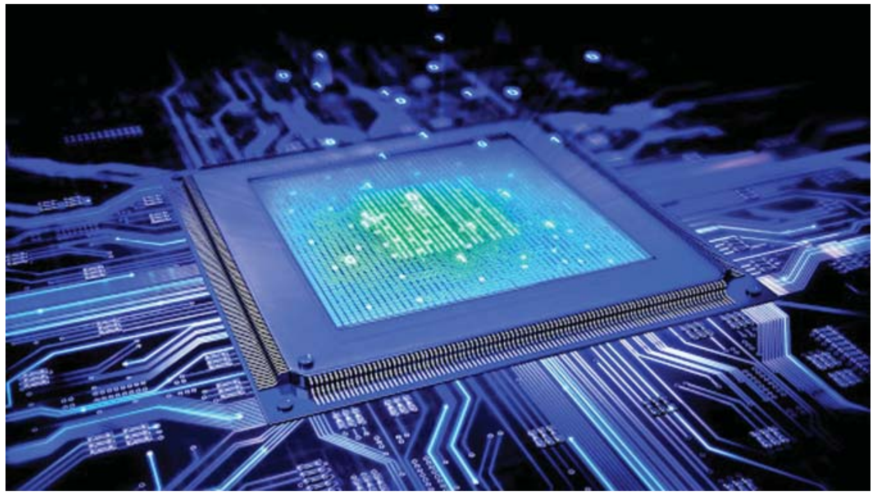
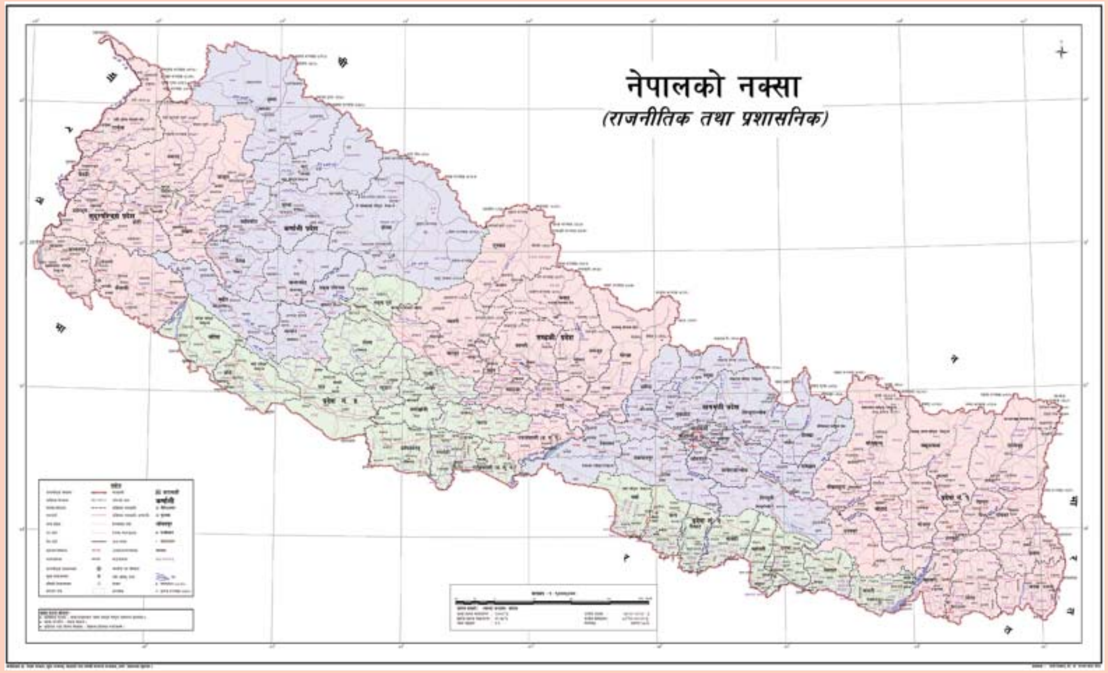
Computer Engineering

10

Microprocessor





Government of Nepal
Ministry of Education, Science and Technology
Curriculum Development Centre
Sanothimi, Bhaktapur

Phone : 5639122/6634373/6635046/6630088
Website : www.moecdc.gov.np

Feedback Copy

Technical and Vocational Stream
Learning Resource Material

Microprocessor

(Grade 10)

Secondary Level
Computer Engineering

Government of Nepal
Ministry of Education, Science and Technology
Curriculum Development Centre
Sanothimi, Bhaktapur

Publisher : Government of Nepal
Ministry of Education, Science and Technology
Curriculum Development Centre
Sanothimi, Bhaktapur

Layout by Khados Sunuwar

Preface

The curriculum and curricular materials have been developed and revised on a regular basis with the aim of making education objective-oriented, practical, relevant and job oriented. It is necessary to instill the feelings of nationalism, national integrity and democratic spirit in students and equip them with morality, discipline and self-reliance, creativity and thoughtfulness. It is essential to develop in them the linguistic and mathematical skills, knowledge of science, information and communication technology, environment, health and population and life skills. it is also necessary to bring in them the feeling of preserving and promoting arts and aesthetics, humanistic norms, values and ideals. It has become the need of the present time to make them aware of respect for ethnicity, gender, disabilities, languages, religions, cultures, regional diversity, human rights and social values so as to make them capable of playing the role of responsible citizens with applied technical and vocational knowledge and skills. This Learning Resource Material for Computer Engineering has been developed in line with the Secondary Level Computer Engineering Curriculum with an aim to facilitate the students in their study and learning on the subject by incorporating the recommendations and feedback obtained from various schools, workshops and seminars, interaction programs attended by teachers, students and parents.

In bringing out the learning resource material in this form, the contribution of the Director General of CDC Dr. Lekhnath Poudel, Pro, Dr. Subarna Shakya, Bibha Sthapit, Kumar Prasun, Yogesh Parajuli, Dr. Romakanta Pandey, Dinesh Khatri, Udayaraj Karki, Rajendra Rokaya is highly acknowledged. The book is written by Sankar Kumar Yadav and the subject matter of the book was edited by Badrinath Timalsina and Khilanath Dhamala. CDC extends sincere thanks to all those who have contributed in developing this book in this form.

This book is a supplimentary learning resource material for students and teachrs. In addition they have to make use of other relevnt materials to ensure all the learning outcomes set in the curriculum. The teachers, students and all other stakeholders are expected to make constructive comments and suggestions to make it a more useful learning resource material.

2076 BS

Ministry of Education, Science and Technology
Curriculum Development Centre

Content

Unit: 1

INTRODUCTION

1.1 Definition of Microprocessor and its applications

Microprocessor is basically a programmable IC. A microprocessor is a computer processor which incorporates the functions of a computer's central processing unit (CPU) on a single integrated circuit (IC) or at most a few integrated circuits. The microprocessor is a multipurpose, clock driven, register based, digital-integrated circuit which accepts binary data as input, processes it according to instructions stored in its memory, and provides results as output. Microprocessors contain both combinational logic and sequential digital logic. Microprocessors operate on numbers and symbols represented in the binary number system.

Applications of Microprocessor

Microprocessor is a mass storage device. They are the advanced form of computers. They are also called as microcomputers. The impact of microprocessor in different fields is significant. The availability of low cost, low power and small weight, computing capability makes it useful in different applications. Now-a-day, a microprocessor based systems are used in instructions, automatic testing product, speed control of motors, traffic light control, light control of furnaces etc. Some of the important areas are mentioned below:

Instrumentation

It is very useful in the field of instrumentation. Frequency counters, function generators, frequency synthesizers, spectrum analyses and many other instruments are available, when microprocessors are used as controller. It is also used in medical instrumentation.

Control

Microprocessor based controllers are available in home appliances, such as microwave oven, washing machine etc., microprocessors are being used in controlling various parameters like speed, pressure, temperature etc. These are used with the help of suitable transduction.

Communication

Microprocessors are being used in a wide range of communication equipment's. In telephone industry, these are used in digital telephone sets. Telephone exchanges and modem etc. The use of microprocessor in television, satellite communication has made teleconferencing possible. Railway reservation and air reservation system also uses this technology. LAN and WAN for communication of vertical information through computer network.

Office Automation and Publication

Microprocessor based microcomputer with software packages has changed the office environment. Microprocessors based systems are being used for word processing, spread sheet operations, storage etc. The microprocessor has revolutionized the publication technology.

Consumer

The use of microprocessor in toys, entertainment equipment and home applications are making them more entertaining and full of features. The use of microprocessors is more widespread and popular. Now the Microprocessors are used in the following areas/equipments;

1. Calculators
2. Accounting system
3. Game machines
4. Complex Industrial Controller
5. Traffic light control
6. Data acquisition systems
7. Military applications.
8. Communications systems
9. Robotics

1.2 Evolution of Microprocessors

The **Intel 4004** is a 4-bit central processing unit (CPU) released by Intel Corporation in 1971. It was the first commercially available microprocessor. It began as the "Busicom Project", a joint development by America's Intel and

Japan's Busicom with initial design concepts by Busicom's Masatoshi Shima and Sharp's Tadashi Sasaki in 1968, before being designed by Intel's Marcian Ted Hoff and Federico Faggin and Busicom'sShima from 1969 to April 1970. it was completed under Faggin's leadership, with Shima's assistance, in January 1971. It was integrated with 2300 transistors with a max CPU clock rate 740KHZ. We can see details of more microprocessors in following table:

Name	Year	Transistors	Data Width	Clock Speed
Intel 4004	1971	2300	4 bits	740 KHZ
Intel 4040	1974	3000	4	500-740 KHZ
Intel 8008	1972	3300	8	0.2-0.8 MHZ
Intel 8080	1974	6000	8	2 MHZ
Intel 8085	1976	6500	8	5 MHZ
Intel 8086	1978	29000	16	5 MHZ
Intel 8088	1979	29000	8	5 MHZ
Intel 80286	1982	134,000	16	6 MHZ
80386	1985	275,000	32	16 MHZ
80486	1989	1,200,000	32	25 MHZ
Pentium	1993	3,100,000	32/64 bits	60 MHZ
Pentium II	1997	7,500,000	64	233 MHZ
Pentium III	1999	9,500,000	64	450 MHZ
Pentium IV	2000	42,000,000	64	1.5 GHZ
Intel Itanium	2001	25,000,000	64	2 GHZ

Table: 1.1

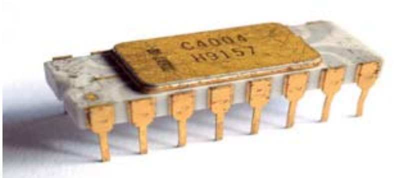


Fig1.1: 16-pin Intel 4004 first microprocessor

Microprocessors were categorized into five generations: first, second, third, fourth, and fifth generations. Their characteristics are described below:

First-generation

The microprocessors that were introduced in 1971 to 1972 were referred to as the first-generation system. First-generation microprocessors processed their instructions serially—they fetched the instruction, decoded it, then executed it. When an instruction was completed, the microprocessor updated the instruction pointer and fetched the next instruction, performing this sequential drill for each instruction in turn.

Second generation

By the late 1970s, enough transistors were available on the IC to usher in the second generation of microprocessor sophistication:16-bit arithmetic and pipelined instruction processing.

