program written in hexadecimal numbers.
3. Therefore each manufacturer of microprocessor has devised a symbolic code for each instruction, called a mnemonic.
4. The mnemonic for a particular instruction consists of letters that suggest the operation to be performed by that instruction.
5. The complete description of each instruction must be supplied by the manufacturer.
6. The complete set of microprocessor mnemonics is called the assembly language.

7. The process of writing the instruction in assembly language which tells the microprocessor what to do is called "Assembly Language Programming".
8. A format for a typical line from assembly language program can be given as

   Label : Mnemonic operand1, operand2 ; comment

B. **Developing an assembly language program is a four step process :**

i. To specify the source code as per the assembly language definition.

ii. Assemble the program to create the object code.

iii. Link the program to create an executable code.

iv. Test and debug the program.

Fig. 3.27.1 shows the steps involved in developing and executing an assembly language program.

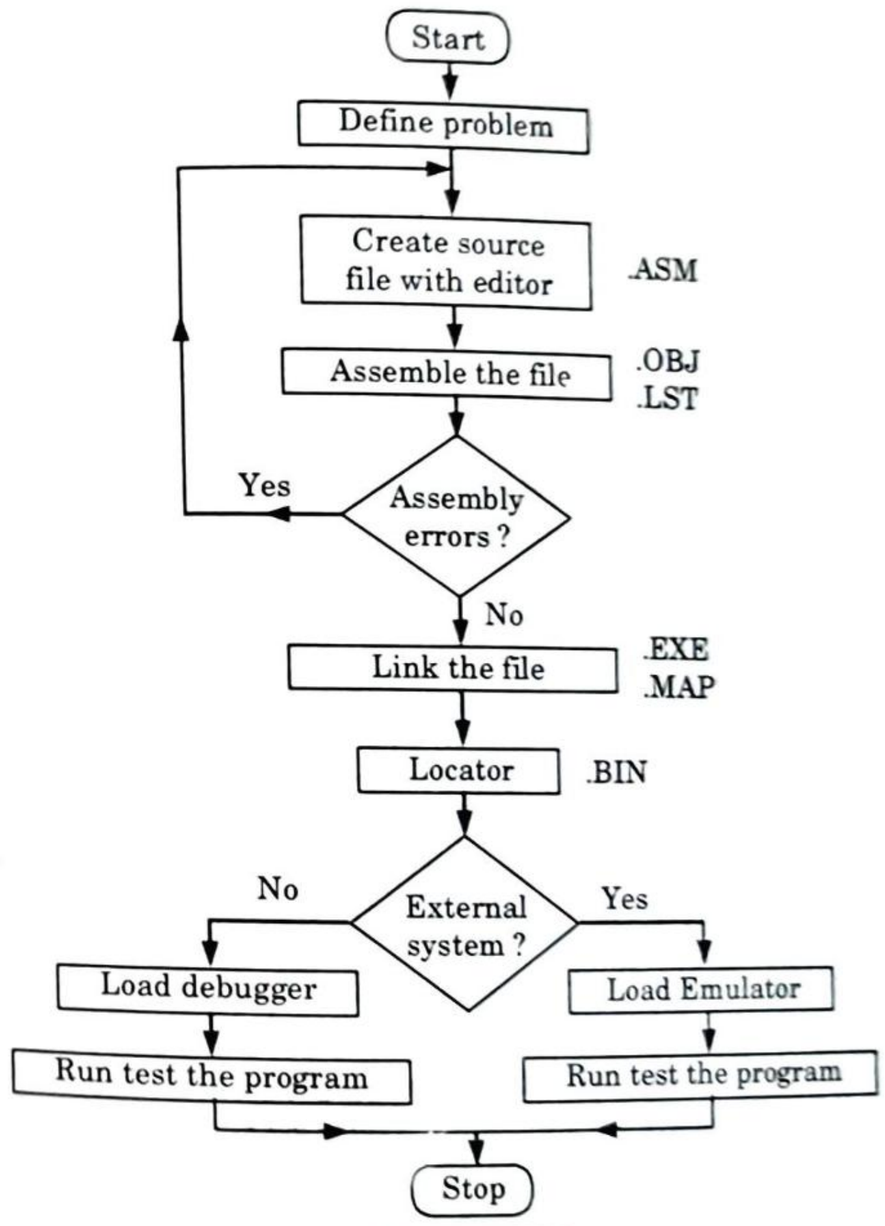


Fig. 3.27.1.

Scanned by TapScanner

Que 3.28. **Why assembly language is used to program microprocessor ? What are the disadvantages of microprocessors ?**

AKTU 2018-19, Marks 07

Answer

A. Reason :

1. The writing of program in machine language is very difficult. Hence, to facilitate programmers easily understandable languages have been developed. Assembly language is one of them.
2. A programmer can easily write a program in alphanumeric symbols instead of zeros and ones. Meaningful and easily rememberable symbols are chosen for the purpose.
3. Examples are : ADD for addition, SUB for subtraction, CMP for comparison etc. Such symbols are called mnemonics.
4. A program written in mnemonics is known as assembly language program.
5. The writing of a program in assembly language is much easier and faster as compared to the writing of a program in machine language.
6. Both assembly language and machine language are microprocessor-specific. A microprocessor-specific language is known as a low-level language.
7. A program which translates an assembly language program into a machine language program is called an assembler.

B. Disadvantages :

1. The microprocessor has a limitation on the size of data.
2. Most of the microprocessor does not support floating point operations.
3. The main disadvantage is its overheating physically.
4. It should not contact with the other external devices.

Que 3.29. **Draw the flow chart and write assembly language program for the addition of two 16-bit numbers considering carry. The numbers are stored in memory starting from 2000H. Store the result of addition and carry from memory 3000H.**

AKTU 2015-16, Marks 10

Answer

A. Flow chart :

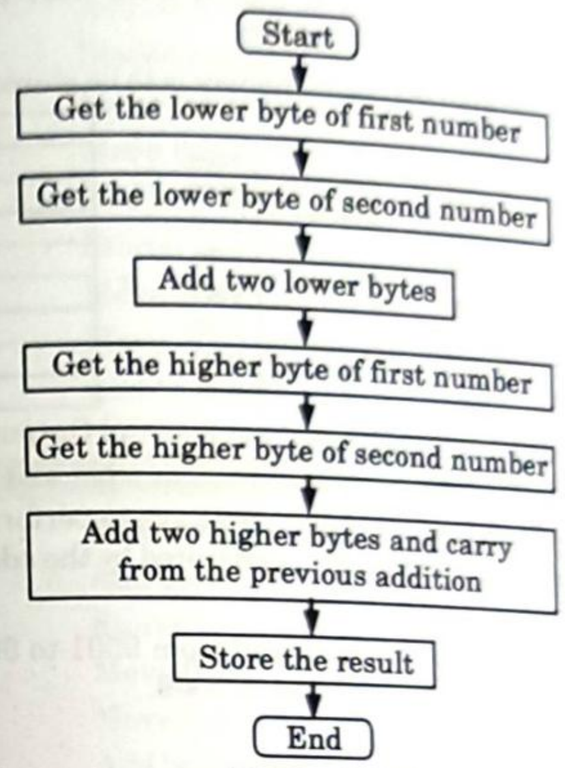


Fig. 3.29.1.

B. Program :

LHLD 2000H	:	Get first 16-bit number in HL
XCHG	:	Save first 16-bit number in DE
LHLD 2002H	:	Get second 16-bit number in HL
MOV A, E	:	Get lower byte of the first number
ADD L	:	Add lower byte of the second number
MOV L, A	:	Store result in L register
MOV A, D	:	Get higher byte of the first number
ADC H	:	Add higher byte of the second number with carry
MOV H, A	:	Store result in H register
SHLD 3000H	:	Store 16-bit result in memory locations 3001H and 3002H
HLT	:	Terminate program execution

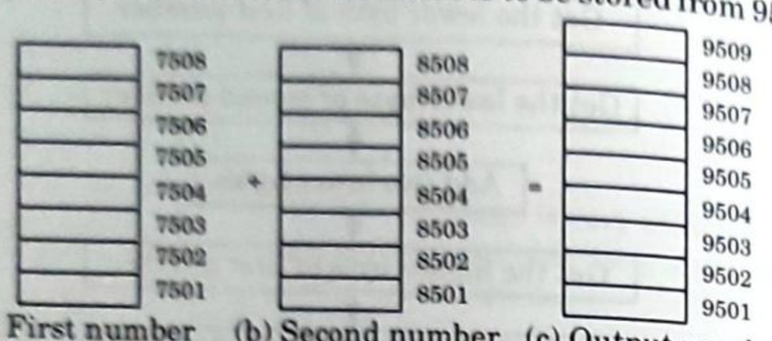
Que 3.30. **Develop a program to add two numbers of 8 byte long whose result is more than 8 byte. The first number is stored from 7501 to 7508 and second number is stored from 8501 to 8508 and result is to be stored from 9501 to 9509.**

AKTU 2016-17, Marks 10

Scanned by TapScanner

Answer

1. Given that the first number is stored from 7501 to 7508 and second number is stored from 8501 to 8508.
2. The output of the sum of these numbers is to be stored from 9501 to 9509



(a) First number (b) Second number (c) Output number

Fig. 3.30.1. Memory location of number and output.

3. For addition of two 8-byte number, we have to perform addition of each byte of each number and the carry generated by the addition is forwarded to the next byte.
4. The sum of each byte is to be stored from 9501 to 9508 and the carry generated at 8th bit is to be stored in 9509.

Program :

LDA, 7501	:	Move 1st byte of first number into A
LDB, 8501	:	Move 1st byte of second number into B
ADD B	:	Add both byte
STA, 9501	:	Stored result to output memory location 9501
LDA, 7502	:	Move 2nd byte of first number into A
LDB, 8502	:	Move 2nd byte of second number into B
ADC B	:	Add both byte
STA, 9502	:	Stored result to output memory location 9502
LDA, 7503	:	Move 3rd byte of first number into A
LDB, 8503	:	Move 3rd byte of second number into B
ADC B	:	Add both byte
STA, 9503	:	Stored result to output memory location 9503
LDA, 7504	:	Move 4th byte of first number into A
LDB, 8504	:	Move 4th byte of second number into B
ADC B	:	Add both byte
STA, 9504	:	Stored result to output memory location 9504

LDA, 7505	:	Move 5th byte of first number into A
LDB, 8505	:	Move 5th byte of second number into B
ADC B	:	Add both byte
STA, 9505	:	Stored result to output memory location 9505
LDA, 7506	:	Move 6th byte of first number into A
LDB, 8506	:	Move 6th byte of second number into B
ADC B	:	Add both byte
STA, 9506	:	Stored result to output memory location 9506
LDA, 7507	:	Move 7th byte of first number into A
LDB, 8507	:	Move 7th byte of second number into B
ADC B	:	Add both byte
STA, 9507	:	Stored result to output memory location 9507
LDA, 7508	:	Move 8th byte of first number into A
LDB, 8508	:	Move 8th byte of second number into B
ADC B	:	Add both byte
STA, 9508	:	Stored result to output memory location 9508
MVI A, 0	:	Move 0 into A
MVI, B, 0	:	Move 0 into B
ADC B	:	Add both byte
STA, 9509	:	Stored carry to output memory location 9509

Que 3.31. **Write 8085 Assembly language program to subtract two 8-bit numbers and store the result at locations 8050H and 8051H.**

Answer

Program :

Address	Mnemonics	Comments
F000	MVI C, 00H	Clear C register
F002	LXI H, 8000H	Load initial address to get operand
F005	MOV A, M	Load Acc with the memory element
F006	INX H	Point to next location
F007	MOV B, M	Load B with the second operand
F008	SUB B	Subtract B from A

Scanned by TapScanner

F009	JNC STORE	When CY = 0, go to STORE
F00C	INR C	Increase C by 1
F00D	LXI H, 8050H	Load the destination address
F010	MOV M, A	Store the result
F011	INX H	Point to next location
F012	MOV M, C	Store the borrow
F013	HLT	Terminate the program

Que 3.32. **Write 8085 Assembly language program to multiply two 8-bit numbers.**

Answer

Program :

Address	Mnemonics	Comments
2000	LHLD 2050	Loads content of 2051 in H and content of 2050 in L
2003	XCHG	Exchanges contents of H with D and contents of L with E
2004	MOV C, D	Copies content of D in C
2005	MVI D 00	Assigns 00 to D
2007	LXI H 0000	Assigns 00 to H and 00 to L
200A	DAD D	Adds HL and DE and assigns the result to HL
200B	DCR C	Decrements C by 1
200C	JNZ 200A	Jumps program counter to 200A if zero flag = 0
200F	SHLD 3050	Stores value of H at memory location 3051 and L at 3050
2012	HLT	Stops executing the program and halts any further execution

Que 3.33. **Write 8085 Assembly language program to divide two 8-bit numbers.**

Answer

Program :

Address	Mnemonics	Comments
2000	LXI H, 2050	Load the HL pair register with the address 2050 of memory location.
2003	MOV B, M	Copies the content of memory into register B.
2004	MVI C, 00	Assign 00 to C.
2006	INX H	Increment register pair HL.
2007	MOV A, M	Copies the content of memory into accumulator.
2008	CMP B	Compares the content of accumulator and register B.
2009	JC 2011	Jump to address 2011 if carry flag is set.
200C	SUB B	Subtract the content of accumulator with register B and store the result in accumulator.
200D	INR C	Increment the register C.
200E	JMP 2008	Control will shift to memory address 2008.
2011	STA 3050	Stores the remainder at memory location 3050.
2014	MOV A, C	Copies the content of register into accumulator.
2015	STA 3051	Stores the remainder at memory location 3051.
2018	HLT	Stops executing the program and halts any further execution.

Que 3.34. **Write an assembly level program to find square root of given number.** **AKTU 2017-18, Marks 10**

Scanned by TapScanner

Answer

Program :

Address	Mnemonics	Comments
2000	MVI D, 01	Initialize register D with 01
2002	MVI E, 01	Initialize register E with 01
2004	LDA 2050	Loads the content of memory location 2050 in accumulator A
2007	SUB D	Subtract value of D from A
2008	JZ 2011	Make jump to memory location 2011 if zero flag is set
200B 200C	INR D INR D	Increments value of register D by 1. Since it is used two times, therefore value of D is incremented by 2
200D	INR E	Increments value of register E by 1
200E	JMP 2007	Make jump to memory location 2007
2011	MOV A, E	Moves the value of register E in accumulator A
2012	STA 3050	Stores value of A in 3050
2015	HLT	Stops executing the program and halts any further execution

Que 3.35. **Write 8085 Assembly language program to find the square of 8-bit numbers.**

Answer

Program :

Address	Mnemonics	Comments
2000	MVI H 20	Initialize register H with 20
2002	MVI L 50	Initialize register L with 50
2004	MVI A 00	Initialize accumulator A with 00
2006	MOV B, M	Moves the content of memory location which is indirectly specified by M in register B

2007	ADD M	Add the content of memory location which is indirectly specified by M in accumulator A
2008	DCR B	Decrement value of register B by 1
2009	JNZ 2007	Jump to memory location 2007 if ZF = 0, *i.e.*, register B does not contain 0
200C	STA 3050	Stores value of A in 3050
200F	HLT	Stops executing the program and halts any further execution

Que 3.36. **Write an assembly language program to generate a delay of 1 msec. Also show the calculation of time delay. Assume that the crystal frequency of 8085 is 6 MHz.** **AKTU 2015-16, Marks 10**

Answer

A. Program :

```
        LXI B, Count
BACK :  DCX B
        MOV A, C
        ORA B
        JNZ BACK
```

B. Calculation of time delay :

1. Operating frequency $= \dfrac{6 \times 10^6}{2} = 3$ MHz
2. Time for one T-state $= \dfrac{1}{3 \times 10^6} = 0.33\ \mu s$
3. Required delay = 1 ms
4. Total number of T-state

$$= \frac{\text{Required delay}}{\text{Time for } T\text{-state}} = \frac{1 \times 10^{-3}}{0.33 \times 10^{-6}}$$

$$= 3 \times 10^3$$

5. Now, calculating count

$$3 \times 10^3 = 10 + (\text{Count} - 1) \times 24 + 21$$

$$\frac{3 \times 10^3 - 31}{24} + 1 = \text{Count}$$

Scanned by TapScanner

$$\text{Count} = 124.708 \approx (124)_{10}$$
$$= 7CH$$

Que 3.37. **Develop an assembly language program for 8085 to add 5 numbers of 8 bits, whose sum is of more than 8 bits. The numbers are stored from 8501 to 8505. The result is to be stored in 8601 and 8602.**