Motorola's MC68000 microprocessor, introduced in 1979, is an example. Another example is Intel's 8080. This generation is defined by overlapped fetch, decode, and execute steps (Computer 1996). As the first instruction is processed in the execution unit, the second instruction is decoded and the third instruction is fetched.

The distinction between the first and second-generation devices was primarily the use of newer semiconductor technology to fabricate the chips. This new technology resulted in a five-fold increase in instruction, execution, speed, and higher chip densities.

Third generation

The third generation, introduced in 1978, was represented by Intel's 8086 and the Zilog-Z8000, which were 16-bit processors with minicomputer-like performance. The third generation came about as IC transistor counts approached 250,000. Motorola's MC68020, for example, incorporated an on-chip cache for the first time and the depth of the pipeline increased to five or more stages. This generation of microprocessors was different from the previous ones in that all major workstation manufacturers began developing their own RISC-based microprocessor architectures (Computer, 1996).

Fourth generation

As the workstation companies converted from commercial microprocessors to in-house designs, microprocessors entered their fourth generation with designs surpassing a million transistors. Leading-edge microprocessors such as Intel’s 80960CA and Motorola’s 88100 could issue and retire more than one instruction per clock cycle.

Fifth generation

Microprocessors in their fifth generation, employed decoupled super scalar processing, and their design soon surpassed 10 million transistors. In this generation, PCs are a low-margin, high-volume-business dominated by a single microprocessor.

1.3 Von Neumann Architecture or IAS (Immediate Access Store)

A Von Neumann Architecture machine, designed by physicist and mathematician John von Neumann (1903–1957) is a theoretical design for a stored program computer that serves as the basis for almost all modern computers. A Von Neumann Machine consists of a central processor with an arithmetic/logic unit and a control unit, a memory, mass storage, and input and output.

The von Neumann machine was created by its namesake, John Von Neumann, a physicist and mathematician, in 1945, building on the work of Alan Turing. The design was published in a document called "First Draft of a Report on the EDVAC."

The report described the first stored-program computer. Earlier computers, such as the ENIAC, were hard-wired to do one task. If the computer had to perform a different task, it had to be rewired, which was a tedious process. With a stored-programcomputer,ageneralpurposecomputer could be built to run different programs.

The theoretical design consists of:

- A central processor consisting of a control unit and an arithmetic/logic unit
- A memory unit
- Input and output unit

The Von Neumann Design thus forms the basis of modern computing. A similar model, the Harvard architecture, had dedicated data address and buses for both reading and writing to memory. The von Neumann Architecture won out because it was simpler to implement in real hardware.

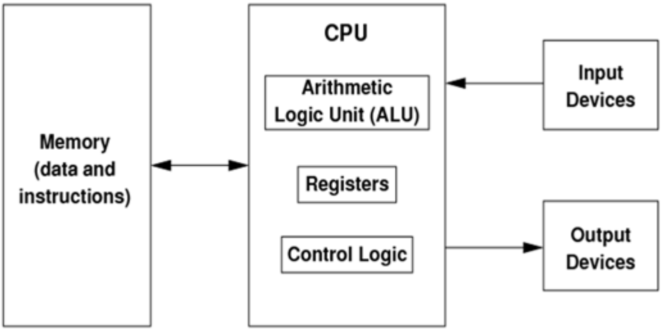


Figure1.2: General Von Neumann Architecture

1.4 Basic Organization of Microprocessors

The basic components of a microcomputer are:

1. CPU
2. Program memory
3. Data memory
4. Output ports
5. Input ports
6. Clock generator

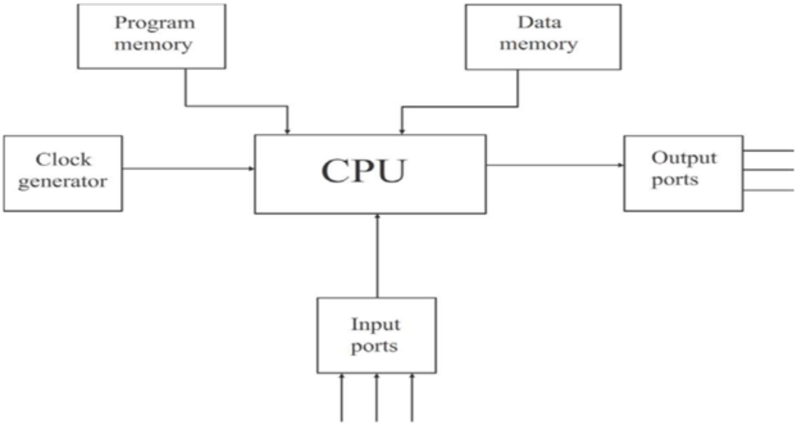


Figure1.3: Components of Microprocessors

Central Processing Unit

The CPU consists of ALU (Arithmetic and Logic Unit), register unit and control unit. The CPU retrieves stored instructions and data wordfrom memory; it also deposits processed data in memory.

a) ALU (Arithmetic and Logic Unit)

This section performs computing functions on data. These functions are arithmetic operations such as additions subtraction and logical operation such as AND, OR etc. Result are stored either in registers or in memory or sent to output devices.

b) Register Unit

It contains various registers. The registers are used primarily to store data temporarily during the execution of a program. Some of the registers are accessible to the uses through instructions.

c) Control Unit

It provides necessary timing & control signals necessary to all the operations in the microcomputer. It controls the flow of databetween the processor and peripherals (input, output & memory). Thecontrol unit gets a clock which determines the speed of the processor.

The CPU has five basic functions

1) It fetches an instructions word stored in memory.
2) It determines what the instruction is telling it to do(decodes theinstruction).
3) It executes the instruction. Executing the instruction mayinclude same of the following major tasks:
    a. Transfer of data from reg. to reg. in the CPU itself.
    b. Transfer of data between a CPU reg. & specified memorylocation.
    c. Performing arithmetic& logical operations on datafrom aspecific memory location or a designated CPU registers.
    d. Directing the CPU to change a sequence of fetchinginstruction, if processing the data created a Specific conditions.
    e. Performing housekeeping function within the CPU itself inorder to establish desired condition at Certain registers.
4) It looks for control signal such as interrupts and providesappropriate responses.
5) It provides states, control, and timing signals that the memoryand input/output section can use.

Program Memory

The basic task of a microcomputer system is to ensure that its CPU executes the desired instruction sequence is the programproperly. The instruction sequence is stored into the program memory on initialization- usually a power up and manual reset the processor starts by executing the instruction in a predetermined location inprogram memory. The first instruction of the program should therefore be in this location in typical μp basic system, the program tobe executed is fixed one which does not change. Therefore μ-program are store on ROM, or PROM, EPROM, EEPROM. In the trainer kit, ROM contains only the monitor program. The user program is not stored in ROM because it needs not to be stored permanently.

Data Memory

A microcomputer manipulates data according to the algorithm given by the instruction in the program in the program memory. These instructions may require intermediate results to be stored, the functional block in μc have same internal register which can also be usedif available for such storage external data memory is needed if the storage requirements are more. Apart from intermediate storage, the data memory may also beused to provide data needed by the program, to store some of the results of the program. Data memory is used for all storage purposesother than storage of program. Therefore, they must have head write capability R/WM or RAM. It stores both the instructions to be executed (i.e. program) and the data involved. It usually contains ROM (Read memory). The ROM can only read and can not be written into and is non-volatile that is, itretains its contents when the power is turned off. A ROM is typically used to store instructions and data that do not change. For example, it stores the monitor program if a microcomputer. One can either read from or write into a R/WM. The RWM is volatile, that is it does not retain its contents when the power is turned off. It is used to store user programmers & data which are temporary and might change during the course of executing a program. During a memory read operation, the content of the addressed location is not destroyed. During a unit operation, the original content of the addressed location is destroyed. Both ROM & R/WM are arranged into words, each of which has a unique address. The address of a word is memory location and it isplaced in parentheses. Therefore, X is an address and (X) is thecontent of that address X. The address decodes taken an address and from the control unit and select the proper memory location and obtaining its content takes a certain amount of time, this time is the access time of the memory. The access time affects the speed of the computer, pins, and the computer must obtain the instruction and data from the memory. Computer memory as usually RAM so that all memory locations have the same access time. The computer must wait shiner of unit's memory, typical memory access time range from several uses. Memory sections often sub-divided into units called pages. The entire memory section may involve millions of cords, when a page contains between 256 & 4k warts. The computer may access a memory location by first decreasing a particular page and the accessing a location on that page. The advantage of paging is that the computer can reach several locations on the same page with justthe address in the page. The process is like describing street address by first specifying aspect and them listing the have numbers. The control section transfers data to or from memory as follows.

1. The control section reads an address to the memory.
2. The control section sends a read and write signal to thememory to indicate, the direction of the transform.
3. The control section waits until transfer has beencompleted. This delay precedes the actual data's transferring the input case andfollows it in the output case.

Input/output Ports

The input & output ports provide the microcomputer the capability to communicate with the outside world. The input ports allow data to pass from the outside world to the microcomputer data which will be used in the data manipulation being done by the microcomputer to send data to output devices. The user can enter instruction (i.e. program) and data inmemory through input devices such as keyboard, or simple switches, CRT, disk devices, tape or card readers. Computers are also used to measure and control physical quantities like temperature, pressure, speed etc. For these purposes, transducers are used to convert physical quantity into proportional electrical signals A/D computersare used to convert electrical signals into digital signals which aresent to the computer. The computer sends the results of the computation to theoutput devices e.g. LED, CRT, D/A converters, printers etc. These I/O devices allow the computer to communicate with the outside world I/O devices are called peripherals.

Clock Generator

Operations inside the µp as well as in other parts of the µc, are usually synchronous by nature. The clock generator generates the appropriate clock periods during which instruction executions arecarried out by the microprocessor. This condition ensures that different path of the systems can proceed in a systematic fashion. Some of the microprocessors have an internal clock generator circuit to generate a clock signal. These microprocessors require an external crystal or RC network to be connected at the appropriate pins for deciding the operating frequency (e.g. 8085). Some microprocessors require an external clock generator (e.g. 8086). These microprocessors also provide an output clock signal which can be used by other devices in the microcomputer system for their can timing and synchronizing.

1.5 Types of Microprocessor

There are basically 5 kinds of microprocessors :

Complex Instruction Set Microprocessors: They are also called as CISM in short and they categorize a microprocessor in which orders can be executed together along with other low-level activities. It mainly performs the task of uploading, downloading and recalling data into and from the memory card. Apart from that it also does complex mathematical calculations within a single command.

Reduced Instruction Set Microprocessor : This processor is also called as RISC in short. These kinds of chips are made according to the function in which the microprocessor can carry out small things within a particular command. In this way it completes more commands at a faster rate.

Superscalar Processors : This is a processor that copies the hardware on the microprocessor for performing numerous tasks at a time. They can be used for arithmetic and as multipliers. They have several operational units and thus carry out more than a one command by constantly transmitting various instructions to the superfluous operational units inside the processor.

The Application Specific Integrated Circuit : This processor is also known as ASIC. They are used for specific purposes that comprises of automotive emissions control or personal digital assistant computer. This kind of processor is made with proper specification but apart from that it can also be made using the off the shelf gears.

Digital Signal Multiprocessors : Also called as DSP's, these are used for encoding and decoding videos or to convert the digital and video to analog and analog to digital. They need a microprocessor that is excellent in mathematical calculations. The chips of this processor are employed in SONAR, RADAR, home theaters audio gears, Mobile phones and TV set top boxes.

Unit: 2

Components of Microprocessor

2.1 I/O (Input/Output)

I/O (input/output) communicates with the outside world. I/O includes two types of devices: input and output; these I/O devices are known as peripherals.The input devices such as keyboard, switches, and an analog to digital (A/D) converter transfer binary information (data and instructions) from the outside world to the microprocessor. The output devices transfer data from the microprocessor to the outside world. They include devices such as light emitting diodes (LEDs), the cathode ray tube (CRT) or video screen, a printer, etc. µCs(PCs) are generally equipped with output devices such as video screen(monitors) and a printer.

2.2 Memory

Memory stores such binary information as instructions and dataand provides that information to the microprocessor whenever necessary. To execute programs, the microprocessor reads instructions and data from the memory and performs the computing operations in its ALU section. Results are either transferred to the output section for display or stored in memory for later use.

Memory can be classified into two groups: Primary (system or main) memory and Secondary (storage) memory.

2.2.1 Primary Memory

This is the memory the microprocessor uses in executing and storing programs. This memory should be able to respond fast enough to keep up with the execution speed of the microprocessor.Therefor it should be random access memory, meaning that the microprocessor should be able to access information from any register with the same speed(independent of its place in the chip). The size of the memory chip is specified in terms of bits.Forexample,1K memory chip means it can store 1024 bits(not bytes).Examples of primary memory are R/WM (RAM) and ROM.

2.2.2 Secondary Memory

The other group is the storage memory, such as magnetic disks and tapes. This memory is used to store programs and results after the completion of program execution. Information stored in these memories is nonvolatile, meaning information remains intact even if the system is turned off. The microprocessor cannot directly execute or process programs stored in these devices; programs need to be copied into the primary memory first. Therefor the size of primary memory, such as 512K or 8M (megabytes), determines how large a program the system can process. The size of the storage memory is unlimited; when one disk or tape is full, the next one can be used.

2.2.3 Cache Memory

Cache memory is a very high-speedsemiconductor memory which can speeds up the CPU. It acts as a buffer between the CPU and the main memory. It is used to hold those parts of data and program which are most frequently used by the CPU. The parts of data and programs are transferred from the disk to cache memory by the operating system, from where the CPU can access them.

2.3 PROCESSOR UNIT

The Microprocessor in many ways similar to the CPU, but includes all the logic circuitry, including the control unit, on one chip. The microprocessor can be divided into three segments for the sake of clarity: arithmetic/logic unit (ALU), register array and control unit.

Arithmetic/Logic Unit: This is the area of the microprocessor where various computing functions are performed on data. The ALU unit performs such arithmetic operations as addition and subtraction, and such logic operations as AND, OR and exclusive OR etc.

Register Unit: This area of microprocessor consists of various registers identified by letters such as B, C, D, E, H, L. These registers are primarily used to store data temporarily during the execution of a program and are accessible to the user through instructions. These registers are also called general purpose registers. So, there are all together six general purpose registers in 8085 Microprocessor.

Control Unit: The control unit provides the necessary timing and control signals to all the operation in the microcomputer. It controls the flow of data between the microprocessor and memory and peripherals.

2.4 Bus Structure of 8085

A *bus* in a microprocessor-based system is defined as a group of separate wires which work together to perform a particular task. A microprocessor-based system, or microcomputer, has three buses which combine to transfer information between the microprocessor and other parts of the system, such as memory or input/output devices. Typical tasks performed by these buses include selecting the source or destination location address for a data transfer, actually moving the data from one part of the system to another, and finally, controlling and synchronizing the electronic devices involved in the data transfer process.

There are three buses in Microprocessor:

1. Address Bus
2. Data Bus
3. Control Bus

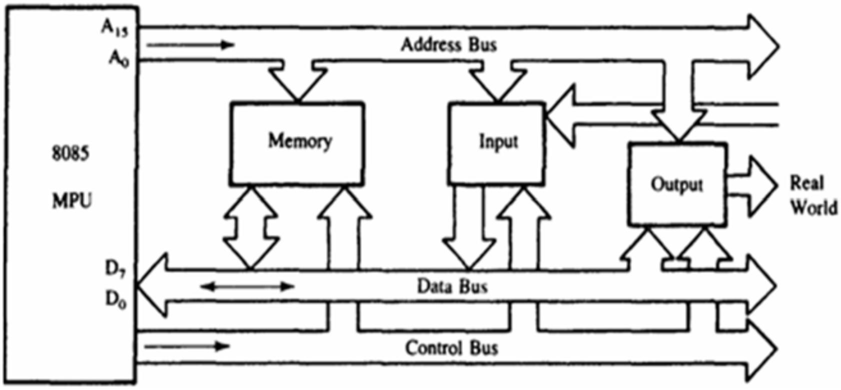


Fig2.1: 8085 Bus Structure

1. **Address Bus:** Generally, microprocessor has 16-bit address bus(A_0-A_{15}). The bus over which the CPU sends out the address of the memory location is known as Address Bus. The address bus carries the address of the memory location to be written or to be read from. The address bus is unidirectional. It means bit flowing occurs only in one direction, only from the microprocessor to the peripheral devices.

We can find how much memory location is using by microprocessor by the formula 2^N where N is the number of bits used for address lines.

Here 2^{16} = 65536 bits or 64KB. So, we can say 8085 can access 64KB of memory locations.

Let us try this question “If a processor has 4GB of memory then how many address lines are required to access this memory?Ans is 32 lines you need to find this by your own.

2. **Data Bus:** 8085 microprocessor has 8-bit data buses. So, it can be used to carry 8 bits of data starting from 00000000(00H) to 11111111(FFH). Here H tells hexadecimal number. The data buses are bidirectional. These lines are used for data flowing in both direction means data can be transferred or received through these lines. The data bus also connects I/O ports and the CPU. The largest number that can be appeared in the bus is 11111111. It has 8 parallel lines of data bus so it can access upto 2^8=256 data bus lines.
3. **Control Bus:**The control bus is used for sending control signals to the memory and I/O devices. The CPU sends control signal on the control bus to enable the outputs of addressed memory devices or I/O port devices.

Some of the control bus signals are as follows:

a) Memory read

b) Memory Write

c) I/O read

d) I/O Write

Unit: 3

Instruction Cycle

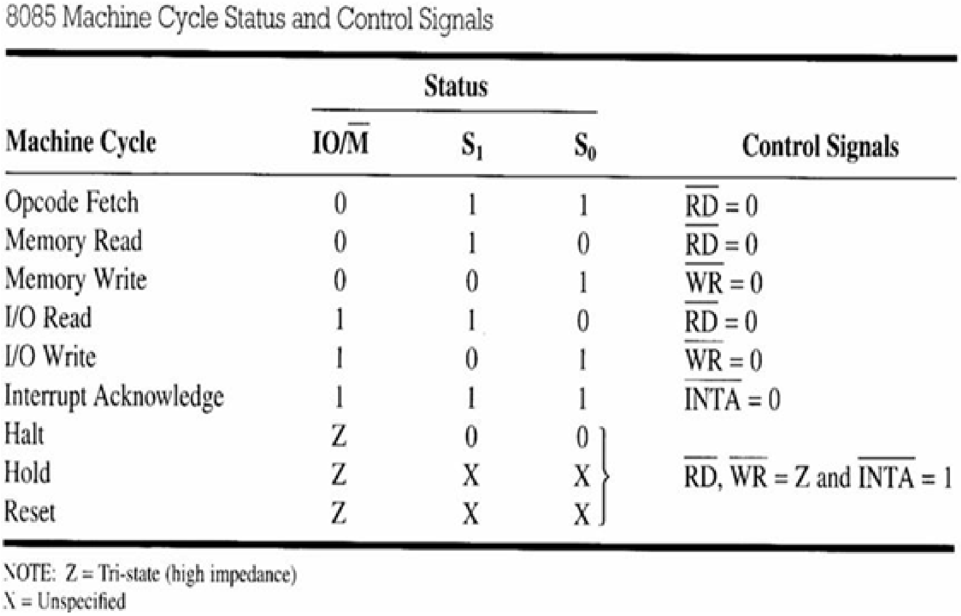
3.1 Instruction cycle, Machine cycle & T-states

Instruction cycle: is defined as the time required completing the execution of an instruction. The 8085 instructions cycle consists of one to six machine cycles or one to six operations.

Machine cycle: is defined as the time required to complete one operation of accessing memory, I/O, or acknowledging the external request. This cycle may consist of 3 to 6 T states.

T states:is defined as one subdivision of the operation performed in one clock period. These subdivisions are internal states synchronized with the system clock, and each T-stateis precisely equal to one clock period. The term T-states and clock period are often used synonymously.

3.2 Machine cycle of 8085 microprocessor



8085 Machine Cycle Status and Control Signals

	Status			
Machine Cycle	IO/$\overline{M}$	S_1	S_0	Control Signals
Opcode Fetch	0	1	1	$\overline{RD}$ = 0
Memory Read	0	1	0	$\overline{RD}$ = 0
Memory Write	0	0	1	$\overline{WR}$ = 0
I/O Read	1	1	0	$\overline{RD}$ = 0
I/O Write	1	0	1	$\overline{WR}$ = 0
Interrupt Acknowledge	1	1	1	$\overline{INTA}$ = 0
Halt	Z	0	0	$\overline{RD}$, $\overline{WR}$ = Z and $\overline{INTA}$ = 1
Hold	Z	X	X	
Reset	Z	X	X	

NOTE: Z = Tri-state (high impedance)
X = Unspecified

Fig3.1 8085 Machine cycle status and control signals

3.2.1 Opcode Fetch Machine Cycle

Each instruction of the processor has one-byte Opcode. The Opcodes are stored in memory. So, the processor executes the Opcode fetch machine cycle to fetch the Opcode from memory. Hence, every instruction starts with Opcode fetch machine cycle. The time taken by the processor to execute the Opcode fetch cycle is 4T.

In this time, the first, 3T-states are used for fetching the Opcode from memory and the remaining T-states are used for internal operations by the processor.

3.2.2 Memory Read Machine Cycle

The memoryread machine cycle is executed by the processor to read a data byte from memory.The processor takes 3Tstates to execute thiscycle.The instruction which have more than one-byte word size will use the machine cycle after the Opcode fetch machine cycle.

3.2.3 Memory Write Machine Cycle

The memory write machine cycle is executed by the processor to write a data byte in a memory location**.** The processor takes, 3T states to execute this machine cycle

3.2.4 I/O Read Machine Cycle

The I/O Read cycle is executed by the processor to read a data byte from I/O port or from the peripheral, which is I/O, mapped in the system.The processor takes 3T states to execute this machine cycle.TheIN instruction uses this machine cycle during the execution**.**

3.2.5 I/O Write Machine Cycle

The I/O write machine cycle is executed by the processor to write a data byte in the I/O port or to a peripheral, which is I/O, mapped in the system.The processor takes 3T states to execute this machine cycle.