AKTU 2016-17, Marks 10

Answer

Opcode	Operand	Comments
XRA	A	Clear (A) to save sum
MOV	B, A	Clear (B) to save carry
MVI	C, 05H	Set up register C as a counter
LXI	H 8501	Set up HL as memory pointer
ADD	M	Add byte from memory
JNC	NXTMEM	If no carry, do not increment carry register
INR	B	If carry, save carry bit
INX	H	Point to next memory location
DCR	C	One addition is completed; decrement counter
JNZ	NXTBYT	If all bytes are not yet added, go back to get next byte output display
STA	8601	Move the content of A to location 8601
MOV A,	B	Move the content of B to A
STA	8602	Move the content of A to location 8602
	HLT	End of program

Que 3.38. **Write an assembly language program in 8085 to transfer the contents of flag register into *D* register.**

AKTU 2018-19, Marks 07

Answer

Address	Mnemonics	Comments
F000	LXI SP, 9000	Initialize stack pointer
F003	MVI D, 45H	Load D with 45H
F005	MOV E, D	Save D into E
F006	PUSH PSW	Store AF into stack
F007	PUSH D	Store DE into stack
F008	POP PSW	Pop and store to AF
F009	POP D	Pop and store to DE
F00A	MOV D, E	Move E to D
F00B	HLT	Terminate the program

Que 3.39. **Write an assembly level program in 8085 to convert a binary number into its equivalent BCD form.**

AKTU 2018-19, Marks 07

Answer

Address	Mnemonics	Comments
F000	LXI H, 8000H	Initialize memory pointer
F003	MVI D, 00H	Clear D register for most significant Byte
F005	XRA A	Clear accumulator
F006	MOV C, M	Get HEX data
F007	ADI 01H	Count the number one by one
F009	DAA	Adjust for BCD count
F00A	JNC SKIP	Jump to SKIP
F00D	INR D	Increase D
F00E	DCR C	Decrease C register
F00F	JNZ LOOP	Jump to LOOP

Scanned by TapScanner

F012	MOV L, A	Load the least significant byte
F013	MOV H, D	Load the most significant byte
F014	SHLD 8050H	Store the BCD
F017	HLT	Terminate the program

Que 3.40. **An 8085 is executing the following program :**

```
2000 :LXI H, 4325H;
LXI SP, 3000H;
MOV A, H;
ADD L;
PUSH PSW;
POP H;
END
```

At the end of program execution, what will be the contents of the HL register pair ? **AKTU 2018-19, Marks 07**

Answer

Address	Mnemonics	Comments
2000	LXI H, 4325H	Store 4325H into the HL register pair H → 43H L → 25H
2003	LXI SP, 300H	Initialize stack pointer with 16-bit data 3000H ⋮ ⋮ ⋮ 2FFFH [ ] SP → 3000H [ ]
2006	MOV A, H	Move 8-bit data of H in Accumulator (A) in A → 43H
2007	ADD L	Add data of A with data of L and then store the result in A A 43H 0100 0011 L 25H 0010 0101 + 0110 1000 in A store → 68H

2008	PUSH PSW	Store the contents of PSW into two locations of stack. PSW is 16-bit register which is combination of A and flag register Flag register → [S | Z | X | AC | X | P | X | C] Result is positive so SF = 0 Result is not zero so ZF = 0 No auxiliary carry is generated so AC = 0 There is odd parity so PF = 0 No carry is generated so CF = 0 Flag register → 00000000 ⇒ 00H 2FFEH [00] (flag register) 2FFFH [68] (data of A) 3000H [X]
2009H	POP H	Load register pair H, L by popping out 2 bytes from top of the stack So, In L → 00H In H → 68H

Contents of register pair HL is 6800H

Que 3.41. **Write 8085 Assembly language program to find the largest number in an array.**

Answer

Program :

Address	Mnemonics	Comments
2000	LXI H 2050	Assigns 20 to H and 50 to L
2003	MOV C, M	Copies content of memory (specified by HL register pair) to C (this is used as a counter)
2004	DCR C	Decrements value of C by 1
2005	INX H	Increases value of HL by 1. This is done to visit next memory location
2006	MOV A, M	Copy content of memory (specified by HL register pair) to A
2007	INX H	Increases value of HL by 1. This is done to visit next memory location

Scanned by TapScanner

2008	CMP M	Compares A and M by subtracting M from A. Carry flag and sign flag becomes set if A – M is negative
2009	JNC 200D	Jumps program counter to 200D if carry flag = 0
200C	MOV A, M	Copies content of memory (specified by HL register pair) to A
200D	DCR C	Decrements value of C by 1
200E	JNZ 2007	Jumps program counter to 2007 if zero flag = 0
2011	STA 3050	Stores value of A at 3050 memory location
2014	HLT	Stops executing the program and halts any further execution

Que 3.42. **Write 8085 Assembly language program to find the ascending order in an array.**

Answer

Memory	Mnemonics	Comments
4400	LXI H, 4200	Load the array size to the HL pair
4403	MOV C, M	Copy the array size to C register
4404	DCR C	Decrement C by 1
4405	MOV D, C	Copy content of C to D register
4406	LXI H, 4201	Load the first data to the HL pair
4409	MOV A, M	Copy the data to the accumulator
440A	INX H	Increment memory by 1
440B	CMP M	Compare accumulator and memory content
440C	JC SKIP	Jump on carry to the label SKIP
440F	MOV B, M	Copy memory content to B register

4410	MOV M, A	Copy accumulator content to memory
4411	DCX H	Decrement memory by 1
4412	MOV M, B	Copy B register's content to memory
4413	INX H	Increment memory by 1
4414	DCR D	Decrement D by 1
4415	JNZ LOOP	Jump on non-zero to the label LOOP
4418	DCR C	Decrement C by 1
4419	JNZ REPEAT	Jump on non-zero to the label REPEAT
441C	HLT	Program ends

Que 3.43. **Give a block diagram and describe the use of microprocessor to control the temperature of an electric oven. With the help of flow chart explain the algorithm used for temperature control.**

AKTU 2015-16, Marks 15

Answer

A. **Block diagram :**

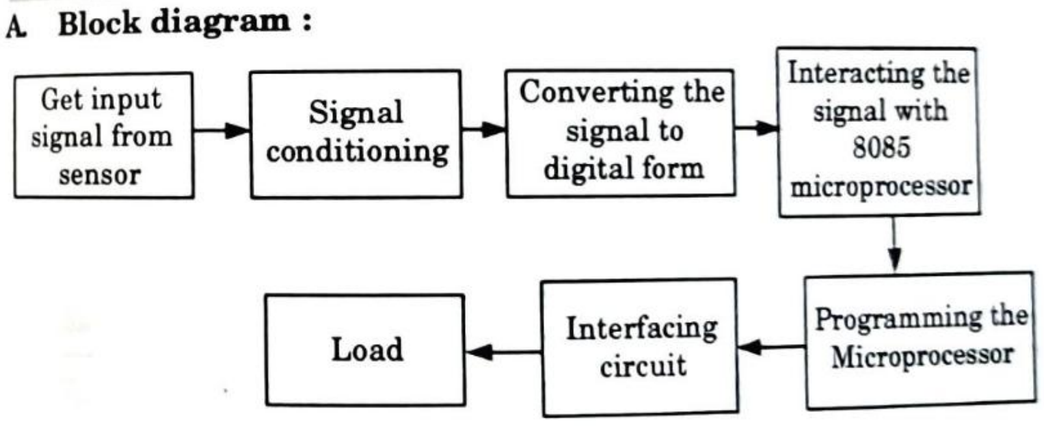


Fig. 3.43.1.

1. **Input from sensor :** For temperature sensing temperature sensor is used.
2. **Signal conditioning :** The signal conditioning is done so as to interface the sensor's output with the analog to digital converter.
3. **Converting to digital form :** A/D converter is used for analog to digital conversion. The analog signal which we get from the sensors is amplified to 0.5 V and is given to A/D converter.

Scanned by TapScanner

4. **Interfacing with microprocessor :** The digital output from the A/D converter is interfaced with the microprocessor through a programmable peripheral interface.

B. Algorithm used for temperature control :

1. Configure programmable peripheral interface I/O ports.
2. 8085 sends SOC command to A/D converter.
3. 8085 waits for EOC signal from A/D converter.
4. 8085 reads 8-bit temperature value.
5. 8085 compares the value with set point value.
6. 8085 generates the control signal to control load.

C. Flow Chart :

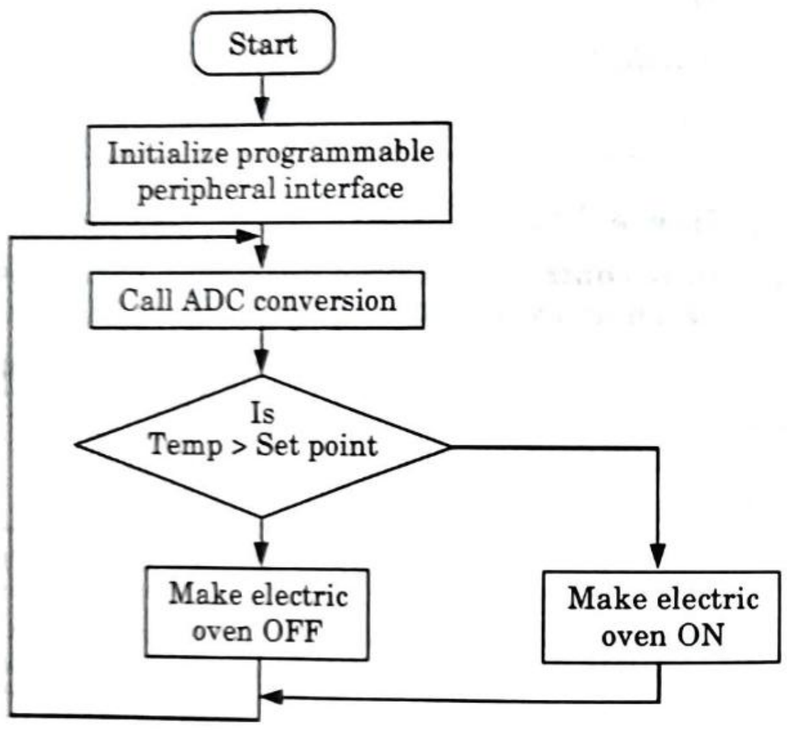


Fig. 3.43.2.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. What do you understand by DMA ? With the help of block diagram explain the working of 8237/8257.

Ans. Refer Q. 3.1.

Q. 2. What is meant by direct memory access ? With the help of neat schematic, explain the interaction between 8086 MPU and DMA controller for providing direct memory access to a peripheral device.

Ans. Refer Q. 3.3.

Q. 3. Draw and explain the internal architecture of 8255 parallel I/O peripheral device. Also describe the bits of control word.

Ans. Refer Q. 3.7.

Q. 4. Draw and explain block diagram and pin configuration of IC-8253.

Ans. Refer Q. 3.12.

Q. 5. Explain the architecture of 8259A.

Ans. Refer Q. 3.16.

Q. 6. Explain the internal architecture of 8251.

Ans. Refer Q. 3.20.

Q. 7. Explain RS-232C.

Ans. Refer Q. 3.22.

Q. 8. Explain the data addressing modes of 8085 with example.

Ans. Refer Q. 3.26.

Q. 9. Explain assembler level programming and draw the flowchart of assembler level programming.

Ans. Refer Q. 3.27.

Q. 10. Develop a program to add two numbers of 8 byte long whose result is more than 8 byte. The first number is stored from 7501 to 7508 and second number is stored from 8501 to 8508 and result is to be stored from 9501 to 9509.

Ans. Refer Q. 3.30.

Q. 11. Write an assembly level program to find square root of given number.

Ans. Refer Q. 3.34.

☺☺☺

Scanned by TapScanner

4 UNIT

8051 Microcontroller Basics

CONTENTS

Part-1 : Inside the Computer, Microcontrollers and Embedded Processors ..... **4–2C to 4–5C**

Part-2 : Block Diagram of 8051, PSW and Flag Bits, 8051' Register Banks and Stack ..... **4–5C to 4–10C**

Part-3 : Internal Memory Organization of 8051, I/O Port Usage in 8051, Types of Special Function Registers and their Uses in 8051, Pins of 8051 ..... **4–10C to 4–14C**

Part-4 : Memory Address Decoding, 8031/51, Interfacing with External ROM and RAM, 8051 Addressing Modes ..... **4–14C to 4–20C**

PART-1

Inside the Computer, Microcontrollers and Embedded Processors.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.1. **Draw and explain the block diagram of computer.**

Answer

1. The internal working of every computer can be broken down into three parts : CPU (Central Processing Unit), memory and I/O (Input/Output) devices as shown in Fig. 4.1.1.

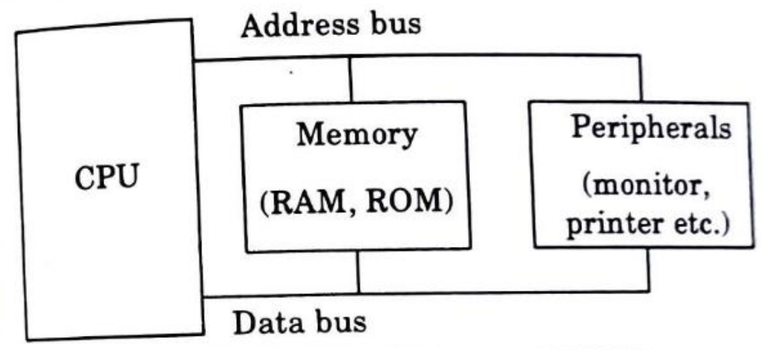


Fig. 4.1.1. Inside the computer.

2. The function of the CPU is to execute (process) information stored in memory.
3. The function of I/O devices such as the keyboard and video monitor is to provide a means of communicating with the CPU.
4. The CPU is connected to memory and I/O through strips of wire called a bus. The bus carries information from place to place inside a computer.
5. In every computer there are three types of buses: address bus, data bus, and control bus.
6. The CPU puts the address (in binary) on the address bus, and the decoding circuitry finds the device. Then the CPU uses the data bus either to get data from that device or to send data to it.
7. The control buses are used to provided read or write signals to the device to indicate if the CPU is asking for information or sending it information.
8. Of the three buses, the address bus and data bus determine the capability of a given CPU.

Scanned by TapScanner

Que 4.2. **What do you mean by embedded system ? Describe its purpose and main application areas.**

Answer

A. Embedded system :

1. An embedded system is an electronic/electro-mechanical system designed to perform a specific function and is a combination of both hardware and firmware (software).
2. Every embedded system is unique, and the hardware as well as the firmware is highly specialised to the application domain.
3. Embedded systems are becoming an inevitable part of any product or equipment in all fields including household appliances, telecommunications, consumer products etc.

B. Purpose :

Each embedded system is designed to serve the purpose of any one or a combination of the following tasks :

1. Data collection/storage/representation.
2. Data communication.
3. Data (signal) processing.
4. Monitoring.
5. Control.
6. Application specific user interface.

C. Application :

1. Consumer electronics : Camcorders, cameras, etc.
2. Household appliances : T.V, washing machine, fridge, etc.
3. Home automation and security system : Air conditioners, sprinklers, fire alarms, etc.
4. Telecom : Cellular telephones, telephone switches, etc.
5. Computer peripherals : Printer, scanners, etc.
6. Healthcare : EEG, ECG machines etc.

Que 4.3. **List some important features and architecture consideration of an embedded system.**

Answer

A. Important features :

1. Embedded systems execute pre-programmed functions and they have a particular set of requirements.

2. They are programmed hardware devices that run on hardware chips that are programmable.
3. Embedded systems perform a specific function or a set of specific functions unlike a computer, which is used to carry out a wide number of functions.
4. Embedded systems form smaller parts of a much larger device that carries out a specific task.
5. Embedded systems can either have no user interface or possess highly advanced graphical interfaces. It mainly depends on the purpose of the device or the function it is designed to carry out.
6. Simple embedded systems use LEDs, buttons or LCD displays with simple menu options.

B. Architecture considerations :

1. The architecture of an embedded system is an abstraction of the embedded device, meaning that it is a generalization of the system.
2. The hardware and software components in an embedded system are instead represented as composition of interacting elements.
3. Architectural element can be internally integrated within the embedded device or exist externally to the embedded system and interact with internal elements.

Que 4.4. **Explain microcontroller. Why microcontrollers are preferred for controlling operation ?**

Answer

A. Microcontroller :

1. A microcontroller contains a fixed amount of RAM, ROM, parallel I/O ports, serial I/O ports, counter and a clock circuit all on a single chip.
2. The block diagram of microcontroller is shown in Fig. 4.4.1.