3.3 Timing Diagram

It is a graphical representation. It represents the execution time taken by each instruction in a graphical format. The execution time is represented in T-states.

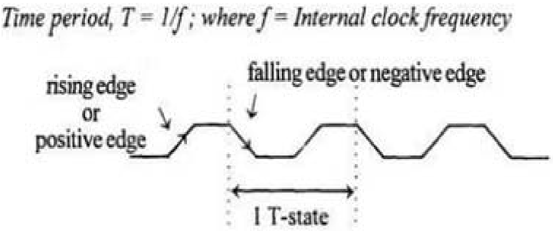
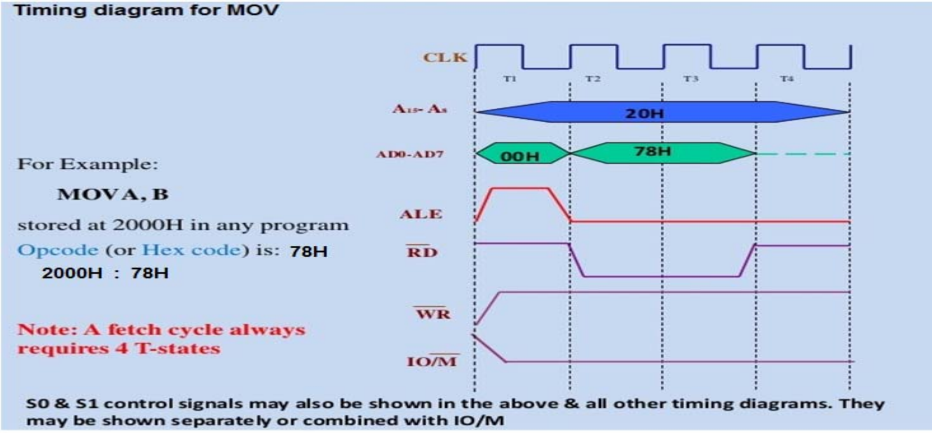

Time period, T = 1/f ; where f = Internal clock frequency
rising edge or positive edge
falling edge or negative edge
1 T-state

Figure 3.2: **Clock Signal**

- Opcode fetch cycle (4T)
- Memory read cycle (3 T)
- Memory write cycle (3 T)
- I/O read cycle (3 T)
- I/O write cycle (3 T)



Timing diagram for MOV

For Example:

MOV A, B

stored at 2000H in any program

Opcode (or Hex code) is: **78H**

2000H : 78H

Note: A fetch cycle always requires 4 T-states

S0 & S1 control signals may also be shown in the above & all other timing diagrams. They may be shown separately or combined with IO/M

Fig3.3 Timing diagram of MOV instruction cycle

The instruction MOV A,B is a 1-byte instruction. Its operation is to copy the content of B in A register. The Microprocessor takes only one machine cycle(OPCODE Fetch) to complete instruction. Hence, Hex code for MOV A,B is passed to the microprocessor.There are 4T-states.

T-1 state

During the T1 state, the contents of the program counter are placed on the 16-bit address bus. The higher order 8 bits are transferred to the address bus(A8-A15) and the lower order 8 bits are transferred to the Multiplexed A/D(AD0-AD7) bus.

ALE(address latch enable)

As soon as ALE goes high, the memory latches the AD0-AD7 bus. At the middle of the T-state the ALE goes low.

T-2 state

During the beginning of the state, the RD' signal goes low to enable memory. During this state,the selected memory location is placed on D0-D7 of the Address/Data multiplexed bus. The Opcode(78H)) is placed in the AD7-AD0 bus.

T3-State

In this state the cycle, the Opcode(78H)) of the A/D bus is transferred to the instruction register in the microprocessor. Now the RD' goes high after this action and thus disables the memory from A/D bus.

T4-State: In this state the Opcode which was fetched from the memory is decoded.

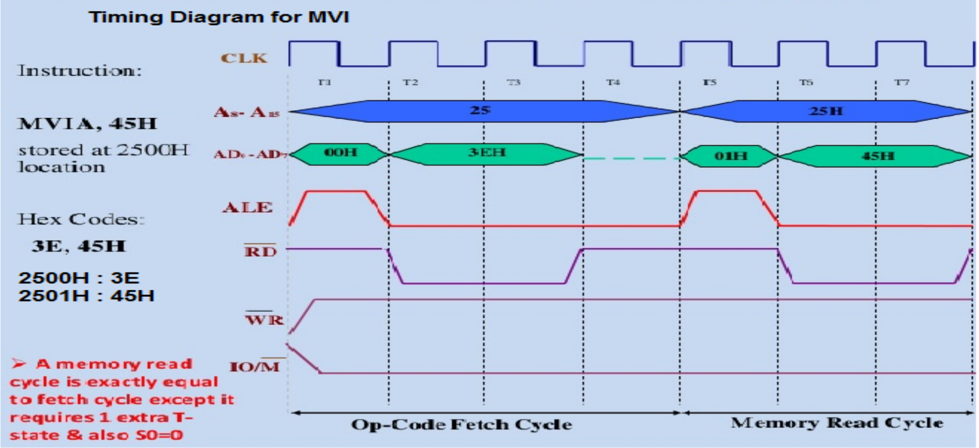


Fig 3.4: Timing diagram of MVI instruction cycle

MVI A, 45H means to immediately load the data 45h to register A of the microprocessor. It is a 2-byte instruction.

The first machine cycle is Opcode fetch and the second is memory read cycle, we have already discussed about Opcode fetch now we will discuss on Memory read cycle.

During the T5, T6 and T7 states the A8-A15 address bus carry the higher bit(25H) of the program counter address(2501H).

During T5 states as the ALE goes high the AD0-AD7 will carry the lower bit(01H) of the program counter address(2501H)

During T6 & T7 states the ALE goes low enabling the data bus and it will read(RD’=0) the data 45H from memory.

Note: please refer to the machine cycle table to know the status of IO/M and S0, S1 signal status for corresponding machine cycle.

Timing Diagram of STA

STA stands for store the content of Accumulator to the given location. Let us take an example of STA 526AH, where the content of Accumulator will store in the location 526AH.

Address	Mnemonics	Opcode
41FF	STA	32H
4200	Lower bit address	6AH
4201	Higher bit address	52H

Table:3.1

It requires 4 machine cycles and 13 T states.

a) Opcode fetch (4T) b) Memory Read (3T)c.) Memory Read (3T) d.) Memory Write(3T)

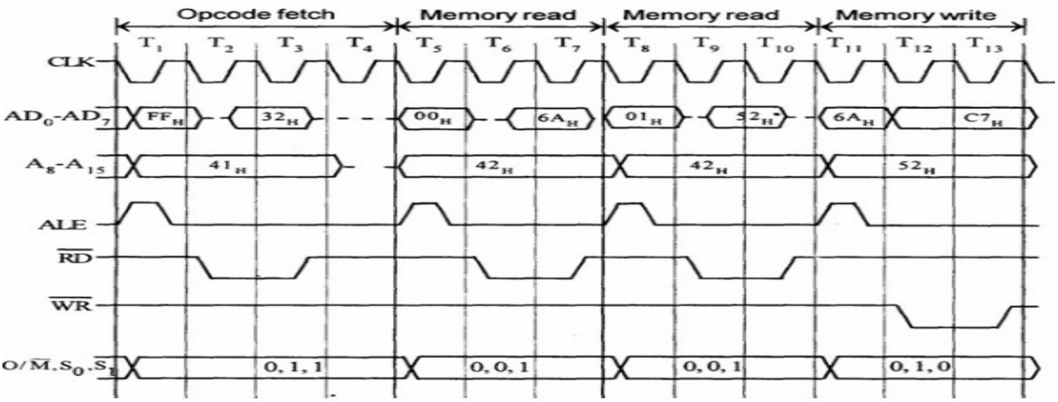


Fig3.5:Timing diagram of STA instructions

Timing Diagram of IN

IN port address refers to input to Accumulator from I/O port. It requires 3 M/C and 10 T states. Let us take example of IN C0H. The following operation are carried out.

- Fetching the Opcode DBH from the memory 4125H.
- Read the port address C0H from 4126H.
- Read the content of port C0H and send it to the accumulator.
- Let the content of port is 5EH.

Table : 3.2

Address	Mnemonics	Op code
4125H	IN	DBH
4126H		C0H

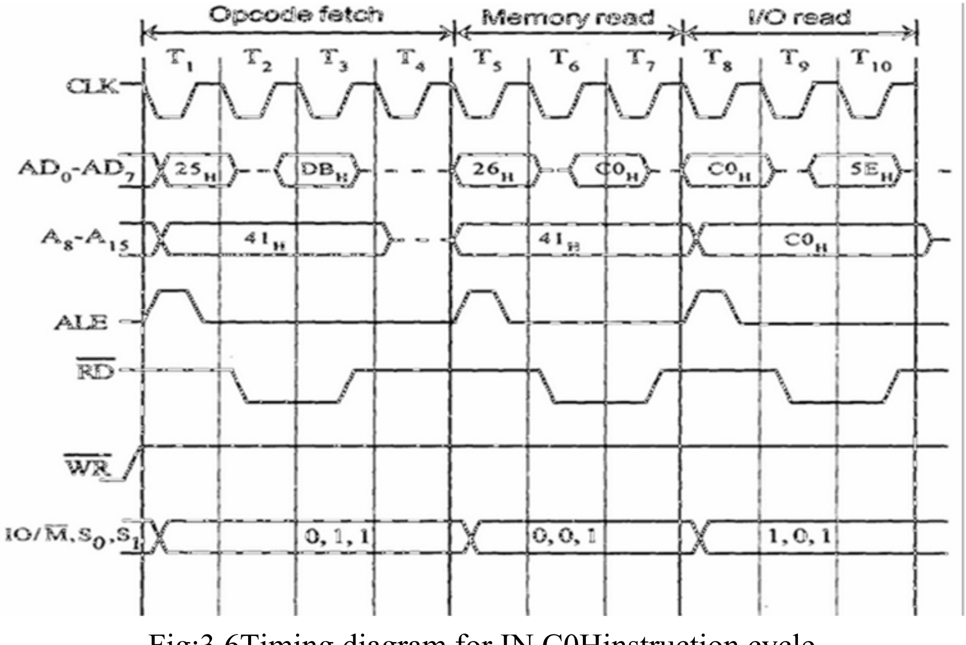


Fig:3.6Timing diagram for IN C0Hinstruction cycle

Timing diagram for OUT

OUT port address it stands for output from accumulator to I/O port. It also requires 3 machines cycles and 10 T states. The machine cycles are: a.) Opcode fetch (4T)b.) Memory Read (3T)c.) I/O write (3T)

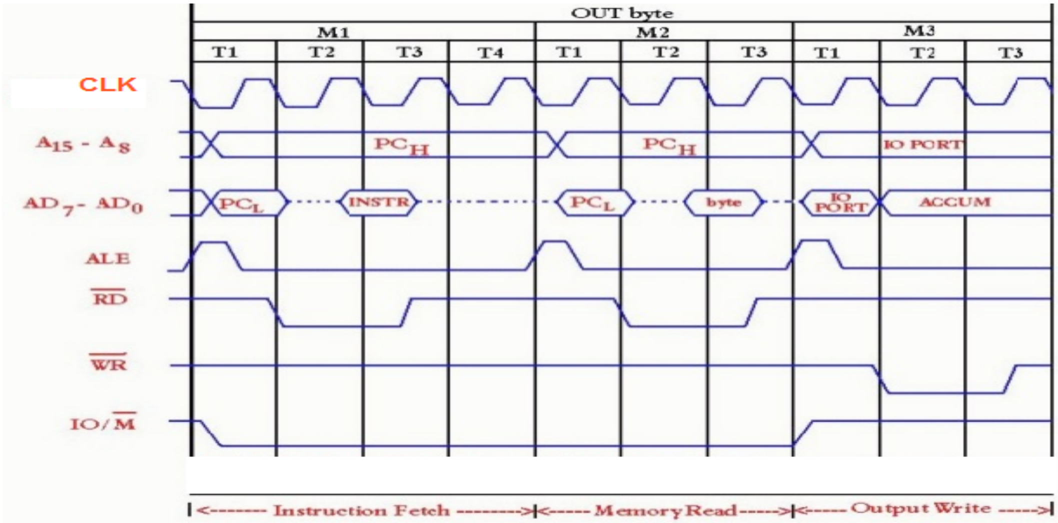


Fig 3.7 Timing diagram of OUT

Numerical

Example : Calculate the total execution to execute the instruction MVI B, 05H. Let the clock frequency of the microprocessor be 3.125 MHZ.