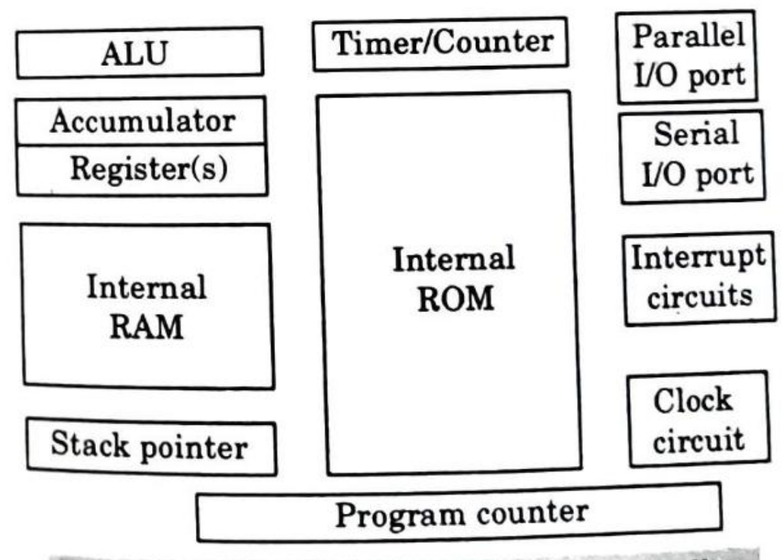


Fig. 4.4.1. Block diagram of a microcontroller.

Scanned by TapScanner

3. Microcontrollers are intended to be special purpose digital computers.
4. Microcontroller is concerned with rapid movement of bits within the chip. It also consists of many bit-handling instructions.
5. Microcontrollers are normally less expensive than microprocessor.

B. Microcontrollers are preferred over microprocessors for controlling operations because :

1. They meet the computing needs of the task efficiently and cost effectively.
2. Software development tools such as compilers, assemblers and debuggers are available for operations.
3. Wide availability and reliable sources of the microcontroller.
4. The power consumption of this device is also very low.

PART-2

Block Diagram of 8051, PSW and Flag Bits, 8051 Register Banks and Stack.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.5. **Give the main features of 8051 microcontrollers.**

Answer

1. 8-bit CPU with registers A (the accumulator) and B.
2. 16-bit program counter (PC) and data pointer (DPTR).
3. 8-bit program status word (PSW).
4. 8-bit stack pointer (SP).
5. Internal ROM or EPROM (8751) of 0 (8031) to 4K (8051).
6. Internal RAM of 128 bytes :
   i. Four register banks, each containing eight registers.
   ii. Sixteen bytes, which may be addressed at the bit level.
   iii. Eighty bytes of general-purpose data memory.
7. 32 input/output pins arranged as four 8-bit ports : P0–P3.
8. Two 16-bit timers/counters : T0 and T1.
9. Full duplex serial data receiver/transmitter : SBUF.
10. Control registers : TCON, TMOD, SCON, PCON, IP and IE.

11. Two external and three internal interrupt sources.
12. Oscillator and clock circuits.

Que 4.6. **Draw the functional block diagram of 8051 microcontroller.**

Answer

1. The 8051 microcontroller contains 128 bytes of RAM, 4 Kbytes of on-chip ROM, two timers, one serial port, and four ports (each 8-bits wide) all on a single chip. It is also referred as a "system on a chip".
2. The 8051 is an 8-bit processor, meaning that the CPU can work on only 8-bits of data at time. Data larger than 8 bits has to be broken into 8-bit pieces to be processed by the CPU.
3. The 8051 has a total of four I/O ports, each 8 bits wide as shown in Fig. 4.6.1.

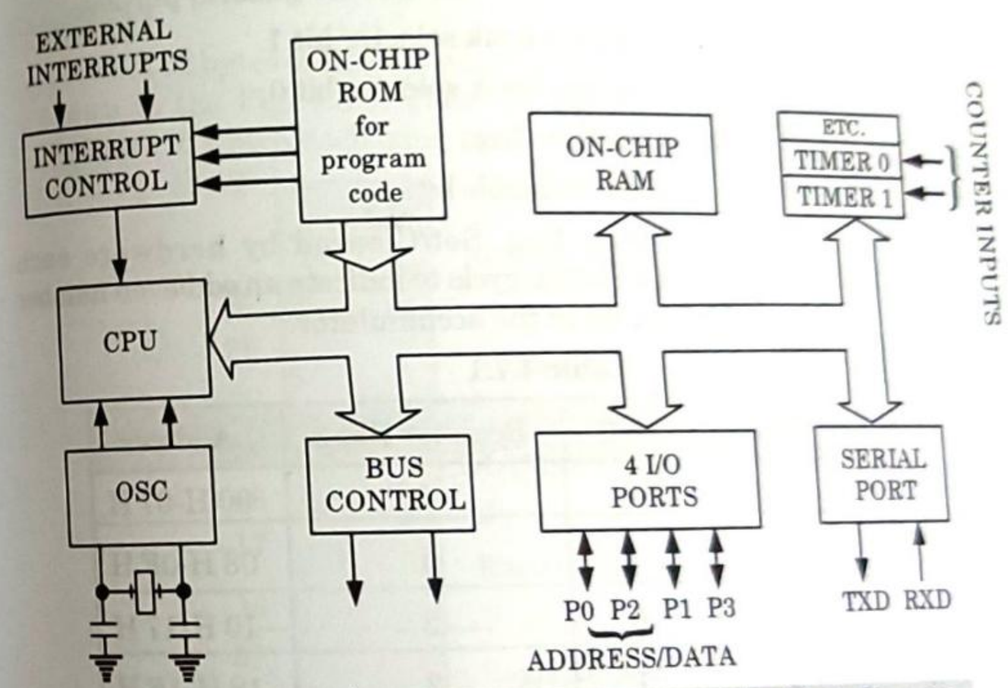


Fig. 4.6.1. Inside the 8051 microcontroller block diagram.

Que 4.7. **What is PSW register ? Explain the format of PSW of 8051.**

Answer

A. **PSW :** The Program Status Word (PSW) register is an 8-bit register. It is also referred to as the flag register. Although the PSW register is 8 bits wide, only 6 bits of it are used by the 8051. The two unused bits are user-definable flags.

Scanned by TapScanner

B. Format of PSW :

1. In this, four flags are called conditional flags, meaning that they indicate some conditions that result after an instruction executed.
2. These four are CY (carry), AC (auxiliary carry), P (parity), and OV (overflow) as shown in Fig. 4.7.1.
3. The bits PSW.3 and PSW.4 are designated as RS0 and RS1 respectively, and are used to change the bank registers.
4. The PSW.5 and PSW.1 bits are general-purpose status flag bits and can be used by the programmer for any purpose. In other words, they are user-definable.

CY	AC	F0	RS1	RS0	OV	–	P

CY	PSW.7	Carry flag.
AC	PSW.6	Auxiliary carry flag.
F0	PSW.5	Available to the user for general purpose.
RS1	PSW.4	Register bank selector bit 1.
RS0	PSW.3	Register bank selector bit 0.
OV	PSW.2	Overflow flag.
–	PSW.1	User-definable bit.
P	PSW.0	Parity flag. Set/Cleared by hardware each instruction cycle to indicate an odd/even number of 1 bit in the accumulator.

Table 4.7.1

RS1	RS0	Register Bank	Address
0	0	0	00 H-07 H
0	1	1	08 H-0F H
1	0	2	10 H-17 H
1	1	3	18 H-1F H

CY, the carry flag : This flag is set whenever there is a carry out from the D7 bit. This flag bit is affected after an 8-bit addition or subtraction. It can also be set to 1 or 0 directly by an instruction such as "SETB C" and "CLR C" where "SETB C" stands for "set bit carry" and "CLR C" stands for "clear carry".

AC, the auxiliary carry flag :

If there is a carry from D3 to D4 during an ADD or SUB operation, this bit is set, otherwise, it is cleared. This flag is used by instructions that perform BCD (binary coded decimal) arithmetic.

P, the parity flag :

The parity flag reflects the number of 1s in the *A* (accumulator) register only. If the *A* register contains an odd number of 1s, then $P = 1$. Therefore, $P = 0$ if *A* has an even number of 1s.

OV, the overflow flag :

This flag is set whenever the result of a signed number operation is too large, causing the high-order bit to overflow into the sign bit. In general, the carry flag is used to detect errors in unsigned arithmetic operations. The overflow flag is only used to detect errors in signed arithmetic operations.

There is no sign flag in 8051 because of the overflow problem.

Que 4.8. Write a short note on register banks in the 8051 microcontroller.

Answer

1. A total of 32 bytes of RAM is set aside for the register banks and stack as shown in the Fig. 4.8.1. These 32 bytes are divided into 4 banks of registers in which each bank has 8 registers, R0-R7.

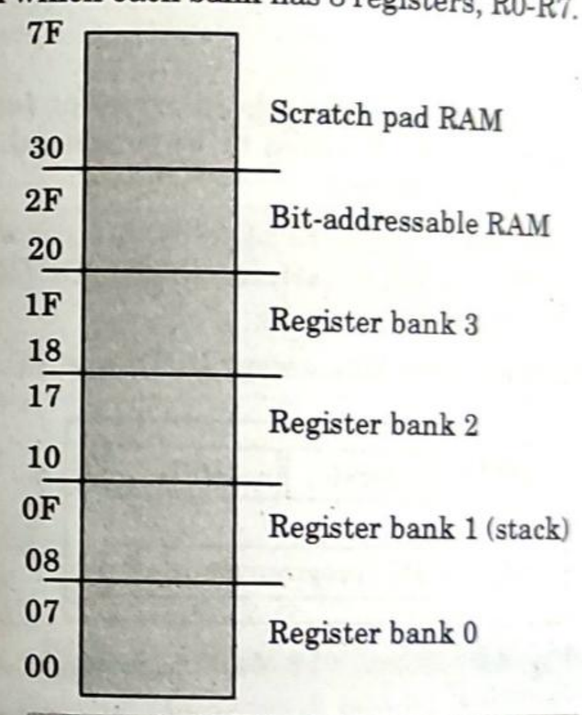


Fig. 4.8.1. RAM allocation in the 8051.

2. RAM locations from 0 to 7 are set aside for bank 0 of R0-R7 where R0 is RAM location 0, R1 is RAM location 1, R2 is location 2, and so on, until memory location 7, which belongs to R7 of bank 0.
3. The second bank of register R0-R7 starts at RAM location 08 and goes to location 0F H.
4. The third bank of R0-R7 starts at memory location 10 H and goes to location 17 H.

Scanned by TapScanner

5. Finally, RAM location 18 H to 1F H are set aside for the fourth bank of R0-R7.
6. Bank 1 uses the same RAM space as the stack. This is a major problem in programming the 8051. One must either not use register bank 1, or allocate another area of RAM for the stack.

Bank 0		Bank 1		Bank 2		Bank 3	
7	R7	F	R7	17	R7	1F	R7
6	R6	E	R6	16	R6	1E	R6
5	R5	D	R5	15	R5	1D	R5
4	R4	C	R4	14	R4	1C	R4
3	R3	B	R3	13	R3	1B	R3
2	R2	A	R2	12	R2	1A	R2
1	R1	9	R1	11	R1	19	R1
0	R0	8	R0	10	R0	18	R0

Fig. 4.8.2. 8051 Register banks and their RAM addresses.

Que 4.9. **Which are the most widely used registers of 8051 microcontroller ?**

Answer

1. Registers are basically used to store the information temporarily. This information can be a byte of a data to be processed, or an address pointing to the data to be fetched.
2. The most widely used registers of 8051 microcontroller are A (accumulator), *B*, R0, R1, R2, R3, R4, R5, R6, R7, DPTR (Data Pointer) and PC (Program Counter).
3. All the above registers are 8-bits, except DPTR and PC.

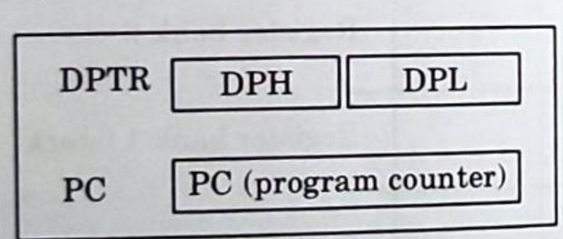


Fig. 4.9.1. Some 8051 16-bit registers.

4. The accumulator or register *A* is used for all arithmetic and logic instructions.
5. The 8051 contains 16-bit registers : the program counter (PC) and the data pointer (DPTR). Each is used to hold the address of a byte in memory. Program instruction bytes are fetched from locations in memory that are addressed by the PC.
6. The DPTR register is made up of two 8-bit registers named DPH and DPL.

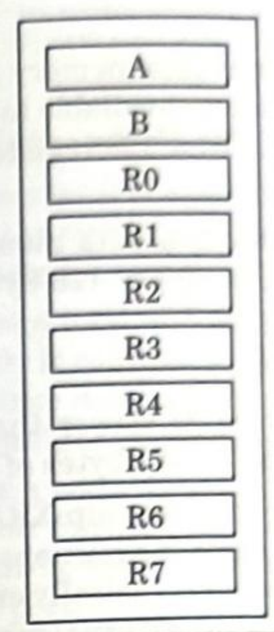


Fig. 4.9.2. Some 8-bit registers of the 8051.

PART-3

Internal Memory Organization of 8051, I/O Port Usage in 8051, Types of Special Function Registers and their Uses in 8051, Pins of 8051.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.10. **Explain the memory organization in 8051 microcontroller.**

Answer

Fig. 4.10.1 shows the basic memory organization for 8051. It can access up to 64 K program memory and 64 K data memory. The 8051 has 4 Kbytes of internal program memory and 256 bytes of internal data memory.

Program Memory : Program Memory (ROM) is used for permanent saving program (CODE) being executed. The memory is read only. Depending on the settings made in compiler, program memory may also used to store constant variables. The 8051 executes programs stored in program memory only.

Scanned by TapScanner

Internal Data Memory :

1. Up to 256 bytes of internal data memory are available depending on the 8051 derivative. Locations available to the user occupy addressing space from 0 to 7F H, *i.e.*, first 128 registers and this part of RAM is divided in several blocks.
2. The first 128 bytes of internal data memory are both directly and indirectly addressable. The upper 128 bytes of data memory can be addressed only indirectly.

External Data Memory :

1. Access to external memory is slower than access to internal data memory. There may be up to 64 Kbytes of external data memory.
2. Several 8051 devices provide on-chip XRAM (External RAM) space that is accessed with the same instructions as the traditional external data space. This XRAM space is typically enabled via proper setting of SFR register and overlaps the external memory space.

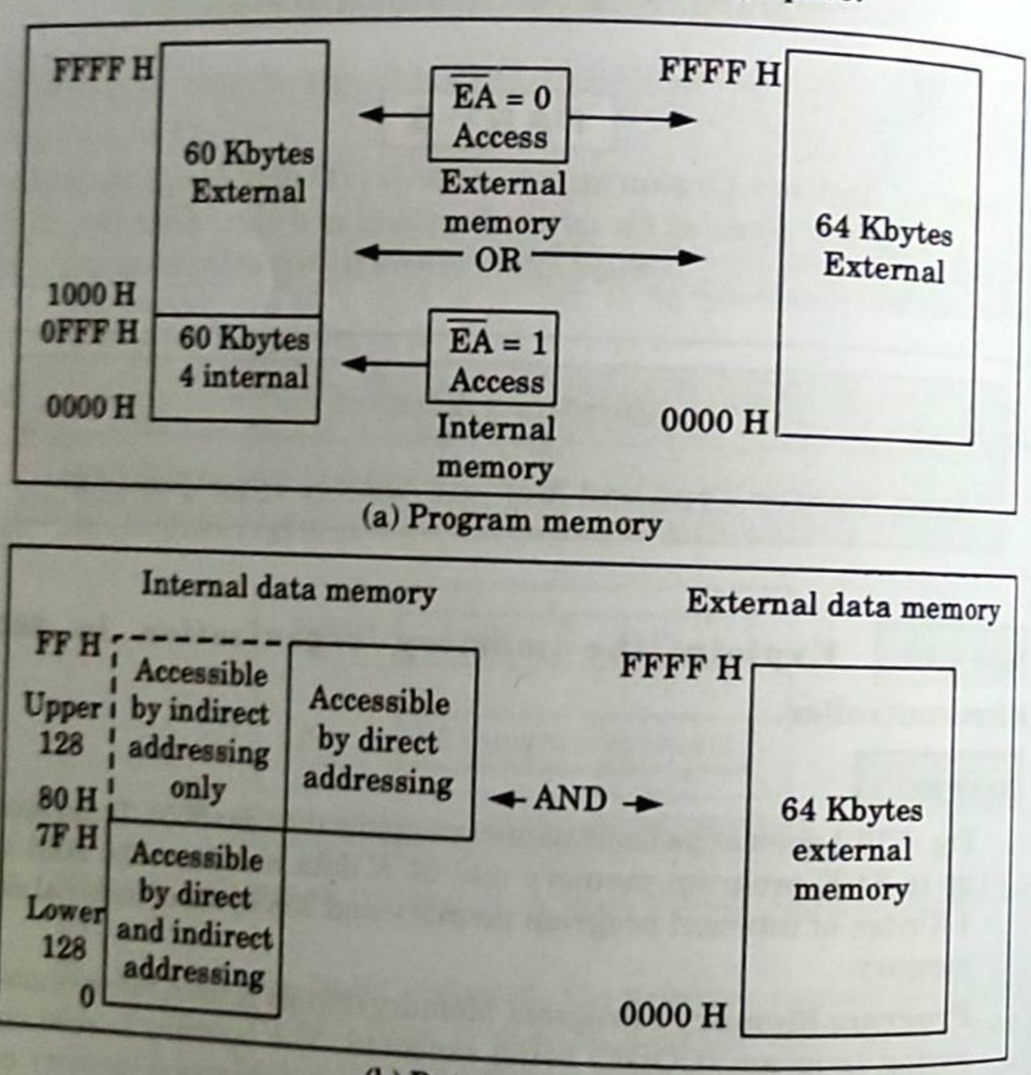


Fig. 4.10.1. Memory organization of 8051.

Que 4.11. Explain I/O ports in 8051 microcontroller.

Answer

There are four I/O ports in 8051 microcontroller :

i. **Port 0 :** Port 0 occupies a total of 8 pins (pins 32-39). It can be used for input or output. To use the pins of port 0 as both input and output ports, each pin must be connected externally to a 10 kΩ pull-up resistor.

ii. **Port 1 :** Port 1 occupies a total of 8 pins (pins 1 through 8). It can be used as input or output. It contrast to port 0, this port does not need any pull-up resistors since it already has pull-up resistors internally. Upon reset, port 1 is configured as an input port.

iii. **Port 2 :** Port 2 occupies a total of 8 pins (pins 21 through 28). It can be used as input or output. Port 2 does not need any pull-up resistors since it already has pull-up resistors internally. Upon reset, port 2 is configured as an input port.

iv. **Port 3 :**

1. Port 3 occupies a total of 8 pins (pins 10 through 17). It can be used as input or output. P3 does not need any pull-up resistors, just as P1 and P2 did not.
2. Although port 3 is configured as an input port upon reset, this is not the way it is most commonly used. Port 3 has the additional function of providing some externally important signals such as interrupts.

Que 4.12. Write a short note on SFR registers and their addresses.

Answer

A. SFR registers :

1. The 8051 operations that do not use the internal 128-bytes RAM addresses from 00 H to 7F H are done by a group of specific internal register, each called a special function register.
2. In 8051, register *A*, *B*, PSW and DPTR are a part of the group of registers referred as SFR.
3. SFR can be accessed by names or by their addresses.

B. SFR register addresses :

1. SFR has addresses between 80 H and FF H. These addresses are above 80H, since address 00 to 7F H are addresses of RAM memory inside the 8051.
2. Not all address space of 80 to FF H are used by SFR.

   Some SFR addresses are

Accumulator	– 0E0 H
B Register	– 0F0 H
PSW	– 0D0 H etc.

Scanned by TapScanner

C. **Use of SFR registers :** The 8051 microcontroller special function register are used to program and control different hardware peripherals like timers, serial port, I/O ports etc.

Que 4.13. Explain pin diagram of 8051.

Answer

1. The pin diagram of 8051 consists of 40 pins. Out of 40 pins 32 pins are set aside for the four ports P0, P1, P2 and P3, where each ports takes 8 pins.
2. The rest of the pins are designated as V_{CC}, GND, XTAL 1, XTAL 2, RST, $\overline{EA}$, ALE/ $\overline{PROG}$ and $\overline{PSEN}$.
3. There are four ports P0, P1, P2 and P3 each use 8 pins, making them 8-bit ports. All the ports upon RESET are configured as inputs, ready to be used as input ports. When the first 0 is written to a port, it becomes an output. To reconfigure as an input, 1 must be sent to port.

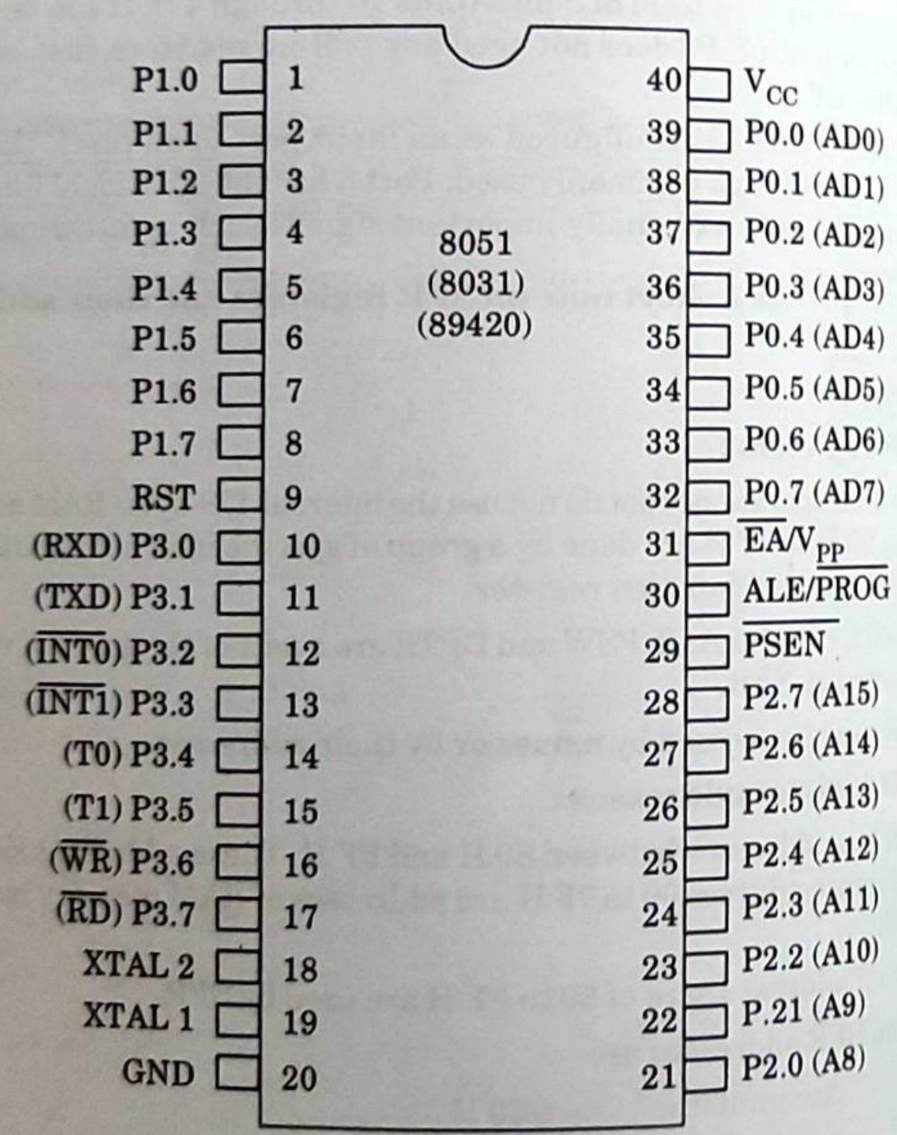


Fig. 4.13.1.

4. V_{CC} : Pin 40 provides supply voltage to the chip. The voltage source is + 5V.

5. **GND :** Pin 20 is the ground.
6. **XTAL 1 and XTAL 2 :** The 8051 has an on chip oscil ator but requires an external clock to run it. A quartz crystal oscillator is connected to inputs XTAL1 (pin 19) and XTAL2 (pin 18). The quartz crystal connected to XTAL1 and XTAL2 needs two capacitors of 30 pF value. One side of each capacitor is connected to ground as shown in Fig. 4.13.2.

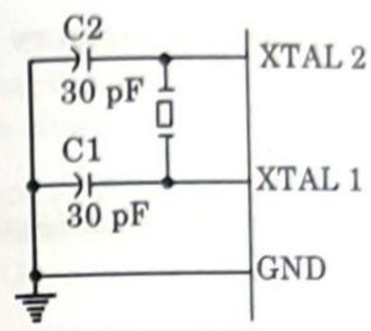


Fig. 4.13.2.

There are various speeds of the 8051 family. Speed refers to the maximum oscillator frequency connected to XTAL.

7. **RST :** Pin 9 is the reset pin. It is an input and is active high. On applying a high pulse to this pin, the microcontroller will reset and terminate all activities. This is referred to as a power-on reset.
8. **$\overline{EA}$:** $\overline{EA}$ stands for external access enable pin. In 8051 chips with on chip ROM such as 8751/52 or DS89C4 × 0, $\overline{EA}$ is connected to V_{CC}. For family members like 8031 and 8032 in which there is no on-chip ROM, code is stored on an external ROM and is fetched by the 8031/32. Thus for 8031, the $\overline{EA}$ pin must be connected to GND to indicate that code is stored externally.

   $\overline{EA}$ pin is an input pin and must be connected to either V_{CC} or GND. It cannot be left unconnected.
9. **$\overline{PSEN}$:** This is an output pin. $\overline{PSEN}$ stands for program store enable. This is an active low output pin. It acts as a strobe to read the external program memory.
10. **ALE :** Address latch enable is an output pin and is active high. It is used for demultiplexing the address and data bus by connecting to the GND pin of 74LS373 chip. This ALE signal is valid only for external memory access.

PART-4

Memory Address Decoding, 8031/51, Interfacing with External ROM and RAM, 8051 Addressing Modes.

Scanned by TapScanner

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.14. Enlist the methods of memory address decoding and explain any of them.

Answer

There are three methods of memory address decoding.

i. Using the 74LS138

ii. Using programmable logic

iii. **Using simple logic gates :**

1. The simplest method of constructing decoding circuitry is the use of a NAND gate. The output of a NAND gate is active low, and the CS pin is also active low, which makes them a perfect match.
2. In cases where the CS input is active high, an AND gate must be used. Using a combination of NAND gates and inverters, one can decode any address range.
3. An example of this is shown in Fig. 4.14.1, which shows that A15 - A12 must be 0011 in order to select the chip.

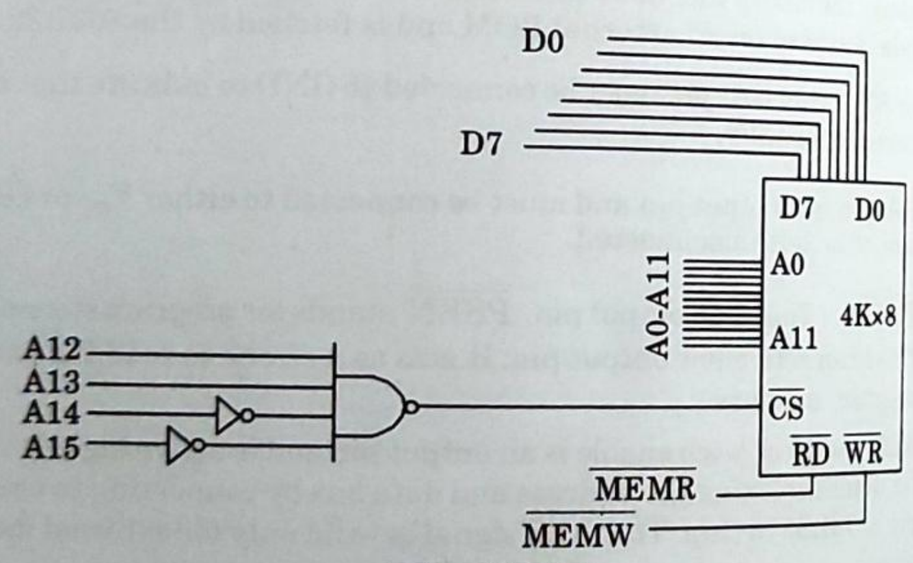


Fig. 4.14.1.

4. This results in the assignment of addresses 3000 H to 3FFF H to this memory chip.

Que 4.15. Explain 74LS138 address decoder.

Answer

1. This is one of the most widely used address decoders.
2. The 3 inputs *A*, *B*, and *C* generate 8 active-low outputs Y0 - Y7 as shown in Fig. 4.15.1.
3. Each *Y* output is connected to CS of a memory chip, allowing control of 8 memory blocks by a single 74LS138.
4. In the 74LS138, where *A*, *B*, and *C* select which output is activated, there are three additional inputs, G2A, G2B, and G1. G2A and G2B are both active low, and G1 is active high.
5. If any one of the inputs G1, G2A, or G2B is not connected to an address signal (sometimes they are connected to a control signal), they must be activated permanently either by V_{CC} or ground, depending on the activation level.

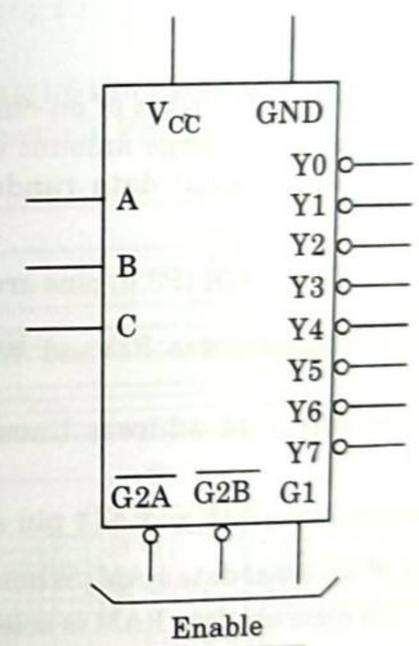


Fig. 4.15.1.

6. Example 4.15.2 shows the design and the address range calculation for the 74LS138 decoder.

Scanned by TapScanner

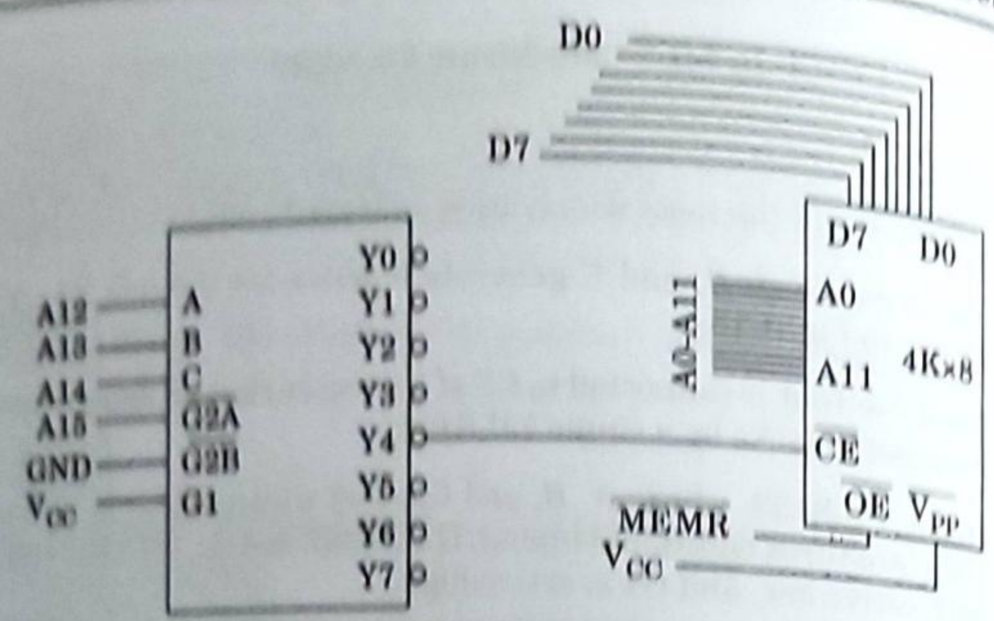


Fig. 4.15.2.

Que 4.16. **With a schematic diagram, explain how an 8051 is interfaced with external RAM ?**

Answer

1. The 8051 devices have only 256 bytes of on-chip data random access memory. In applications where large amount of data random access memory is required, the external data random access memory is interfaced with 8051.
2. Address, data, $\overline{RD}$ (P3.7) and $\overline{WR}$ (P3.6) pins are used to interface data RAM. P3.7 and P3.6 are connected to $\overline{RD}$ and $\overline{WR}$ pins of data RAM as shown in Fig. 4.16.1.
3. The memory chip requires 14 address lines (A13–A0) to decode 16384 × 8 registers.
4. The remaining address lines A15 and A14 pin are connected through OR gate to $\overline{CS}$ pin of external data RAM. When the address lines A15 and A14 are low, then external data RAM is selected.
5. The address of external data RAM is 0000H to 3FFFH. Since $\overline{EA}$ is connected to V_{cc}, internal program memory is selected for address 0000H to 0FFFH.