Solution

Mnemonic	Instruction byte	Machine Cycle	T states
MVI B	Opcode	Opcode Fetch	4T
05H	Immediate Data	Read Immediate Data	3T

Here total T states =4T+3T=7T

Clock frequency of microprocessor (f) = 3.125 MHZ

Time(T) for one clock = $\frac{1}{3.125\ MHZ}$ = 0.32 µs

So, total execution time for instruction=7T

= 7*0.32 µs

= 2.24 µs

Therefor the microprocessor can execute MVI 05H in 2.24 µs.

Try It: (a) Calculate the execution time of (a) STA 2050H (b) LDA 4001H (c) MOV A,B

b) Draw timing diagram for LDA 2050H instruction 2050H hold data 5FH.

Unit: 4

Intel 8085 Microprocessor

4.1 Fnctional Block Diagram

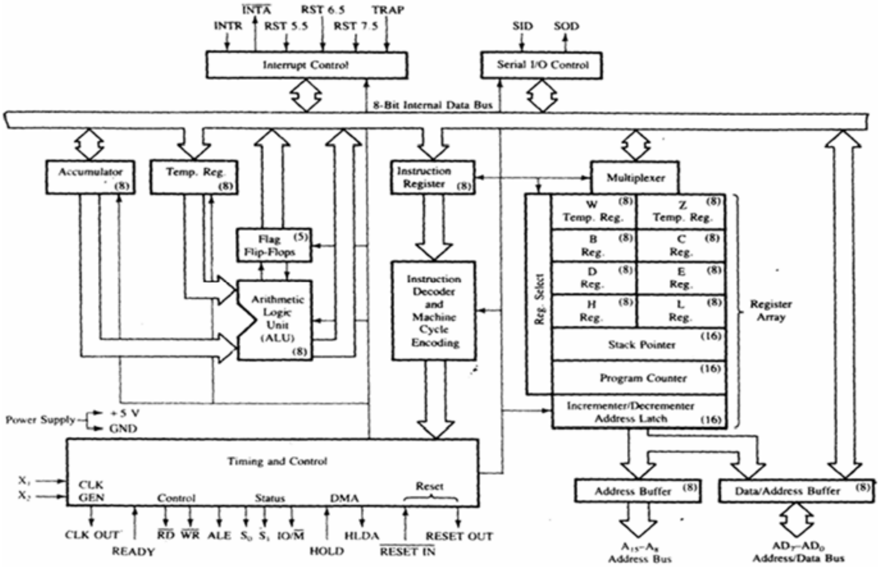


Fig: 4.1 Functional block diagram 8085

4.2 Pin Configuration Of 8085

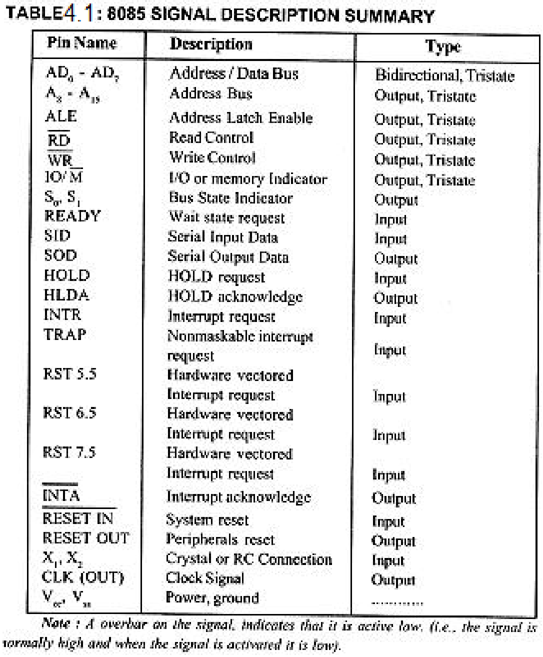
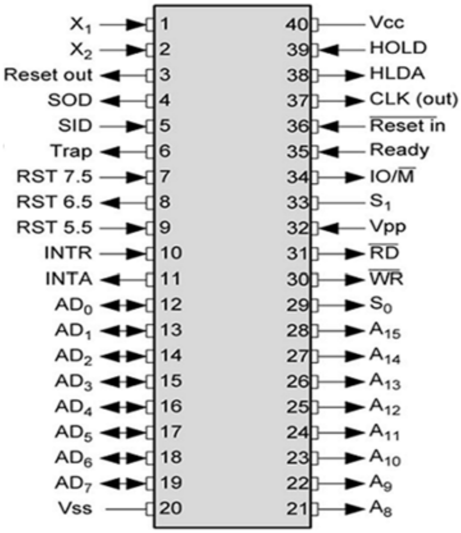


TABLE4.1: 8085 SIGNAL DESCRIPTION SUMMARY

Pin Name	Description	Type
AD_0 - AD_7	Address / Data Bus	Bidirectional, Tristate
A_8 - A_{15}	Address Bus	Output, Tristate
ALE	Address Latch Enable	Output, Tristate
$\overline{RD}$	Read Control	Output, Tristate
$\overline{WR}$_	Write Control	Output, Tristate
IO/$\overline{M}$	I/O or memory Indicator	Output, Tristate
S_0, S_1	Bus State Indicator	Output
READY	Wait state request	Input
SID	Serial Input Data	Input
SOD	Serial Output Data	Output
HOLD	HOLD request	Input
HLDA	HOLD acknowledge	Output
INTR	Interrupt request	Input
TRAP	Nonmaskable interrupt request	Input
RST 5.5	Hardware vectored Interrupt request	Input
RST 6.5	Hardware vectored Interrupt request	Input
RST 7.5	Hardware vectored Interrupt request	Input
$\overline{INTA}$	Interrupt acknowledge	Output
RESET IN	System reset	Input
RESET OUT	Peripherals reset	Output
X_1, X_2	Crystal or RC Connection	Input
CLK (OUT)	Clock Signal	Output
V_{cc}, V_{ss}	Power, ground	....

Note : A overbar on the signal, indicates that it is active low. (i.e., the signal is normally high and when the signal is activated it is low).

Fig 4.2: pin configuration 8085

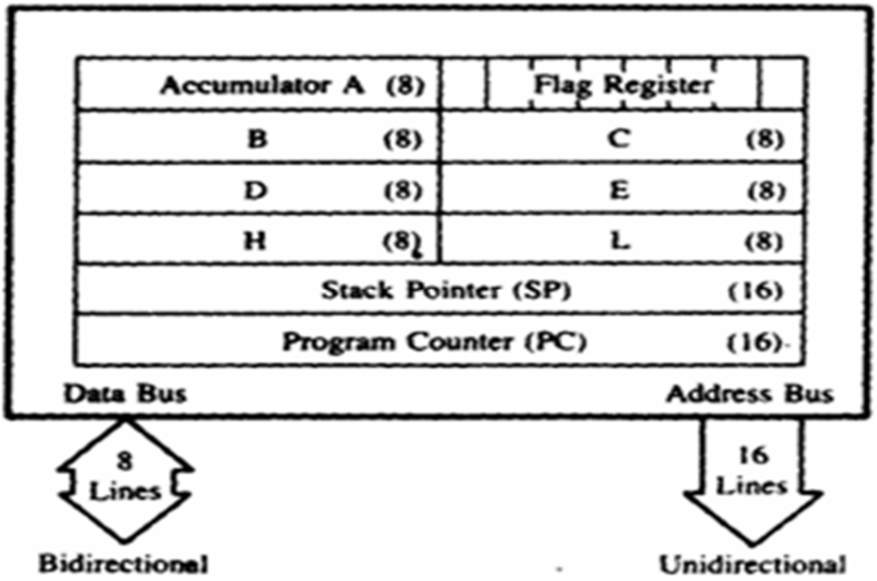


4.3 Description of each blocks: Registers, Flag, Data and Address Bus, Timing and Control Unit, Interrupts

Accumulator: It is an 8-bit register which is used to perform arithmetical and logical operation. It stores the output of any operation. Italso works as registers for I/O accesses.

Temporary Register:-It is an 8-bit register which is used to hold the data on which the accumulator is computing operation. It is also called as operand register because it provides operands to ALU

Registers: These are general purposes registers. Microprocessor consists 6 general purpose registers of 8-bit each named as B, C, D, E, H and L. Generally, theses registers are not used for storing the data permanently. It carries the 8-bits data. These are used only during the execution of the instructions. These registers can also be used to carry the 16 bits data by making the pair of 2 registers. The valid register pairs available are BC, DE and HL. We cannot use other pairs except BC,DEand HL. These registers are programmed by the user.



Accumulator A (8)	Flag Register
B (8)	C (8)
D (8)	E (8)
H (8)	L (8)
Stack Pointer (SP) (16)	
Program Counter (PC) (16)	

Fig 4.3: Array of Registers

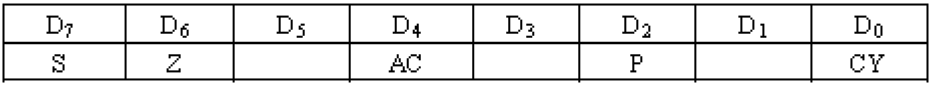
ALU: Algorithm Logical Units performs the arithmetic operations and logical operation.

Flag Registers: It consists of 5 flip flop which changes its status according to the result stored in an accumulator. It is also known as status registers. It is connected to the ALU.

There are five flip-flops in the flag register are as follows:

1. Sign(S)
2. Zero(z)
3. Auxiliary carry(AC)
4. Parity(P)
5. Carry(C)

The bit position of the flip flop in flag register is:



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

All of the threeflip flop set and reset according to the stored result in the accumulator.

1. Sign(S)- If D7 of the result is 1 then sign flag is set otherwise reset. As we know that a number on the D7 always decides the sign of the number. if D7 is 1: the number is negative. if D7 is 0: the number is positive.