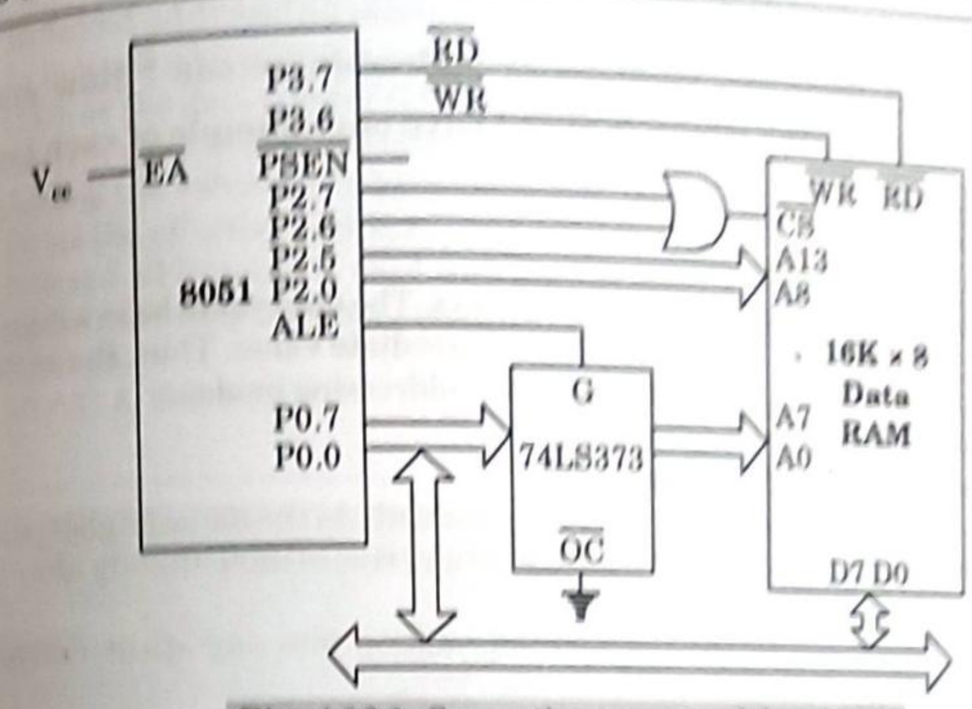


Fig. 4.16.1. Connection to external data RAM.

Que 4.17. **Show the connection of an 8051 to external ROM.**

Answer

1. To connect the 8031/51 to external ROM containing data, we use RD (pin P3.7). It is shown in Fig. 4.17.1.

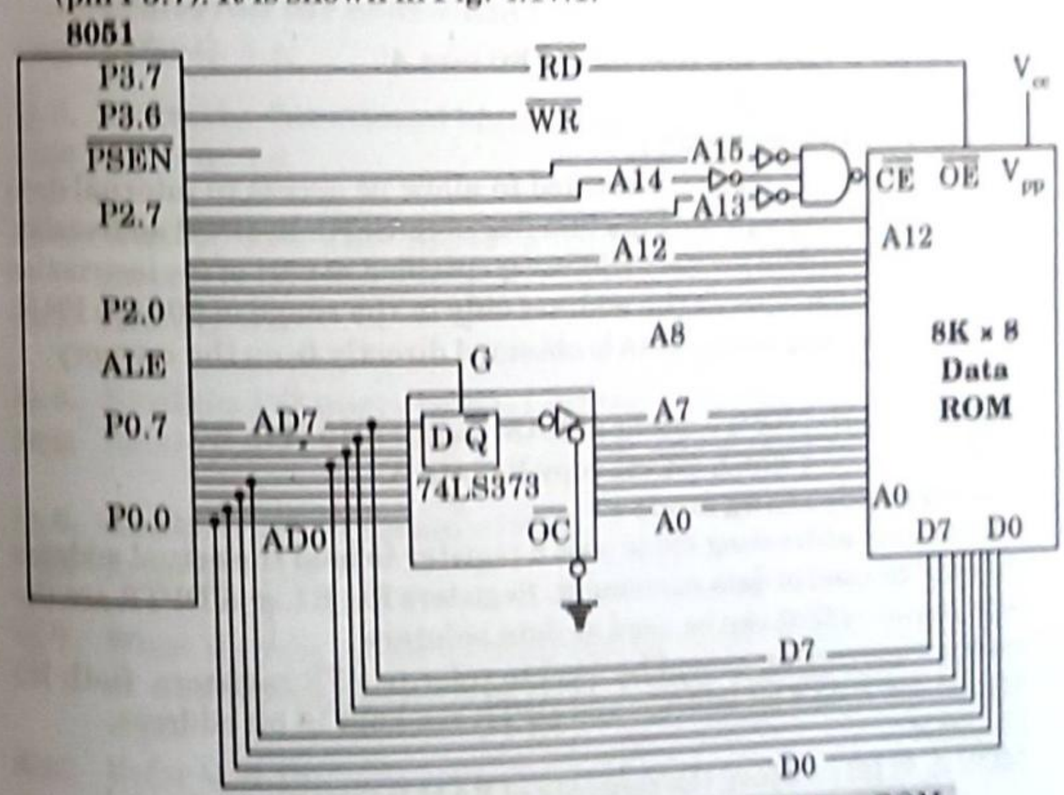


Fig. 4.17.1. 8051 connection to external data ROM.

2. For the ROM containing the program code, PSEN is used to fetch the code. For the ROM containing data, the RD signal is used to fetch the data.

Scanned by TapScanner

Que 4.18. **What do you mean by addressing mode ? How many types of addressing modes in 8051 ? Give one example of each type.**

Answer

A. Addressing mode :

1. The CPU can access data in many ways. The data could be in a register or in memory or be provided as an immediate value. Thus, the various ways of accessing the data are called addressing modes.

B. Types of addressing mode :

i. Immediate addressing mode :

1. In this mode, the source operand is constant. As the name implies, when the instruction is assembled the operand comes immediately after the opcode.
2. This mode can be used to load information into any of the registers including DPTR.
3. Here immediate data must be preceded by the pound sign "#".
   Examples :
   MOV *A*, #25 H; Load 25 H into *A*
   MOV DPTR, #2550 H

ii. Register addressing mode :

1. This mode involves the use of registers to hold the data to be manipulated.
2. Here the source and destination registers must match in size.
   Example :
   MOV *A*, R0 ; Copy the contents of R0 into *A*
   MOV DPTR, #25F5 H

iii. Direct addressing mode :

1. Direct addressing mode is provided to allow us access to internal data memory, including Special Function Register (SFR). In direct addressing, an 8 bit internal data memory address is specified as part of the instruction and hence, it can specify the address only in the range of 00 H to FF H.
2. In this addressing mode, data is obtained directly from the memory.
   Example :
   MOV R0, 40 H ; Save content of RAM location 40 H in R0
   MOV *A*, R4 ; which means copy R4 into A.

iv. Indirect addressing mode :

1. The indirect addressing mode uses a register to hold the actual address that will be used in data movement. Registers R0, R1, and DPTR are the only registers that can be used as data pointers.
2. Indirect addressing cannot be used to refer to SFR registers. Both R0 and R1 can hold 8 bit address and DPTR can hold 16 bit address.
   Example :
   MOV *A*, @ R0 ; Move the contents of RAM location whose address is held by R0 into *A*
   MOV @ R1, *B* ; Move the contents of *B* into RAM location whose address is held by R1.

v. Indexed addressing mode :

1. In indexed addressing, a separate register (either the program counter (PC), or the data pointer (DTPR)) is used to hold the base address, and the *A* is used to hold the offset address.
2. Adding the value of the base address to the value of the offset address forms the effective address.
3. Indexed addressing is used with JMP or MOVC instructions. Look up tables are easily implemented with the help of index addressing.
   Example :
   MOVC A, @ A + DPTR

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. Draw and explain the block diagram of computer.

Ans. Refer Q. 4.1.

Q. 2. Explain microcontroller. Why microcontrollers are preferred for controlling operation ?

Ans. Refer Q. 4.4.

Q. 3. Draw the functional block diagram of 8051 microcontroller.

Ans. Refer Q. 4.6.

Q. 4. Which are the most widely used registers of 8051 microcontroller ?

Ans. Refer Q. 4.9.

Q. 5. Explain I/O ports in 8051 microcontroller.

Ans. Refer Q. 4.11.

Q. 6. Explain pin diagram of 8051.

Ans. Refer Q. 4.13.

Q. 7. What do you mean by addressing mode ? How many types of addressing modes in 8051 ? Give one example of each type.

Ans. Refer Q. 4.18.

☺☺☺

Scanned by TapScanner

5 UNIT

Assembly Programming and Instruction of 8051

CONTENTS

Part-1 : Introduction to 8051 Assembly Programming, Assembling and Running an 8051 Program, Data Types and Assembler Directives ..... 5–2C to 5–7C

Part-2 : Arithmetic, Logic Instructions and Programs, Jump, Loop and Call Instructions ..... 5–7C to 5–11C

Part-3 : I/O Port Programming, Programming 8051 Timers, Serial Port Programming, Interrupts Programming ..... 5–11C to 5–21C

Part-4 : Comparison of Microprocessor, Microcontroller, PIC and ARM Processors and Their Application Areas ..... 5–22C to 5–24C

PART-1

Introduction to 8051 Assembly Programming, Assembling and Running an 8051 Program, Data Types and Assembler Directives.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.1. **Write a short note on assembly language programming. Explain it with an example.**

OR

What do you mean by assembler ? Explain machine language, low level language and high level language.

Answer

A. Machine language :

1. In the early days of the computer, programmers coded programs in machine language. A program that consists of 0s and 1s is called machine language.
2. Although hexadecimal system was used as an efficient way to represent binary numbers, the process of working in machine code was still cumbersome for humans.
3. Eventually, assembly languages were developed that provided mnemonics for the machine code instructions, plus other features that made programming faster and less prone to error.
4. The term mnemonic is frequently used in computer science to refer codes and abbreviations that are relatively easy to remember.

B. Assembler : Assembly language programs must be translated into machine code by a program called an assembler.

C. Low-level language :

1. Assembly language is referred to as low-level language because it deals directly with the internal structure of the CPU.
2. To program in assembly language, the programmer must know all the registers of the CPU and the size of each.

D. High-level language :

1. Today, one can use many different programming languages, such as BASIC, C, C++, Java, and numerous others.

Scanned by TapScanner

2. These languages are called high-level languages because the programmer does not have to be concerned with the internal details of the CPU whereas an assembler is used to translate an assembly language program into machine code (sometimes also called object code or opcode for operation code).
3. High-level languages are translated into machine code by a program called a compiler.

E. Assembly language programming :

1. An assembly language program consists of a series of lines of assembly language instructions.
2. These instructions consist of mnemonic, optionally followed by one or two operands. The operands are the data items being manipulated, and the mnemonics are the commands to the CPU.
3. An assembly language instruction consists :

   [Label :] mnemonic [Operands] [; Comment]

   i. The label field allows the program to refer to a line of code by name. The label field cannot exceed a certain number of characters.

   ii. The assembly language mnemonic (instruction) and operand(s) fields together perform the real work of the program and accomplish the tasks for which the program was written.

   For example :

   ADD *A*, *B*

   MOV *A*, #67

   iii. The comment field begins with semicolon comment indicator ";". Comments may be at the end of a line or on a line by themselves.
4. An assembly language program given below is a series of statements, or lines, which are either assembly language instructions such as ADD and MOV, or statements called directives.
5. Here instructions tell the CPU what to do and directives (also called pseudo-instructions) give directions to the assembler.
6. For example, in the given program the MOV and ADD instructions are commands to the CPU and ORG, END are directives to the assembler.
7. ORG tells the assembler to place the opcode at memory location 0 while END indicates to the assembler the end of the source code. In other words, one is for the start of the program and the other one for the end of the program.

For example :

```
ORG   0 H          ; start (origin) at location 0
MOV   R5, #25 H    ; load 25 H into R5
MOV   R7, #34 H    ; load 34 H into R7
MOV   A, #0        ; load 0 into A
```



```
        ADD   A, R5        ; add contents of R5 to A
                           ; now A = A + R5
        ADD   A, R7        ; add contents of R7 to A
                           ; now A = A + R7
        ADD   A, #12 H     ; add A to value 12 H
                           ; now A = A + 12 H
HERE :  SJMP HERE          ; stay in this loop
        END                ; end of asm source file
```

Que 5.2. How an assembly language program is created, assembled and made ready to run ?

OR

Describe in brief about "asm", "obj" and "lst" file.

Answer

1. The steps that illustrate how an assembly language program is created, assembled and made ready to run is given in the Fig. 5.2.1.

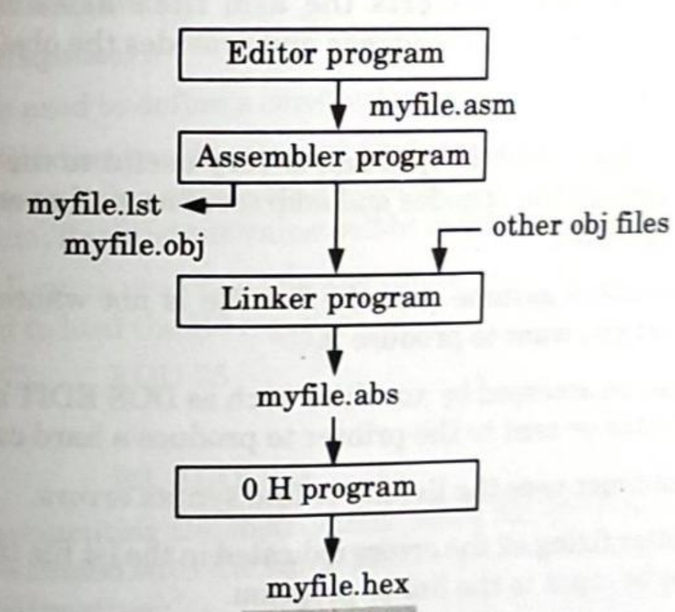


Fig. 5.2.1.

Steps 1 :

i. Firstly an editor is used to create and/ or edit the program. A widely used editor is the MS-DOS edit program (or notepad in windows), which comes with all Microsoft operating systems.

ii. The editor must be able to produce an ASCII file. For many assemblers, the file names follow the usual DOS convention, but the source file has the extension "asm" or "src", depending on which assembler you are using. Check your assembler for the convention.

Scanned by TapScanner

iii. The "asm" extension for the source file is used by an assembler in the next step.

Steps 2 : The "asm" source file containing the program code created in step 1 is fed to an 8051 assembler. The assembler converts the instructions into machine code. The assembler will produce an object file and a list file. The extension for the object file is "obj" while the extension for the list file is "lst".

Steps 3 : Assemblers require a third step called linking. The link program takes one or more object files and produces an absolute object file with the extension "abs". This abs file is used by 8051 trainers that have a monitor program.

Steps 4 : Next the "abs" file is fed into a program called "0 H" (object to hex converter), which creates a file with extension "hex" that is ready to burn into ROM. This program comes with all 8051 assemblers.

asm and obj files :

1. The "asm" file is also called the source file and for this reason some assemblers require that this file have the "src" extension.
2. The 8051 assembler converts the asm file's assembly language instructions into machine language and provides the obj (object) file.

lst (list) file :

1. The lst (list) file, which is optional, is very useful to the programmer because it lists all the opcodes and addresses as well as errors that the assembler detected.
2. Many assemblers assume that the list file is not wanted unless you indicate that you want to produce it.
3. This file can be accessed by an editor such as DOS EDIT and displayed on the monitor or sent to the printer to produce a hard copy.
4. The programmer uses the list file to find syntax errors.
5. It is only after fixing all the errors indicated in the lst file that the obj file is ready to be input to the linker program.

Que 5.3. **How does an instruction differ from directive ? Discuss the different types of assembler directives of 8051. Why are the ORG and END directives also called pseudocode ?**

Answer

A. Difference :

S. No.	Instruction	Directive
1.	An instruction consists of a mnemonic, optionally followed by one or two operands. It basically tells the CPU what to do.	Directives give the directions to the assembler.
2.	The instructions are translated into machine code (opcode) for the CPU to execute.	Directives do not generate any machine code and are used only by the assembler.

B. Types of assembler directive :

i. ORG (Origin) :

1. The ORG directive is used to indicate the beginning of the address. The number that comes after ORG can be either in hex or in decimal.
2. If the number is not followed by H, it is decimal and the assembler will convert it to hex.

ii. EQU (Equate) :

1. This is used to define a constant without occupying a memory location.
2. The EQU directive does not set aside storage for a data item but associates a constant value with a data label so that when the label appears in the program; its constant value will be substituted for the label.
3. The following uses EQU for the counter constant and then the constant is used to load the R3 register.

```
COUNT     EQU 25
...       ....
MOV       R3, # COUNT
```

4. When executing the instruction "MOV R3, #COUNT", the register R3 will be loaded with the value 25.

iii. END directive :

1. Another important pseudocode is the END directive. This indicates to the assembler the end of the source (asm) file.
2. The END directive is the last line of an 8051 program, meaning that in the source code anything after the END directive is ignored by the assembler.

C. Reason :

1. Pseudocodes are the codes that instruct the assembler in doing its job. ORG and END directives are called pseudocodes because they are directives to the assembler.

Scanned by TapScanner

2. ORG tells the assembler to place the opcode at memory location 0 while END indicates to the assembler, the end of the source code.

PART-2

Arithmetic, Logic Instruction and Programs, Jump, Loop and Call Instructions.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.4. **Explain arithmetic instruction with example.**

Answer

Arithmetic instruction : These are the instructions used for arithmetic operation like addition, subtraction etc.

1. **ADD :** This instruction is used to add two operands. The destination operand is always in register *A* while the source operand can be a register, immediate data or in memory.

Format : ADD *A*, Source

For example :

ADD *A*, *R*2 ; Adds contents of *A* and *R*2 and store result in *A*.

ADD *A*, # 20 H ; Add contents of *A* and 20 H and store result in *A*.

2. **ADDC (Add with carry) :** This instruction is used to add two 16-bit operands. In this, add register, immediate data, or in memory to accumulator with carry and store result in *A*.

Format : ADDC *A*, source

For example :

ADDC *A*, *R*2 ; Adds the contents of *A*, *R*2 and carry flag, and stores result in *A*.

ADDC *A*, # 50 H ; Adds the contents of *A* and carry flag and 20 H and stores result.

3. **DA :**

i. The DA (decimal adjust for addition) instruction in the 8051 is provided to correct the aforementioned problem associated with BCD addition. The mnemonic "DA" has as its only operand the accumulator "*A*".

ii. The DA instruction will add 6 to the lower nibble or higher nibble if needed; otherwise, it will leave the result alone.

Format : DA A

For example :

```
MOV  A, #47 H    ; A = 47 H first BCD operand
MOV  B, #25 H    ; B = 25 second BCD operand
ADD  A, B        ; hex (binary) addition (A = 6C H)
DA   A           ; adjust for BCD addition (A = 72 H)
```

4. **SBB (Subtract with borrow) :** This instruction is used for multibyte numbers and will take care of the borrow of the lower operand. If CY = 1 prior to executing the SUBB instruction, it also subtracts 1 from the result.

Format : SUBB *A*, source

For example :

SUBB *A*, *R*3 : Subtracts contents of *R*3 and carry together from *A* and stores result in *A*.

SUBB A, # 20 H : Subtracts 20 H from *A* and storage result in *A*.

5. **MUL :** In this multiplication, one of the operands must be in register *A*, and the second operands must be in register *B*. After multiplication, the result is in the *A* and *B* register; the lower byte is in *A*, and the upper byte is in *B*.

Format : MUL AB

For example :

```
MOV  A, # 25 H   ; load 25 H to reg. A
MOV  B, # 65 H   ; load 65 H to reg. B
MUL  AB          ; 25 H * 65 H = E99 where
                 ; B = 0E H and A = 99 H
```

6. **DIV :** In this, the numerator must be in register A and the denominator must be in B. After the DIV instruction is performed, the quotient is in A and the remainder is in B.

Format : DIV AB

For example :

```
DIV  AB          ; now A = quotient and B = remainder
```

Que 5.5. **Write a program to add two 16-bit numbers the numbers are FC45 H and 02EC H.**

Answer

```
CLR  C           ; make CY = 0
MOV  A, #45 H    ; load the low byte into A
```

Scanned by TapScanner

```
ADD   A, #0EC H   ; add the low byte, now A = 31, CY = 1
MOV   R0, A       ; save the low byte of sum in R0
MOV   A, #02 H    ; load the high byte into A
ADDC  A, #0FC H   ; add the high bytes with carry
                  ; 02 + FC H + 1 = FF H
MOV   R1, A       ; save the high byte of result in R1
```

Finally, we get the result as R0 = 31 H and R1 = FF H

Que 5.6. **What do you mean by logical instructions ?**

Answer

The logical instructions are :

1. **AND :** This instruction will perform a logical AND on the two operands and place the result in the destination. The destination is normally the accumulator. The source operand can be a register, in memory, or immediate.

   Format : ANL destination, source.

   For example :

   ANL A, R2 ;

2. **OR :** The destination and source operands are ORed, and the result is placed in the destination. The ORL instruction can be used to set certain bits of operands to 1. The destination is normally the accumulator. The source operand can be a register, in memory, or immediate.

   Format : ORL destination, source

   For example : ORL C, ACC 7 ;

3. **XOR :** This instruction will perform the XOR operation on the two operands, and place the result in the destination. The destination is normally the accumulator. The source operand can be a register, in memory, or immediate.

   Format : XRL destination, source

   For example : XRL *A*, # 78 H

4. **CPLA (complement accumulator) :** This instruction complements the contents of register A. The complement action changes the 0s to 1s and the 1s to 0s. This is also called 1's complement.

   Format : CPL *A*

   For example : MOV A, # 55H

   CPL A ; now A = AA H

Que 5.7. **Explain the following instructions :**

i. **Jump instructions.**
ii. **Loop instructions.**
iii. **Call instructions.**

Answer

i. **Jump instruction :**

a. **JZ (jump if A = 0) :** In this instruction the content of register *A* is checked. If it is zero, it jumps to the target address.

b. **JNC (Jump if no carry) :**

1. In this instruction, the carry flag bit in the flag (PSW) register is used to make the decision whether to jump.
2. In executing "JNC label", the processor looks at the carry flag to see if it is raised (CY = 1).
3. If it is not, the CPU starts to fetch and execute instructions from the address of the label. If CY = 1, it will not jump but will execute the next instruction below JNC.

c. **LJMP (Long jump) :**

1. LJMP is an unconditional long jump. It is a 3-byte instruction in which the first byte is the opcode, and the second and third bytes represent the 16-bit address of the target location.
2. The 2 byte target address allows a jump to any memory location from 0000 to FFFF H.

d. **SJMP (Short jump) :**

1. In this 2-byte instruction, the first byte is the opcode and the second byte is the relative address of the target location.
2. The relative address range of 00 = FF H is divided into forward and backward jumps; that is, within – 128 to + 127 bytes of memory relative to the address of the current PC.
3. If the jump is forward, the target address can be within a space of 127 bytes from the current PC.
4. If the target address is backward, the target address can be within – 128 bytes from the current PC.

ii. **Loop instructions :**

1. Repeating a sequence of instructions a certain number of times is called a loop. The loop is one of most widely used actions that any microprocessor performs.
2. In the 8051, the loop action is performed by the instruction "DJNZ reg, label".
3. In this instruction, the register is decremented; if it is not zero, it jumps to the target address referred to by the label.

Scanned by TapScanner

4. Prior to the start of the loop the register is loaded with the counter for the number of repetitions.

iii. CALL instruction :

1. Call instruction is used to call a subroutine. Subroutines are often used to perform tasks that need to be performed frequently.
2. In the 8051 there are two instructions for call : LCALL (long call) and ACALL (absolute call).

a. LCALL (Long call) :

1. In this 3-byte instruction, the first byte is the opcode and the second and third bytes are used for the address of the target subroutine.
2. Therefore, LCALL can be used to call subroutines located anywhere within the 64 Kbyte address space of the 8051.

b. ACALL (Absolute call) :

1. ACALL is a 2-byte instruction in contrast to LCALL, which is 3 bytes.
2. Since ACALL is a 2-byte instruction, the target address of the subroutine must be within 2 Kbytes because only 11 bits of the 2 bytes are used for the address.
3. There is no difference between ACALL and LCALL in terms of saving the program counter on the stack or the function of the RET instruction.
4. The only difference is that the target address for LCALL can be anywhere within the 64 Kbytes address space of the 8051 while the target address of ACALL must be within a 2 Kbyte range.

PART-3

I/O Port Programming, Programming 8051 Timers, Serial Port Programming, Interrupts Programming.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.8. **Write a test program for the DS89C420/30 chip to toggle all the bits of P0, P1 and P2 every 1/4 of a second. Assume a crystal frequency of 11.0592 MHz.**

Answer

```
; Tested for the DS89C420/30 with XTAL = 11.0592 MHz.
        ORC     0
BACK:   MOV     A, #55 H
```



```
        MOV     P0, A
        MOV     P1, A
        MOV     P2, A
        ACALL   QSDELAY         ; Quarter of a second delay
        MOV     A, # 0AAH
        MOV     P0, A
        MOV     P1, A
        MOV     P2, A
        ACALL   QSDELAY
        SKMP    BACK
; -------------1/4 SECOND DELAY
QSDELAY :
        MOV     R5, #11
H3 :    MOV     R4, #248
H2 :    MOV     R3, #255
H1 :    DJNZ    R3, H1          ; 4 MC for DS89C4x0
        DJNZ    R4, H2
        DJNZ    R5, H3
        RET
        END
```

Delay = 11 × 248 × 255 × 4 MC × 90 ns = 250,430 μs

Use an oscilloscope to verify the delay size.

Que 5.9. **A switch is connected to pin P1.7. Write a program to check the status of SW and perform the following :**

a. If SW = 0, send letter *N* to P2.

b. If SW = 1, send letter *Y* to P2.

Answer

```
        SETB    P1 . 7          ; make P1.7 an input
AGAIN   JB      P1.2 OVER       ; jump if P1.7 = 1
        MOV     P2, # N         ; SW = 0, issue N to P2
        SJMP    AGAIN           ; Keep monitoring
OVER :  MOV     P2, # Y         ; SW = 1, issue Y to P2
        SJMP    AGAIN           ; keep monitoring
```

Que 5.10. **How many timers do 8051 microcontrollers have ?**

Answer

The 8051 has two timers :

Scanned by TapScanner

A. Timer 0 register :

1. This is a 16-bit register accessed as low byte and high byte.
2. The low byte register is called TL0 (Timer 0 low byte) and high byte register is referred as TH0 (Timer 0 high byte).

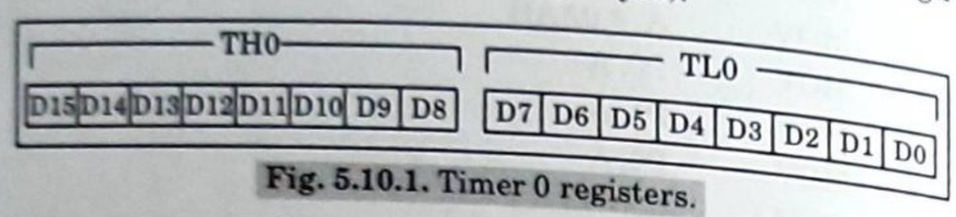


Fig. 5.10.1. Timer 0 registers.

3. These registers can be accessed like any other register such as *A*, *B*, R0, R1, R2 etc.

For example :

MOV TL0, # 4F H ; Moves the value 4F H into TL0, the low byte of timer 0.

MOV R5, TH0 ; Saves the TH0 (high byte of timer 0) in R5.

B. Timer 1 register :

1. It is also 16-bit register which is also accessed as low byte and high byte.
2. The low byte register is called TL1 (Timer 1 low byte) and high byte register is called TH1 (Timer 1 high byte).
3. These registers are accessible in same way as the registers of Timer 0.

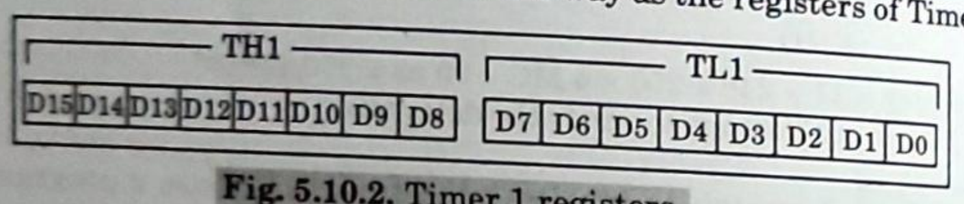


Fig. 5.10.2. Timer 1 registers.

Que 5.11. Explain the function of each bit of TMOD register of 8051.

Answer

1. TMOD register is a timer mode register and both timers (0 and 1) use this register to set the various timer operation modes.
2. It is an 8-bit register in which lower 4-bits are for Timer 0 and upper 4-bits for Timer 1. In each case, the lower two bits are used to set the timer mode and upper two bits are used to specify the operation.

MSB							LSB
GATE	C/T	M1	M0	GATE	C/T	M1	M0
Timer 1				Timer 0			

Fig. 5.11.1. TMOD register.

Table 5.11.1.

M1	M0	Mode	Operating Mode
0	0	0	13-bit timer mode 8-bit timer/counter THx with TLx as 5-bit prescaler
0	1	1	16-bit timer mode 16-bit timer/counters THx and TLx are cascaded; there is no prescaler
1	0	2	8-bit auto reload 8-bit auto reload timer/counter; THx holds a value that is to be reloaded into TLx each time it overflows.
1	1	3	Split timer mode

M1 and M0 :

1. M0 and M1 select the timer mode. There are basically three modes : 0, 1 and 2.
2. Mode 0 is a 13-bit timer, mode 1 is a 16-bit timer and mode 2 is an 8-bit timer. The most widely used modes are 1 and 2.

C/T (Clock/Timer) :

1. This bit in the TMOD register is used to decide whether the timer is used as a delay generator or an event counter.
2. If C/T = 0, it is used as a timer for time delay generation and the crystal frequency attached to 8051 act as source of the clock for the timer.
3. This means that the size of the crystal frequency attached to 8051 also decides the speed at which 8051 timer ticks.
4. The frequency for the timer is always $1/12^{th}$ the frequency of the crystal attached to the 8051.

GATE :

1. In the TMOD register both timers 0 and 1 have the GATE bit. It is because every timer has a means of starting and stopping.
2. Some timers do this by software, some by hardware, and some have both software and hardware controls. The timers in the 8051 have both.
3. The start and stop of the timer are controlled by way of software by the TR (timer start) bit TR0 and TR1.
4. This is achieved by the instructions "SETB TR1" and "CLR TR1" for Timer 1, and "SETB TR0" and "CLR TR0" for Timer 0.
5. The SETB instruction starts it, and it is stopped by the CLR instruction. These instructions start and stop the timers as long as GATE = 0 in the TMOD register.

Scanned by TapScanner

6. The hardware way of starting and stopping the timer by an external source is achieved by making GATE = 1 in the TMOD register.

Que 5.12. Explain the structure of TCON register.

Answer

Structure of TCON register :

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Fig. 5.12.1.

Table 5.12.1.

Bit	Symbol	Function
7	TF1	Timer 1 overflow flag. Set when timer rolls from all 1s to 0. Cleared when processor vectors to execute interrupt service routine located at program address 001B H.
6	TR1	Timer 1 run control bit. Set to 1 by program to enable timer to count; cleared to 0 by program to halt timer. Does not reset timer.
5	TF0	Timer 0 overflow flag. Set when timer rolls from all 1s to 0. Cleared when processor vectors to execute interrupt service routine located at program address 000B H.
4	TR0	Timer 0 run control bit. Set to 1 by program to enable timer to count; cleared to 0 by program to halt timer. Does not reset timer.
3	IE1	External interrupt 1 edge flag. Set to 1 when a high-to-low edge signal is received on port 3 pin 3.3 ($\overline{INT1}$). Cleared when processor vectors to interrupt service routine located at program address 0013 H. Not related to timer operations.
2	IT1	External interrupt 1 signal type control bit. Set to 1 by program to enable external interrupt 1 to be triggered by a falling edge signal. Set to 0 by program to enable a low-level signal on external interrupt 1 to generate an interrupt.

1	IE0	External interrupt 0 edge flag. Set to 1 when a high-to-low edge signal is received on port 3 pin 3.2 ($\overline{INT0}$). Cleared when processor vectors to interrupt service routine located at program address 0003H. Not related to timer operations.
0	IT0	External interrupt 0 signal type control bit. Set to 1 by program to enable external interrupt 0 to be triggered by a falling edge signal. Set to 0 by program to enable a low-level signal on external interrupt 0 to generate an interrupt.

Table 5.12.2. Equivalent instructions for the Timer control register (TCON)

For Timer 0				
SETB	TR0	=	SETB	TCON.4
CLR	TR0	=	CLR	TCON.4
SETB	TF0	=	SETB	TCON.5
CLR	TF0	=	CLR	TCON.5
For Timer 1				
SETB	TR1	=	SETB	TCON.6
CLR	TR1	=	CLR	TCON.6
SETB	TF1	=	SETB	TCON.7
CLR	TF1	=	CLR	TCON.7

Que 5.13. Timer-0 of 8051 microcontroller is to be programmed in mode-1 for creating a square wave of duty cycle 50 % on the port P1.5. Write an algorithm for programming the counter.