2. **Zeros(Z)-**If the result stored in an accumulator is zero then this flipflop is set otherwise it is reset.
3. **Auxiliary carry(AC)-**If any carry goes from D3 to D4 in the output then it is set otherwise it is reset.
4. **Parity(P)**-If the no of 1's is even in the output stored in the accumulator then it is set otherwise it is reset for the odd.
5. **Carry(C)-**If the result stored in an accumulator generates a carry in its final output then it is set otherwise it is reset.

Instruction registers(IR): It is a 8-bit register. When an instruction is fetched from memory then it is stored in this register.

Instruction Decoder: Instruction decoder identifies the instructions. It takes the information from instruction register and decodes the instruction to be performed.

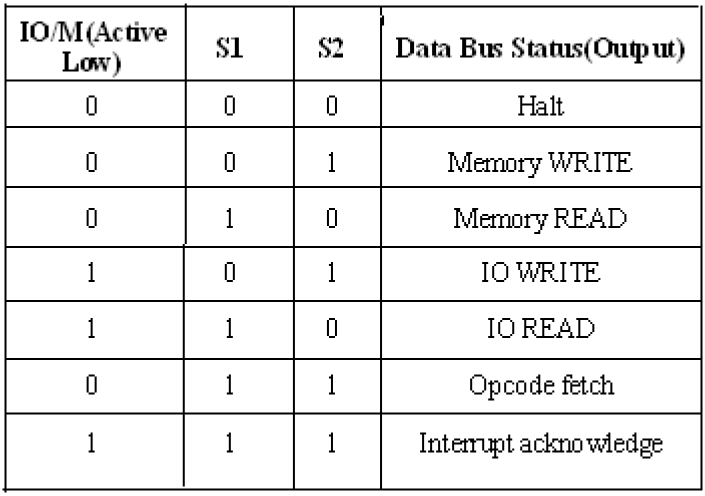
Program Counter: It is a 16-bit register used as memory pointer. It stores the memory address of the next instruction to be executed. So, we can say that this register is used to sequencing the program. Generally the memory have 16-bit addresses so that it has 16 bit memory. The program counter is set to 0000H.

Stack Pointer: It is also a 16-bit register used as memory pointer. It points to the memory location called stack. Generally, stack is a reserved portion of memory where information can be stores or taken back together.

Timing and Control Unit:- It provides timing and control signal to the microprocessor to perform the various operation. It has three control signals. It controls all external and internal circuits. It operates with reference to clock signal. It synchronizes all the data transfers. There are three control signals:

1. **ALE**-Arithmetic Latch Enable, itprovides control signal to synchronize the components of microprocessor.
2. **RD**-This is active low used for reading operation.
3. **WR**-This is active low used for writing operation.

There are three status signals used in microprocessor **S0, S1** and **IO/M**. It changes its status according the provided input to these pins.



IO/M(Active Low)	S1	S2	Data Bus Status(Output)
0	0	0	Halt
0	0	1	Memory WRITE
0	1	0	Memory READ
1	0	1	IO WRITE
1	1	0	IO READ
0	1	1	Opcode fetch
1	1	1	Interrupt acknowledge

Fig 4.4: Timing and control signals

Serial Input Output Control- There are two pins in this unit. This unit is used for serial data communication.

Interrupt Unit

An interrupt is a signal to the processor emitted by hardware or software indicating an event that needs immediate attention. An interrupt alerts the processor to a high-priority condition requiring the interruption of the current code the processor is executing. The processor responds by suspending its current activities, saving its state, and executing a function called an interrupt handler (or an interrupt service routine, ISR) to deal with the event. This interruption is temporary, and, after the interrupt handler finishes, the processor resumes normal activities. There are two types of interrupts: hardware interrupts and software interrupts. We will discuss in unit7 about the interrupts and their priority.

There are 6 interrupt pins in this unit. Generally, an external hardware is connected to these pins. These pins provide interrupt signal sent by external hardware to microprocessor and microprocessor sends acknowledgement for receiving the interrupt signal. Generally, INTAis used for acknowledgement.

Register Section: Many registers have been used in microprocessor.

4.4. Instructions- Opcode and Operands

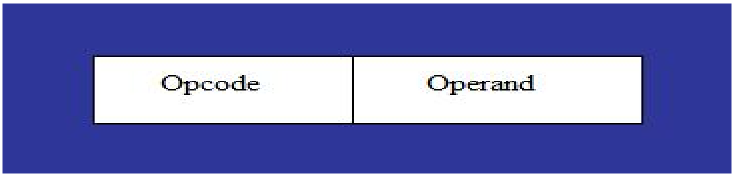
*What is Instruction?

An instruction is a binary pattern designed inside the microprocessor to perform a specific function. In other words, it is actually a command to the microprocessor to perform a given task on specified data.

Instruction Set

The entire group of these instructions are called instruction set. The instruction setdetermines what functions the microprocessor can perform.

Instruction Format



Each instruction has two parts: one is the task to be performed called the operation code (opcode) and the other is the data to be operated on called the operand (data).

Instruction and Word Size

1. One word or one-byte Instruction

   It includes the opcode and operand in the same byte.

Example: `ADD B`

2. Two word or two-byte Instruction

   First byte specifies the opcode and second byte specifies the operand.

Example: `MVI A, 05`

3. Three word or three-byte Instruction

   The first byte specifies the opcode and the following two bytes specify the 16-bit address. The second byte is lower order and the other is higher order.

Example: `JMP 2085H`

4.5 Addressing Modes in 8085

Addressing modes Instructions are command to perform a certain task in microprocessor. The instruction consists of op-code and data called operand. The operand may be the source only, destination only or both of them. In these instructions, the source can be a register, a memory or an input port. Similarly, destination can be a register, a memory location, or an output port. The various format (way) of specifying the operands are called addressing mode. So, addressing mode specifies where the operands are located rather than their nature. The 8085 has 5 addressing modes:

1) Direct addressing mode

The instruction using this mode specifies the effective address as part of instruction. The instruction size either 2-bytes or 3-bytes with first byte op-code followed by 1 or 2 bytes of address of data.

e.g.

LDA 9500H; A ← [9500H]

IN 40H; A ← [40H]

This type of addressing is called absolute addressing.

2) Register direct addressing mode

This mode specifies the register or register pair that contains the data.

e.g.

MOV A, B

ADD H

XOR C etc.

3) Register indirect addressing mode

In this mode the address part of the instruction specifies the memory whose contents are the address of the operand. So, in this type of addressing mode, it is the address of the address rather than address itself.

e.g. MOV A, M, STAX B etc.

4) Immediate addressing mode

In this mode, the operand position is the immediate data. For 8-bit data, instruction size is 2-bytes and for 16-bit data, instruction size is 3-bytes.

E.g. MVI A, 32H, LXI B, 4567H, SBI 76H etc.

5) Implied or Inherent addressing mode:

The instructions of this mode do not have operands. E.g. NOP: No operation HLT: Halt EI: Enable interrupt DI: Disable interrupt.

Unit: 5

Programming with Intel 8085 Microprocessor

5.1 Instruction format and data format

An 8085program instructions format may be one, two or three length.

1. 1-byte instruction (8 bit)

1-byte instructionincludesoperand and opcode in the same byte. It occupies 1-byte space in memory.

E.g.

Opcode operand

MOV	A, B
ADD	B
CMA	

2. Byte instruction (16 bit)

In a 2-byte instruction, the first byte specifies the operation and second byte specifies opcode. It occupies 2-byte space in memory

Opcode

DATA OR ADDRESS

E.g.

Opcode	Operand
MVI	A, 45H
MVI	B, F2H

3. 3-byte instruction

In 3-byte instruction the first byte specifies the opcode and following two byte specifies the 16-bit address. It occupies 3-byte space in memory.

E.g.

Opcode	Operand	Hex code
LDA	2050H	3A
		50
		20
JMP	2085H	C3
		85
		20

Instruction with a Memory Address

Operation: go to address 2085.

Instruction: JMP 2085

Opcode: JMP

Operand: 2085

Binary code

1100 0011	C3	1st byte
1000 0101	85	2nd byte
0010 0000		203rd byte

Data Format

In 8085, data stores in the form of 8-bit binary integer. In the Intel 8085, bit 0 is referred as the least signification bit (LSB) and bit 7 is referred as most signification bit (MBS).

Data can be represented in four formats

1. ASCII (American Standard Code for Information Interchange) e.g. - A-Z, a-z, 0-9 etc.

2. BCD (Binary Code Decimal) e.g. - 0-F
3. Signed Integer e.g. +2, +3,
4. Unsigned Integer e.g. -2, -3,

Assume the accumulator contains the following value: 0100 0001

There are four ways of reading this value:

It is an unsigned integer expressed in binary; the equivalent decimal number would be 65.

It is a number expressed in BCD (Binary Coded Decimal) format. That would make it, 41.

It is an ASCII representation of a letter. That would make it the letter A.

It is a string of 0's and 1's where the 0th and the 6th bits are set to 1 while all other bits are set to 0.

5.2 Intel 8085 instruction Set& Instructions types

An Instruction is a command given to the computer to perform a specified operation on given data. The instruction set of a microprocessor is the collection of the instructions that the microprocessor is designed to execute. The instructions described here are of Intel 8085.The *instruction set* of the 8085 microprocessor consists of 74 instructions with 246 different bit patterns.These instructions are of Intel Corporation. They cannot be used byother microprocessor manufactures. The programmer can write a program in assembly language using these instructions. These instructions have been classified as below:

1. Data transfer instruction

This instruction copies data from one location to another location. There are various types of data transfers.

Between register

E.g. MOV A, B; MOV C, B etc.

Specify the data byte to register or memory location.

E.g. MVI B, 23H; LDA 2500H etc.

Between memory location and register

E.g. LXI 2050H

Between input and output

E.g. IN 02H; OUT PORT1 etc.

2. Arithmetic instruction

This instruction performs arithmetic operation such as addition, subtraction, increment, and decrement. E.g. ADD B, ADI 25H, SUB C, SUI 23H, SUB M, INR B, DCR A etc.

Addition (ADD, ADI)

– Any 8-bit number.

– The contents of a register.

– The contents of a memory location.

- Can be added to the contents of the accumulator and the result is stored in the accumulator.

Subtraction (SUB, SUI)

– Any 8-bit number

– The contents of a register

– The contents of a memory location

- Can be subtracted from the contents of the accumulator. The result is stored in the accumulator.

Increment (INR) and Decrement (DCR)

- The 8-bit contents of any memory location or any register can be directly incremented or decremented by 1.
- No need to disturb the contents ofthe accumulator

3. Logical instruction

This instruction performs logical operation like AND, OR, COMPARE, ROTATED.

CMP- compare

RAL- rotation left

RAR- rotation right

ANA- AND

ORA- OR

These instructions perform logic operations on the contents of the accumulator – ANA, ANI, ORA, ORI, XRA and XRI

- **Source: Accumulator and**

  – An 8-bit number

  – The contents of a register

  – The contents of a memory location

- **Destination: Accumulator**

ANA	R/M	AND Accumulator withReg/Mem
ANI	#	AND Accumulator With an 8-bit number
ORA	R/M	OR Accumulator withReg/Mem
ORI	#	OR Accumulator With an 8-bit number
XRA	R/M	XOR Accumulator withReg/Mem
XRI	#	XOR Accumulator With an 8-bit number

- **Complement**
- 1’s complement of the contents of the accumulator.