Answer

A. Program :

```
        MOV     TMOD, #01       ; Timer 0, mode 1(16-bit mode)
HERE :  MOV     TL0, #0F2 H     ; TL0 = F2 H, the low byte
        MOV     TH0, #0FF H     ; TH0 = FF H, the high byte
        CPL     P1.5            ; toggle P1.5
        ACALL   DELAY
```

Scanned by TapScanner

```
        SJMP    HERE            ; load TH, TL Again
Delay   using Timer 0
        DELAY
        SET B   TR0             ; start Timer 0
AGAIN   JNB     TF0, AGAIN      ; Monitor Timer 0 flag until
                                ; it roll over
        CLR     TR0             ; stop timer 0
        CLR     TF0             ; clear timer 0 flag
        RET
```

B. Algorithm :

1. TMOD is loaded.
2. FFF2 H is loaded into TH0 - TL0.
3. P1.5 is toggled for the high and low portions of the pulse.
4. The DELAY subroutine using the timer is called.
5. In the DELAY subroutine, Timer 0 is started by the SETB TRO instruction.

i. Timer 0 counts up with the passing of each clock, which is provided by the crystal oscillator. As the timer counts up, it goes through the states of FFF3, FFF4, FFF5, FFF6, FFF7, FFF8, FFF9, FFFA, FFFB, and so on until it reaches FFFF H.

ii. One more clock rolls it to 0, raising the timer flag (TF0 = 1). At that point, the JNB instruction falls through.

iii. Timer 0 is stopped by the instruction CLR TR0. The DELAY subroutine ends and the process is repeated.

Que 5.14. **Explain how serial communication takes place in 8051. State special function registers used for serial communication in 8051 microcontroller.**

Answer

1. The serial port of 8051 is full duplex, means it can transmit and receive simultaneously. It uses register SBUF to hold data. Register SCON controls data communication, register-PCON controls data rates and pin RxD (P3.0) and TxD (P3.1) do the data transfer.
2. SBUF is an 8-bit register dedicated for serial communication in 8051. Its address is 99 H. It can be addressed like any other register in 8051.
3. Writing to SBUF loads data to be transmitted and reading SBUF accesses received data.
4. There are two separate and distinct register, the transmit write-only register, and the receive read-only register. This is shown in Fig. 5.14.1.
5. The way in which SBUF is used for the transmission and reception of the data during serial communication is explained below :

a. **Transmission :** When a byte of data is to be transmitted via the TxD pin, the SBUF is loaded with this data byte. As soon as a data byte is written into SBUF, it is framed with the start and stop bits and transmitted serially via the TxD pin.

b. **Reception :** When 8051 receives data serially via RxD pin of it, the 8051 deframes it. The start and stop bits are separated out from a byte of data. This type is placed SBUF register.

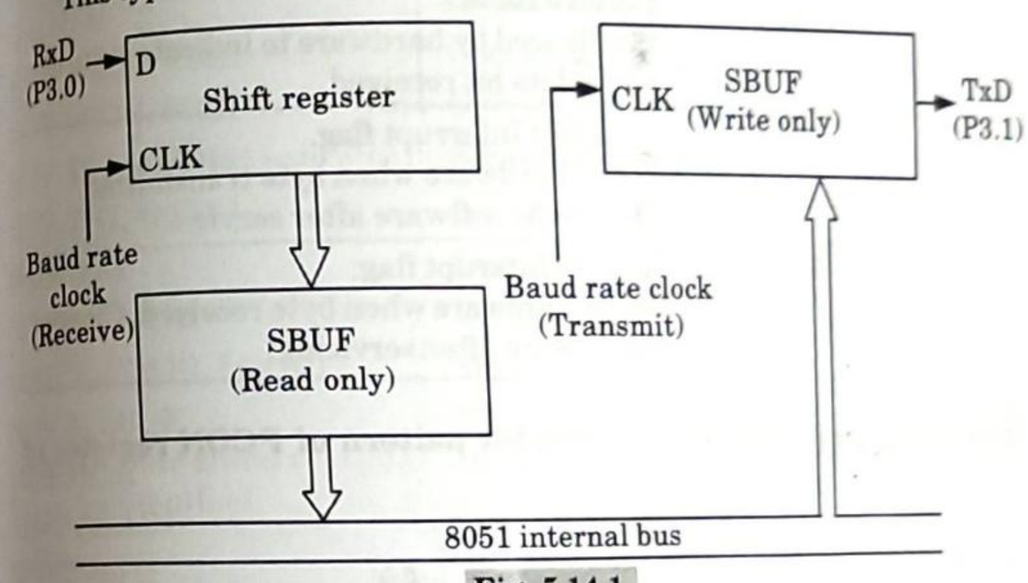


Fig. 5.14.1.

Que 5.15. **Explain the bit pattern of SCON register of 8051.**

Answer

1. The 8051 provides four programmable modes for serial data communication. A particular mode can be selected by setting the SM0 and SM1 bits in SCON. The mode selection also decides the baud rate.
2. The Fig. 5.15.1 shows the bit patterns for SCON.

7	6	5	4	3	2	1	0
SM0	SM1	SM2	REN	TB8	RB8	T1	RI

Fig. 5.15.1.

Symbol	Position	Name and Significance
SM0	SCON.7	Serial port mode control bit 0. Set/cleared by software.
SM1	SCON.6	Serial port mode control bit 1. Set/cleared by software.
SM2	SCON.5	Serial port mode control bit 2. Set by software to disable reception of frame for which bit 8 is zero.

Scanned by TapScanner

REN	SCON.4	Receiver enable control bit. Set/cleared by software to enable/disable serial data reception.
TB8	SCON.3	Transmit Bit 8. Set cleared by hardware to determine state of ninth data bit transmitted in 9-bit UART mode.
RB8	SCON.2	Receive Bit 8. Set/cleared by hardware to indicate state of ninth data bit received
TI	SON.1	Transmit Interrupt flag. Set by hardware when byte transmitted. Cleared by software after servicing.
RI	SCON.0	Receive Interupt flag. Set by hardware when byte received. Cleared by software after servicing.

Que 5.16. **Draw and explain the bit pattern of PCON register of 8051.**

Answer

Bit pattern of PCON register :

7	6	5	4	3	2	1	0
SMOD	-	-	-	GF1	GF0	PD	IDL

Fig. 5.16.1.

Symbol	Position	Name and Significance
SMOD	PCON.7	Serial baud rate modify bit. It is 0 at reset. It is set to 1 by program to double the baud rate.
–	PCON.6	Not defined.
–	PCON.5	Not defined.
–	PCON.4	Not defined.
GF1	PCON.3	General purpose user flag bit 1. Set/cleared by program.
GF0	PCON.2	General purpose user flag bit 0. Set/cleared by program.
PD	PCON.1	Power down bit. It is set to 1 by program to enter power down configuration for CHMOS microcontrollers.
IDL	PCON.0	Idle mode bit. It is set to 1 by program to enter idle mode configuration for CHMOS microcontrollers.

Que 5.17. **Write a program to initialise 8051 serial port to operate in mode 1 for receiving a serial byte through RXD pin, send the received data to port 2 and operate timer 1 in auto reload mode.**

Answer

```
CLR IE.4            ; Disable serial port interrupt
MOV TMOD, #20H      ; Choose, mode 2 operation for timer 1
MOV SCON, #50H      ; Operate serial port in mode 1
MOV TL1, #0FDH      ; Set the initial value
MOV TH1, #0FDH      ; Set the reset value
SETB TR1            ; Start timer 1
AGAIN: CLR RI       ; Clear RI bit
LOOP1: JNB RI, LOOP1 ; Wait for last bit to receive
MOV A, SBUF         ; Save incoming data in A
MOV P2, A           ; Send to port 2
SJMP AGAIN          ; Keep getting data
```

Que 5.18. **How many interrupts in 8051 ? Also draw interrupt vector table for the 8051.**

Answer

The six interrupts in the 8051 shown in Table 5.18.1 :

Table 5.18.1 : Interrupt vector table

S. No.	Interrupt	ROM Location (Hex)	Pin
1.	Reset	0000	9
2.	External hardware interrupt 0 (INT0)	0003	P3.2(12)
3.	Timer 0 interrupt (TF0)	000B	
4.	External hardware interrupt 1 (INT1)	0013	P3.3(13)
5.	Timer 1 interrupt (TF1)	001B	
6.	Serial COM interrupt (RI and TI)	0023	

1. Reset : When the reset pin is activated, the 8051 jumps to address location 0000.

Scanned by TapScanner

2. Two interrupts are set aside for the timers : one for timer 0 and one for timer 1. Memory locations 000B H and 001B H in the interrupt vector table belong to timer 0 and timer 1, respectively.
3. Two interrupts are set aside for hardware external hardware interrupts. Pin numbers 12 (P3.2) and 13(P3.3) in port 3 are for the external hardware interrupts INT0 and INT1 respectively. These external interrupts are also referred to as EX1 and EX2.
4. Serial communication has a single interrupt that belongs to both receive and transmit. The interrupt vector table location 0023H belongs to this interrupt.

Que 5.19. **Write a program in which 10 bytes of data stored in RAM locations starting from 45H are transferred serially. At the end of data transfer, the value of R0 (*i.e.*, 0) is displayed on P1.**

Answer

```
        ORG   0000H
        LJMP  MAIN
        ORG   0023H          ; jump to ISR for serial transmission
        LJMP  SERIAL
        ORG   003 0H
MAIN:   MOV   TMOD,#2 0H     ; timer 1 in mode 2
        MOV   TH1,#-6        ; set baud rate
        MOV   SCON, #50H     ; 8 bit, 1 stop, REN enabled
        MOV   IE,#90H        ; serial port interrupt enable
        SETB  TR1            ; start timer 1
        MOV   R0, #10        ; counter for number of bytes
        MOV   R1, #45H       ; R1 is the pointer to RAM
BACK:   MOV   A,@R1          ; move data from RAM to A
        MOV   SBUF, A        ; data to be transmitted is loaded into
                               SBUF
        DJNZ  R0,BACK        ; repeat till all data is sent
HERE:   SJMP  HERE
;.....................serial port ISR
SERIAL: JNB   TI, RECE       ; if TI is not high, it implies reception
        MOV   A, R0          ; if TI is high, move value of R0 into A
        MOV   P1, A          ; transfer it to P1
        CLR   TI             ; clear TI for next transmission
        RETI
RECE:   MOV   A, SBUF        ; if reception, move received data to SBUF
        CLR   RI             ; clear RI to enable next reception
        RETI
        END
```


PART-4

Comparison of Microprocessor, Microcontroller, PIC and ARM Processors and Their Application Areas.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.20. **Give the comparison of microprocessor and microcontroller.**

Answer

Comparison of microprocessor and microcontroller :

S.No.	Microprocessor	Microcontroller
1.	Microprocessor is complete functional CPU, *i.e.*, it contains ALU, registers, control unit and interrupt processing circuits.	Microcontroller is complete functional microcomputer, *i.e.*, it contains the circuitry of microprocessor with memory (ROM, RAM), I/O circuit and peripherals.
2.	Microprocessor instructions are mainly nibble or byte addressable.	Microcontroller instructions are both bit as well as byte addressable.
3.	Access time for external memory and I/O devices are more, resulting in a slower system.	Access time for on-chip memory and I/O devices are less, resulting in faster system.
4.	It is used for general purpose application.	It is used for special purpose application.
5.	Software protection is not possible because of the requirement of external code memory.	Software protection is possible because of on-chip code memory.
6.	Expensive.	Less expensive.

Que 5.21. **Write the comparison of PIC and ARM processors.**

Scanned by TapScanner

Answer

Comparison of PIC and ARM controller :

S. No.	Feature	PIC	ARM
1.	Device type	Complete microcontroller	Processor core
2.	Architecture	Harvard architecture	Von Neumann (ARM7) and modified Harvard architecture (Higher versions)
3.	Bus width	8/16/32-bit	32 / 64-bit.
4.	Registers	No separate registers. Data RAM memory is treated as registers.	32-bit registers
5.	Speed	4 Clock /Instruction cycle	1 Clock / Instruction cycle
6.	ISA (Instruction Set Architecture)	RISC (Reduced instruction set computers)	RISC (Reduced instruction set computers)
7.	Instruction execution	Sequential	Pipelining
8.	Memory	Internal data and program memory.	No internal memory, only memory interface.
9.	Memory address space	Small memory address space necessary for small applications.	Large memory address space needed for high end applications.

Que 5.22. Give the applications of microprocessor, microcontroller, PIC and ARM processor.

Answer

A. **Application of microprocessor :**
1. The microprocessor is used in personal computers (PCs).
2. The microprocessor is used in LASER printers for good speed and making automatic photo copies.
3. The microprocessors are used in modems, telephone, digital telephone sets, and also in air reservation systems and railway reservation systems.
4. The microprocessor is used in medical instrument to measure temperature and blood pressure.
5. It is also used in mobile phones and television.

6. It is used in calculators and game machine.

B. **Applications of microcontroller :**
1. The microcontroller is used in microwave oven.
2. It is used in barcode readers.
3. It is used in hotel card key writers.
4. It is used in automotive including cruise control, ignition system and radiator fan.

C. **Application of PIC :**
1. The PIC is used in peripherals devices.
2. It is used in audio accessories. 3. It is used in video games.

D. **Application of ARM processor :** It is used in portable devices like digital cameras, mobile phones, home networking modules and wireless communication technologies and other embedded systems.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. Write a short note on assembly language programming. Explain it with an example.

Ans. Refer Q. 5.1.

Q. 2. How does an instruction differ from directive ? Discuss the different types of assembler directives of 8051. Why are the ORG and END directives also called pseudocode ?

Ans. Refer Q. 5.3.

Q. 3. Explain the following instructions :
i. Jump instructions. ii. Loop instructions.
iii. Call instructions.

Ans. Refer Q. 5.7.

Q. 4. How many timers do 8051 microcontrollers have ?

Ans. Refer Q. 5.10.

Q. 5. Explain how serial communication takes place in 8051. State special function registers used for serial communication in 8051 microcontroller.

Ans. Refer Q. 5.14.

Q. 6. How many interrupts in 8051 ? Also draw interrupt vector table for the 8051.

Ans. Refer Q. 5.18.

Q. 7. Write the comparison of PIC and ARM processors.

Ans. Refer Q. 5.21.

☺☺☺

Scanned by TapScanner

1 UNIT

Introduction to Microprocessor (2 Marks Questions)

1.1. What is microprocessor ? Give the power supply and clock frequency of 8085. **AKTU 2015-16, Marks 02**

Ans. **Microprocessor :** The microprocessor is a programmable integrated device that has computing and decision making capability similar to that of the central processing unit (CPU) of a computer.

Power supply = +5 V

Clock frequency = 3 MHz

1.2. Write basic operations of microprocessor with block diagram. **AKTU 2017-18, Marks 02**

Ans.

1. Using its ALU (Arithmetic Logic Unit), a microprocessor can perform mathematical operations.
2. A microprocessor can move data from one memory location to another.
3. A microprocessor can make decisions and jump to a new set of instructions based on those decisions.

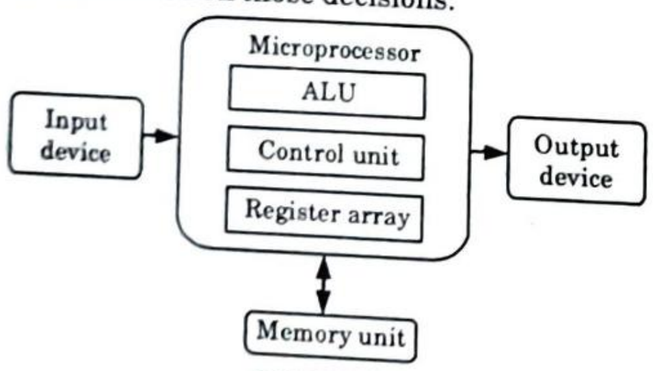


Fig. 1.2.1.

1.3. What is flag ? Discuss in brief. **AKTU 2016-17, Marks 02**

OR

Draw flag register of 8085. **AKTU 2017-18, Marks 02**

Ans. It is an 8-bit register, in which five of the bits carry significant information in the form of flags : S (Sign flag), Z (Zero flag), AC (Auxiliary Carry flag), P (Parity flag) and CY (Carry flag) as shown in Fig. 1.3.1.

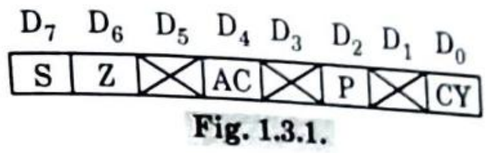


Fig. 1.3.1.

1.4. Logic calculations are done in which type of registers ? Describe with neat diagram. **AKTU 2016-17, Marks 02**

Ans. Accumulator register is only the register in which arithmetical and logical operation are done.

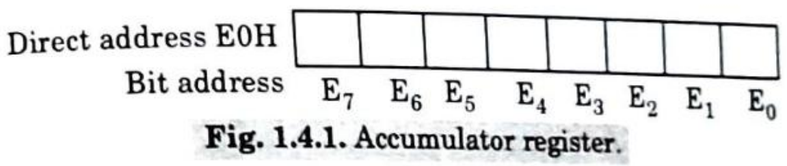


Fig. 1.4.1. Accumulator register.

1.5. What is stack memory ? Explain. **AKTU 2016-17, Marks 02**

Ans. The memory which is used to store useful data of different registers in the form of stack of data is called stack memory.

1.6. Which segment is used to store interrupt and subroutine return address register ? **AKTU 2016-17, Marks 02**

Ans. Stack segment is used to store interrupt and subroutine return address register.

1.7. What is meant by MAR and MDR ? **AKTU 2018-19, Marks 02**

Ans. **MAR stand for Memory Address Register :**
This register holds the memory addresses of data and instructions. This register is used to access data and instructions from memory during the execution phase of an instruction.

MDR stand for Memory Data Register :
MDR is the register of a computer's control unit that contains the data to be stored in the computer storage or the data after a fetch from the computer storage.

1.8. What is the fundamental difference between Harvard and Von-Neumann architecture ? **AKTU 2018-19, Marks 02**

Scanned by TapScanner

Ans.

S. No.	Harvard architecture	Von-Neumann architecture
1.	It uses separate memories for program and data with their independent address and data buses.	Programs and data share the same memory space.
2.	There is no need to have any time division multiplexing of address and data bus.	Program and data fetches are done using time division multiplexing.
3.	Example : MCS-51 Microcontroller	Example : Motorola 68HC11

1.9. Explain different types of interrupts in 8085.

AKTU 2017-18, Marks 02

Ans.

A. **Hardware interrupts :** In hardware interrupts MPU pins are used to receive the interrupts request from peripheral devices.
Types : TRAP, RST 7.5, RST 6.5, RST 5.5, INTR.

B. **Software interrupts :** In software interrupts execution of an instruction is done to interrupt the program.
Types : RST 0 to RST 7.

1.10. Define instruction cycle, machine cycle and *T*-state in microprocessor operation. **AKTU 2015-16, Marks 02**

OR

How instruction cycle, machine cycle, and clock cycle are related ? **AKTU 2018-19, Marks 02**

Ans.

i. **Instruction cycle :** It is defined as the time required to complete the execution of an instruction.

ii. **Machine cycle :** It is defined as the time required to complete one operation of accessing memory, or acknowledging or external request.

iii. **T-state :** It is defined as one subdivision of the operation performed in one clock period.

iv. **Clock cycle :** Each microprocessor needs a clock signal for its operation. The time period of this clock signal is called clock cycle.

1.11. How does the microprocessor differentiate between data and instruction ? **AKTU 2015-16, Marks 02**

Ans. When the first machine code of an instruction is fetched and decoded in the instruction register the microprocessor recognizes the number

of bytes required to fetch the entire instruction. For example MVI A, Data, the second byte is always considered as data. If the data byte is omitted by mistake whatever is in that memory location will be considered as data and the byte after the data will be treated as the next instruction.

1.12. Compare RET and POP instructions in microprocessor.

AKTU 2015-16, Marks 02

Ans.

S. No.	RET Instruction	POP Instruction
1.	This instruction pops the return address from the stack and loads program counter with this return address.	This instruction copies the contents of memory location pointed by the stack pointer into the lower byte of specified register pair and increment the stack pointer by one.
2.	It transfers program control to the instruction next to CALL in the main program.	It copies the contents of memory location pointed by stack pointer into higher bytes of specified register pair and increment the stack pointer again by one.

1.13. Calculate the execution time for the following code using 8085 operated at 3 MHz clock frequency.
MVI B, 37H
HLT **AKTU 2015-16, Marks 02**

Ans. Total number of *T*-states = 7 + 5 = 12

Execution time = $\frac{1}{3 \times 10^6} \times 12 = 4\ \mu s.$

1.14. What is SIM and RIM instruction ? Explain with example.

AKTU 2016-17, Marks 02

Ans.

A. **SIM :** This instruction makes the interrupts as desired. It also sends out serial data through the SOD pin. For this instruction command byte must be loaded in the accumulator.
Example : A = 0EH
SIM instruction will mask RST 7.5 and RST 6.5 interrupts where as RST 5.5 interrupt will be unmasked. It will also disable serial output.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
SOD	SOE	X	RST7.5	MSE	M7.5	M6.5	M5.5	Register A
0	0	0	0	1	1	1	0	= 0EH

Scanned by TapScanner

B. **RIM** : This instruction copies the status of the interrupts into the accumulator. It also reads the serial data through the SID pin.

Example : After execution of RIM instruction if the contents of accumulator are 4BH then we get following information, *i.e.*,

a. RST 7.5 is pending
b. RST 6.5 and RST 6.5 are masked
c. Interrupt enable flip flop is set
d. Serial input data is zero

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
SID	I7.5	I6.5	I5.5	IE	M7.5	M6.5	M5.5	Register A
0	1	0	0	1	0	1	1	= 4BH

1.15. What is the position of the stack pointer after the POP instruction ? Discuss it. **AKTU 2016-17, Marks 02**

Ans. After every POP instruction, the stack pointer decreased by one.

1.16. What are interfacing logical devices ? **AKTU 2017-18, Marks 02**

Ans. To interconnect peripherals with the microprocessor interfacing logic devices, are necessary. These circuits include devices such as buffers, decoders, encoders and latches.

1.17. What is meant by memory mapped I/O ? **AKTU 2018-19, Marks 02**

Ans. Memory mapped I/O is an interfacing technique in which memory related instructions are used for data transfer and the device is identified by a 16 bit address. Here, the I/O devices are treated as memory locations.

1.18. List advantages of memory-mapped I/O and I/O mapped I/O technique of data transfer in microprocessor. **AKTU 2015-16, Marks 02**

Ans. **Memory mapped I/O :**

1. All arithmetic and logical operation can be performed on I/O data directly.
2. To map with other I/O operations, no separate operation required.

I/O mapped I/O :

1. Less logic is needed to decode a discrete address.
2. Whole memory address space is available.

2 UNIT

Intel 8086 Microprocessor (2 Marks Questions)

2.1. Differentiate between 8085 and 8086 microprocessor.

Ans.

S. No.	8085	8086
1.	8-bit microprocessor	16-bit microprocessor
2.	16 address lines	20 address lines
3.	It does not support pipelining	It supports pipelining

2.2. List the functional units of 8086.

Ans. i. Bus interface unit (BIU) ii. Execution unit (EU)

2.3. Give the conditions in which internal queue will be flushed out.

Ans. When 8086 microprocessor changes mode from real mode to protected mode and vice-versa, the internal queue has to be flushed.

2.4. Why there are two ground pins in 8086 ? Explain. **AKTU 2016-17, Marks 02**

Ans. There are two ground pins in 8086 to distribute power for noise reduction.

2.5. What is the role of execution unit of 8086 microprocessor ?

Ans. The execution unit of 8086 tells the BIU from where to fetch instruction or data, decodes instruction and executes instructions.

2.6. What are general purpose registers in 8086 microprocessor ? **AKTU 2016-17, Marks 02**

Ans. The registers AX, BX, CX and DX are the general purpose registers of 16 bits.

Scanned by TapScanner

2.7. List the types of segment registers.

Ans.

i. Code segment registers
ii. Extra segment registers
iii. Data segment registers
iv. Stack segment registers

2.8. Give the significance of MN/$\overline{MX}$ signal of 8086.

Ans. This pin decides whether 8086 is to be operated in minimum mode or maximum mode.

2.9. Explain the need of memory segmentation in 8086.

AKTU 2015-16, Marks 02

Ans.

i. It allows the memory addressing capacity to be 1 MB even though the address associated with individual instruction is only 16-bit.
ii. If facilitates use of separate memory areas for program data and stack.

2.10. How many types of interrupts in 8086 ? Give its significance.

AKTU 2016-17, Marks 02

Ans.

A. **Interrupts :** Types of 8086 interrupts :
i. Divide by zero interrupt (Type 0)
ii. Single step interrupt (Type 1)
iii. Non-maskable interrupt (Type 2)
iv. Break interrupt (Type 3)
v. Overflow interrupt (Type 4)
vi. Software interrupt (Type 0 – 255)
vii. Maskable interrupt (INTR)

B. **Significance :** Interrupts are useful when interfacing I/O device at relatively low data transfer rates, such as keyboard inputs.

☺☺☺

3 UNIT

Peripheral Devices (2 Marks Questions)

3.1. What are peripheral devices ? Why are they required ?

AKTU 2018-19, Marks 02

Ans.

A. **Peripheral devices :** A peripheral device is defined as a computer device, such as a keyboard or printer that is not part of the essential computer (such as the memory and microprocessor).

B. **Required :** Peripheral devices are required to enable microprocessor to communicate with the user or an external process and to ease the circuit operations.

3.2. What do you mean by general purpose peripherals ?

Ans. These are the devices that perform a specific task but may be used for interfacing a variety of input/output devices to the microprocessor.

3.3. What is DMA ?

Ans. DMA (Direct Memory Access) is an I/O technique commonly used for high speed data transfer.

3.4. Explain the role of READY in 8237 DMA controller.

Ans. READY is an active high input used to match the read or write speed of 8237 with slow memories or I/O devices.

3.5. List any two salient features of mode 2 in 8255.

Ans.

i. The 8-bit port is bidirectional and additionally a 5-bit control port is available.
ii. Inputs and outputs are both latched.

3.6. List the available addressing modes in 8085.

AKTU 2018-19, Marks 02

Ans.

1. Immediate addressing mode.
2. Register addressing mode.
3. Direct addressing mode.
4. Indirect addressing mode.
5. Implied or implicit addressing mode.

3.7. Specify the memory addressing capacity of 8085 microprocessor. How many address lines are required to address 2 MB memory ?

AKTU 2015-16, Marks 02

Scanned by TapScanner

Ans. Memory addressing capacity of 8085 = 64 kB.
Number of address lines required to address 2 MB memory = 21.

3.8. Define compiler or interpreter in programming languages.

AKTU 2017-18, Marks 02

Ans. A compiler is a computer program or set of programs that transforms source code written in a programming language into another computer language.

3.9. Define following :

i. Nibble ii. Word

AKTU 2017-18, Marks 02

Ans.

i. **Nibble :** Nibble is four binary digits or half of an eight-bit byte.
ii. **Word :** Group of bytes is known as words. A word consists of 2 or more bytes.

3.10. Define following :

i. Mnemonics ii. Program

AKTU 2017-18, Marks 02

Ans.

i. **Mnemonics :** Each manufacturer of a microprocessor has devised a symbolic code for each instruction, called a mnemonic.
ii. **Program :** A program is a specific set of ordered operations for a computer to perform.

3.11. What is meant by cross-compiler ?

AKTU 2016-17, Marks 02

Ans. A complier which can convert instructions into machine code or low level code for a computer other than that on which it can run is called cross-compiler.

3.12. Write about different languages of digital computer.

AKTU 2017-18, Marks 02

Ans.

A. **High level languages :** Programming languages that are intended to be machine independent are known as high level languages.
B. **Low level languages :**
1. **Machine Language :** The number of bits in a word for a given machine is fixed and words are formed through various combinations of these bits.
2. **Assembly Language :** Each manufacturer of a microprocessor has devised a symbolic code for each instruction, called a mnemonic. The mnemonic for a particular instruction consists of letters that suggest the operation to be performed by that instruction.

☺☺☺

4 UNIT

8051 Microcontrollers Basics (2 Marks Questions)

4.1. Define microcontroller.

Ans. **Microcontroller :** A device that includes microprocessor, memory, and I/O signal lines on a single chip, fabricated using VLSI technology.

4.2. What do you mean by embedded system ?

Ans.

1. An embedded system is an electronic/electromechanical system designed to perform a specific function and is a combination of both hardware and firmware (software).
2. Each embedded system is unique, and the hardware as well as the software is highly specialized to the application domain.

4.3. What are the main application areas of embedded system ?

Ans. The main application areas are :

1. Consumer electronics : camcorders, cameras etc.
2. Household appliances : T.V., fridge etc.
3. Telecom
4. Healthcare
5. Computer peripherals.

4.4. Give the features of 8051 microcontroller.

Ans. **Main features of 8051 :**

1. 32 Input / Output pins arranged as four ports : PO–P3.
2. Two 16-bit timers / counters : T0 and T11.
3. 16-bit program counter (PC) and data pointer.

4.5. What are flags available in 8051 ?

Ans. **The flags available in 8051 are :** CY (Carry flag), AC (Auxiliary carry flag), OV (Over flow flag) and P (Parity flag)

4.6. Give the comparison of 8051 family members.

Scanned by TapScanner

Ans.

Feature	8051	8052	8031
ROM (on-chip program space in bytes)	4K	8K	0K
RAM (bytes)	128	256	128
Timers	2	3	2
I/O pins	32	32	32
Serial port	1	1	1
Interrupt sources	6	8	6

4.7. Explain the function of the $\overline{PSEN}$ pin of 8051.

Ans. This is an output pin. $\overline{PSEN}$ stands for program store enable. This is an active low output pin. It acts as a strobe to read the external program memory.

4.8. What do you mean by SFR ?

Ans.

1. The 8051 operations that do not use the internal 128-bytes RAM addresses from 00 H to 7F H are done by a group of specific internal register, each called a Special Functions Register (SFR).
2. In 8051, register *A*, *B*, PSW and DPTR are a part of the group of registers referred as SFR.

4.9. What is the size of the program counter register ? What does the program counter do ?

Ans. The size of program counter is 20 bit. A program counter is a register that contains the address of the instruction being executed at the current time.

4.10. What do you mean by addressing modes ?

Ans. The CPU can access data in many ways. The data could be in a register or in memory or can be provided as an immediate value. Thus, the various ways of accessing the data are called addressing modes.

4.11. Name the various types of addressing modes used in 8051 microcontroller.

Ans. The 8051 consists of 5 addressing modes :

1. Immediate.
2. Register addressing mode.
3. Register indirect addressing mode.
4. Indexed addressing mode.
5. Direct addressing mode.

4.12. What is the advantage of register indirect addressing mode over direct addressing mode ?

Ans. Advantage of register indirect addressing mode is that it makes accessing data dynamic rather than static as in the case of direct addressing mode.

☺☺☺

5 UNIT Assembly Programming and Instruction of 8051 (2 Marks Questions)

5.1. What do you mean by assembly language ?

Ans. An assembly language program consist a series of lines of assembly language instructions. These instructions consist of mnemonic, optionally followed by one or two operands. The operands are the data items being manipulated, and mnemonics are the commands to CPU.

5.2. List the different assembler directives.

Ans. The following are the widely used 8051 assembler directives :

i. ORG (Origin). ii. EQU (Equate).
iii. END. iv. DB (Define byte).

5.3. How does an instruction differ from directive ?

Ans.

S. No.	Instruction	Directive
1.	An instruction consists of a mnemonic, optionally followed by one or two operands. It basically tells the CPU what to do.	Directives give the directions to the assembler.
2.	The instructions are translated into machine code (opcode) for the CPU to execute.	Directives do not generate any machine code and are used only by the assembler.

5.4. What do you mean by END directive ?

Ans. END directive indicates to the assembler about the end of the source (asm) file. The END directive is the last line of an 8051 program, meaning that in the source code anything after the END directive is ignored by the assembler.

5.5. Why are the ORG and END directives also called pseudocode ?

Ans.

1. ORG and END directives are called pseudocodes because they are directives to the assembler.

Scanned by TapScanner

2. ORG tells the assembler to place the opcode at memory location 0 while END indicates to the assembler the end of source code.

5.6. How many timers are there in 8051 ?

Ans. There are two timers in 8051 :

1. Timer 0 2. Timer 1

5.7. Write the interrupts in 8051 microcontroller.

Ans.

1. Timer 0 (TFO)
2. Timer 1 (TF1)
3. Serial com interrupt (RI and TI)
4. INT0
5. INT1
6. Reset

5.8. Draw the structure of TMOD register.

Ans.

MSB LSB

GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
Timer 1				Timer 0			

Fig. 5.8.1.

5.9. Give the structure of TCON register.

Ans.

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Fig. 5.9.1.

5.10. What are the applications of the microprocessor ?

Ans.

1. The microprocessor is used in personal computers.
2. It is used in mobile phones and television.
3. It is used in traffic light control.

5.11. Write the applications of microcontroller.

Ans. Application of microcontrollers :

1. It is used in microwave oven. 2. It is used in barcode readers.

5.12. Write about the basic difference between microprocessor and microcontroller. **AKTU 2017-18, Marks 02**

Ans.

S. No.	Microprocessor	Microcontroller
1.	It is just a processor. Memory and I/O component have to be connected externally.	Microcontroller has external processor along with internal memory and I/O components.
2.	Microprocessor have less number of registers, hence more operations are memory based.	They have more number of registers hence the programs are easier to write.

☺☺☺

Scanned by TapScanner