  CMA No operand
- Rotate – Rotate the contents of the accumulator one position to the left or right.

– RLC Rotate the accumulator left. Bit 7 goes to bit 0AND the Carry flag.

– RAL Rotate the accumulator left through the carry. Bit 7 goes to the carry and carry goes to bit 0.

– RRC Rotate the accumulator right. Bit 0 goes to bit 7 AND the Carry flag.

– RAR Rotate the accumulator right through the carry. Bit 0 goes to the carry and carry goes to bit 7.

- **Compare**
- Compare the contents of a register or memory location with the contents of the accumulator.

– CMP R/M compares the contents of the register or memory location to the contents of the accumulator.

– CPI # Compare the 8-bit number to the contents of the accumulator.

- The compare instruction sets the flags (Z, Cy, and S).
- The compare is done using an internal subtraction that does not change the contents of the accumulator.

  A –(R / M / #)

4. Branch instruction

Two types

Unconditional branch

Go to a new location no matter what.

Conditional branch

Go to a new location if the condition is true.

Unconditional Branch

– JMP Address Jump to the address specified (Go to).

– CALL Address Jump to the address specified but treat it as a subroutine.

– RET Return from a subroutine.

– The addresses supplied to all branch operations must be 16-bits.

Conditional Branch

– Go to new location if a specified condition is met.

JZ Address (Jump on Zero) – Go to address specified if the Zero flag is set.

JNZ Address (Jump on NOT Zero) – Go to address specified if the Zero flag is not set.

JC Address (Jump on Carry) – Go to the address specified if the Carry flag is set.

JNC Address (Jump on No Carry) – Go to the address specified if the Carry flag is not set.

JP Address (Jump on Plus) – Go to the address specified if the Sign flag is not set

JM Address (Jump on Minus) – Go to the address specified if the Sign flag is set.

5. Machine Control

– HLT

Stop executing the program. – NOP

No operation

Exactly as it says, do nothing.

Usually used for delay or to replace instructions during debugging.

These instruction control machines function such as HLT, NOP, and EI etc.

5.3 Assembly language program and flowchart

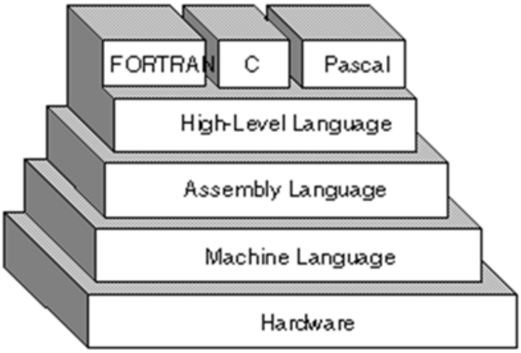


fig:5.1: programming languages

A vocabulary and set of grammatical rules for instructing a computer to perform specific tasks. The term programming language usually refers to high-level languages, such as BASIC, EDA, FORTAN C, C++ etc. Each language has a unique set of keywords (words that it understands) and a special syntax for organizing programming instructions.

High-level programming languages, while simple compared to human languages, are more complex than the languages the computer actually understands, called machine languages. Each different type of CPUhas its own unique machine language.

Lying between machine languages and high-level languages are languages called assembly languages. Assembly languages are similar to machine languages, but they are much easier to program in because they allow a programmer to substitute names for numbers. Machine languages consist of numbers only.

Lying above high-level languages are languages called fourth-generation languages (usually abbreviated 4GL). 4GLs are far removed from machine languages and represent the class of computer languages closest to human languages.

Regardless of what language you use, you eventually need to convert your program into machine language so that the computer can understand it. There are two ways to do this:

1) Compile the program

2) interpret the program

The question of which language is best is one that consumes a lot of time and energy among computer professionals. Every language has its strengths and weaknesses. For example, FORTRAN is a particularly good language for processing numerical data, but it does not lend itself very well to organizing large programs. Pascal is very good for writing well-structured and readable programs, but it is not as flexible as the C programming language. C++ embodies powerful object-oriented features, but it is complex and difficult to learn.

The choice of which language to use depends on the type of computer the program is to run on, what sort of program it is, and the expertise of the programmer.

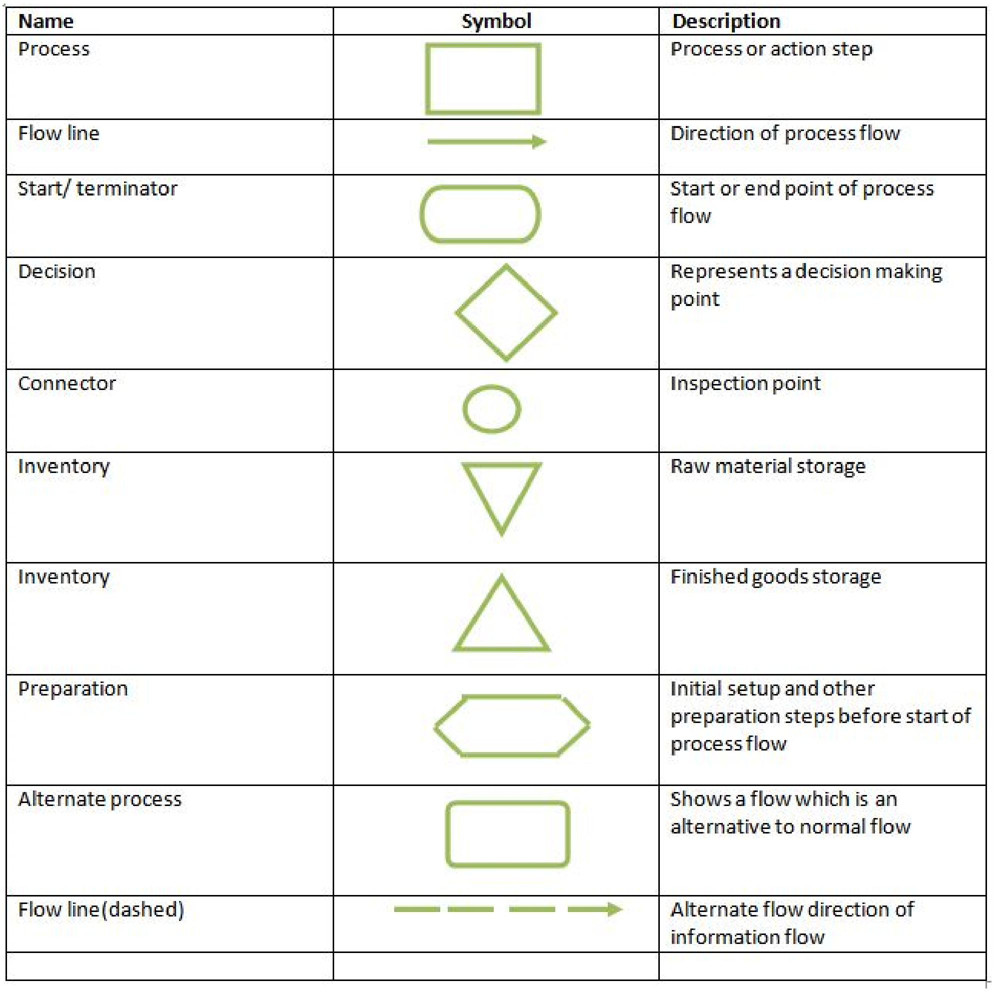
Flowchart:1. it is a diagram of the sequence of operations in a computer program.

2. This means that you should create your flow diagram such that anyone could implement the program

in any programming language. Thus, a flow diagram or chart is independent of the programming language.

Flow Chart

Complex Program → Break it down in modules you can easily test.



Name	Symbol	Description
Process	▭	Process or action step
Flow line	→	Direction of process flow
Start/ terminator	⬭	Start or end point of process flow
Decision	◇	Represents a decision making point
Connector	○	Inspection point
Inventory	▽	Raw material storage
Inventory	△	Finished goods storage
Preparation	⬡	Initial setup and other preparation steps before start of process flow
Alternate process	▢	Shows a flow which is an alternative to normal flow
Flow line(dashed)	- - - →	Alternate flow direction of information flow

Fig5.2: Flowchart.

Microprocessor program

#) Program-01

Write a program to store data type 32H into memory location.

```
MVI A, 32H     : store 32H in the accumulator.
STA 2000H      : copy accumulator content to memory location.
HLT            : stop
```

#) Program-02

Write a program to store data of register B into memory location 4000H.

```
    MOV A, B
    STA 4000H
    HLT
```

#) Program-03

Write a program to exchange the content of memory location 2000H and 4000H.

```
LDA 2000H     : get the content of memory location 2000H into accumulator
MOV B, A      : save the content into B.
LDA 4000H     : get the content of memory location 4000H into accumulator
STA 2000H     : store the content of accumulator at 2000H.
MOV A, B      : save the content back to accumulator.
STA 4000H     : store the content of accumulator into memory location 4000H.
HLT           : stop
```

#) Program-04

Write a program to add two numbers.

2000H	14H
2001H	89H
2002H	14+89=9D (Result)

```
    LXI H, 2000H :     HL point 2000H.
    MOV A, M :         get first operand.
    INX H :            HL points 2001H.
    ADD M :            Add
    INX H :            HL points 2002H.
    MOV M, A :         store the result 2003H.
    HLT :              stop.
```

#) Program-05

Write a program to subtract two numbers.

```
4000H            51H
4001H            19H
4002H            51-19=38H (Result)
LXI H, 4000H     : HL point 4000H.
MOV A, M         : get first operand.
INX H            : HL points 4001H.
SUB M            : Add
INX H            : HL points 4002H.
MOV M, A         : store the result 4003H.
HLT              : stop.
```

#) Program-06

Write a program to find the 1's complements of the number stored at memory location 4400H and stored the complements number at memory location 4300H.

```
4400H            55H
43OOH            AAH (Result)
LDA 4400H        : get the number
CMA              : complement
STA 4300H        : stored the result.
HLT              : stop
```

#) Program-07

Write a program to find the 1's complements of the number stored at memory location 4400H and stored the complements number at memory location 4300H.

```
4400H            55H
43OOH            AAH+1= ABH (Result)
LDA 4400H        : get the number
CMA              : complement
ADI, 01H         : add one in the number.
```

```
STA 4300H          : stored the result.
HLT                : stop
```

#) Program-08

Write a program to shift an 8-bit data four bits right. Assume that the data is in the register C.

```
MOV A, C
RAR
RAR
RAR
RAR
MOVC, A
HLT
```

#) Program -09

Write a program to find the larger of two 8-bit numbers.

```
LXI H, 3000H       : Load the H-L pair with address 3000H.
MOV A, M           : copy the content of memory to acc.
INX H              : increment H-L pair.
MOV B, M           : copy the content of memory to register B.
CMP B              : compare B with A.
JNC 200BH          : jump to the address 200B if there is no carry.
MOV A, B           : move registers B to A.
INX H              : increment H-L pair.
MOV M, A           : move acc. to memory.
HLT                : stop
```

#) Program- 10

Multiply two 8-bits number.

```
LXI H, 3000H
MOV B, M
```

```
INX H
MOV C, M
MVI A, OOH
ADD B
DCR C
JNZ 200BH
INX H
MOV M, A
HLT
```

#) Program -11

Write a program to add two 16 bits number.

```
2000H           15H
2001H           10H
2002H           25H
2003H           20H
Results         1015+2025= 303A
2004H           3A
2005H           30
LHLD 2000H      : get first 16-bit number in HL
XCHG            : save first 16-bits in DE
LHLD 2002H      : get second 16-bits number.
MOVA, E         : get lower byte of first number.
ADD L           : add lower byte of second number
MOV L, A        : stored the result in L register.
MOV A, D        : get higher byte of first number.
ADD H           : add higher byte of second number with no carry.
MOV H, A        : stored result in H register.
SHLD 2004H      : stored result in memory.
HLT             : stop
```

#) program-12

Write a program to find the greatest number.

Address	Mnemonics	Operand	Opcode	Remarks
2000	LXI	H, 3000H	21	Load H-L pair with address 3000H.
2001	00			Lower-order of 3000H.
2002	30	Higher-order of 3000H.		
2003	MOV	A, M	7E	Move the 1st operand from memory to reg. A.
2004	INX	H	23	Increment H-L pair.
2005	MOV	B, M	46	Move the 2nd operand from memory to reg. B.
2006	CMP	B	B8	Compare B with A.
2007	JC	200BH	DA	Jump to address 200BH if there is .no carry
2008	0B			Lower-order of 200BH.
2009	20			Higher-order of 200BH.
200A	MOV	A, B	78	Move smallest from reg. B to reg. A
200B	INX	H	23	Increment H-L pair.
200C	MOV	M, A	77	Move the result from reg. A to M
200D	HLT	76	Halt.	

Unit: 6

Basic I/O, Memory R/W and Interrupt Operations

6.1 Memory Read Operation:

The MPU send a pulse called memory read as the control signal. The pulse activated the memory chip and the content of memory location is place on the data bus and brought inside the microprocessor.

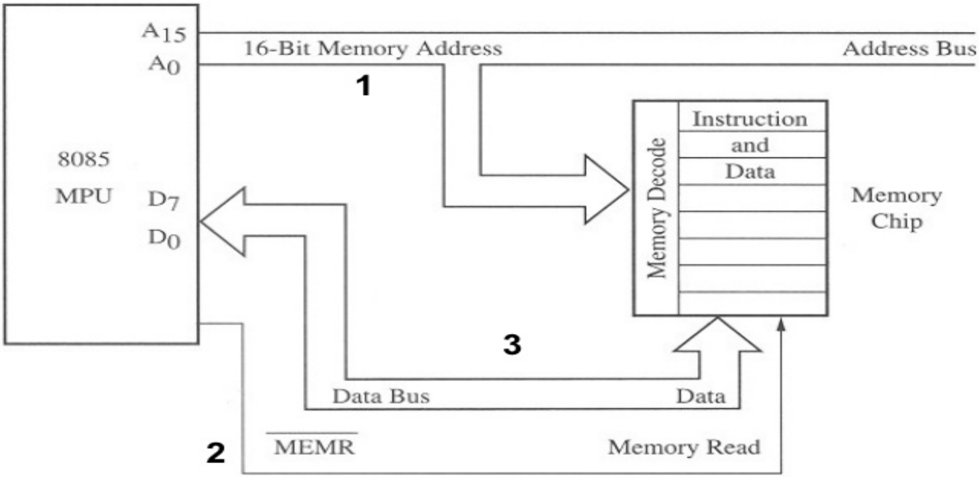


Fig 6.1 Mermory

Memory Write Operation:

The MPU send a pulse called memory write as the control signal. The pulse activated the memory chip and the content of memory location is place on the data bus and brought inside the microprocessor.

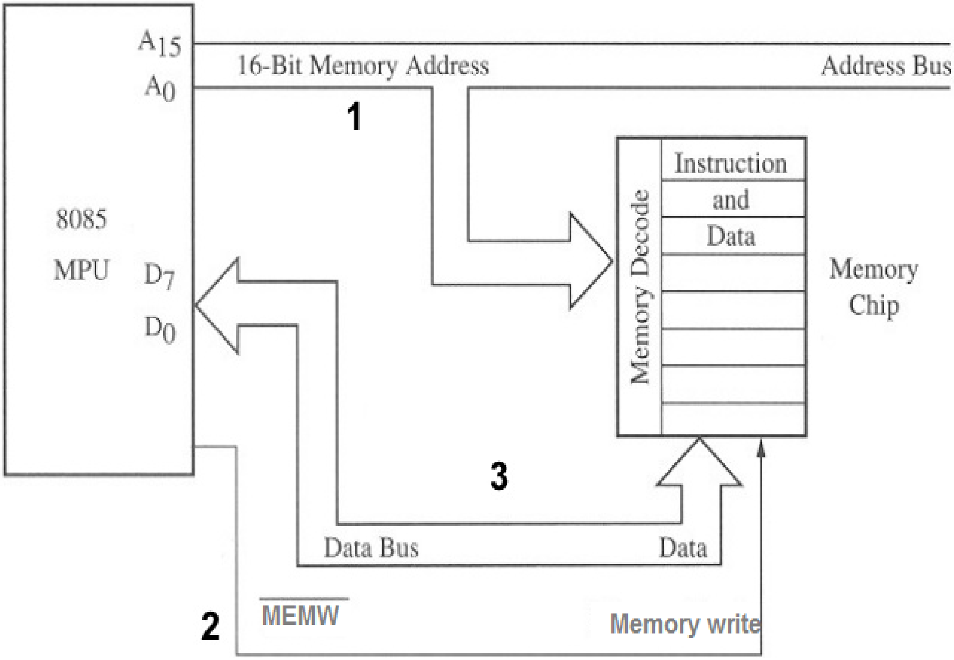


Fig 6.2: Memory write operation

6.2 Direct Memory Access (DMA)

What is DMA ?

A technique for transferring data from memory to a device without passing it through the CPU.

Application of DMA

1. For making quick backup.
2. For real time applications.
3. In expansion board such as CD-ROM card are capable to accessing to computer's DMA channel.

Advantage of DMA

1. High data transmission.
2. Low power requirement.
3. Decrease the CPU load.

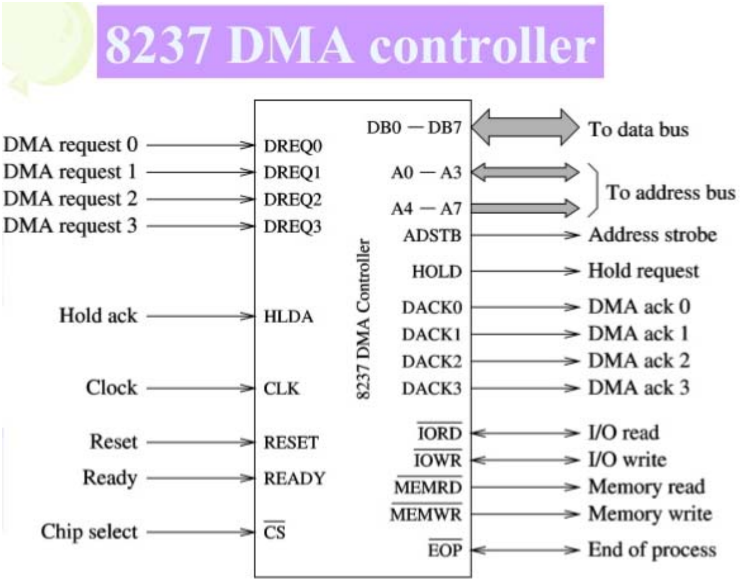


Fig 6.3: 8237 DMA controller

How DMA techniques work?

Basic DMA operation

- The direct memory access (DMA) technique provides direct access to the memory while the microprocessor is temporarily disabled.
- A DMA controller temporarily borrows the address bus, data bus, and control bus from the microprocessor and transfers the data bytes directly between an I/O port and a series of memory locations.
- The DMA transfer is also used to do high-speed memory-to memory transfers.
- Two control signals are used to request and acknowledge a DMA transfer in the microprocessor-based system.
- The HOLD signal is a bus request signal which asks the microprocessor to release control of the buses after the current bus cycle.
- The HLDA signal is a bus grant signal which indicates that the microprocessor has indeed released control of its buses by placing the buses at their high-impedance states.
- The HOLD input has a higher priority than the INTR or NMI interrupt inputs.

DMA Data Transfer scheme

- Data transfer from I/O device to memory or vice-versa is controlled by a DMA controller.
- This scheme is employed when large amount of data is to be transferred.
- The DMA requests the control of buses through the HOLD signal and the MPU acknowledges the request through HLDA signal and releases the control of buses to DMA.
- It's a faster scheme and hence used for high speed printers.

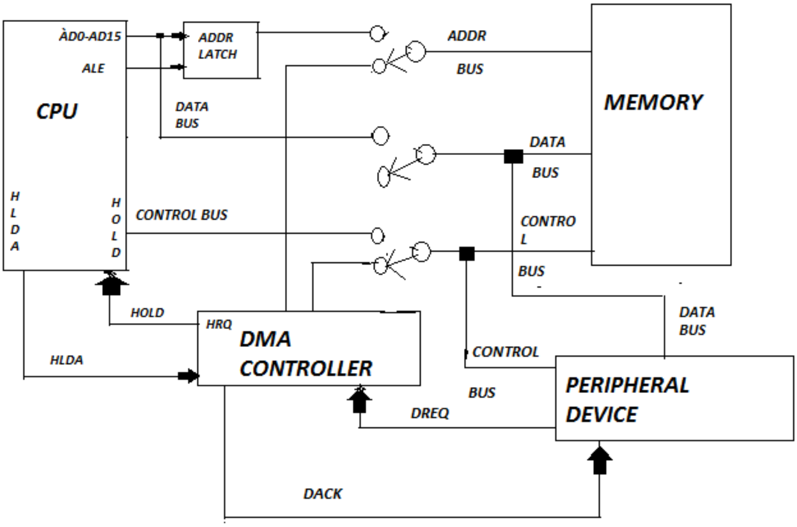


Fig 6.4: DMA Data transfer scheme

6.3 Interrrupts

What is interrupt?

Interrupt is a signal that a peripheral board sends to central processor in order to request attention.

OR,

Interrupt is a process where an external device can get the attention of the microprocessor.

The process starts from the I/O device

The process is asynchronous

Type of interrupt:

1. External interrupt
2. Internal interrupt
3. Software interrupt

1. External interrupt

They are initiated via the microprocessor's interrupt pin by external device. They are two types:

a. Maskable interrupt (can be delayed)

It is a signal which can be enabled and disabled by external instruction, Such as EI, DI.

b. Non maskable interrupt (cannot be delayed)

It is a signal which cannot be enabled and disabled by external instruction. It has higher priority. E.g. TRAP.

2. Internal interrupt

They are initiated via the microprocessor's interrupt pin by internal device.

There are two types of services:

1. Polled interrupt

Several external devices (device1, device2, device N) are connected to a single interrupt line (INT) of the processor. When one and more devices are activated then INT line high. It is very simple for large number of device.

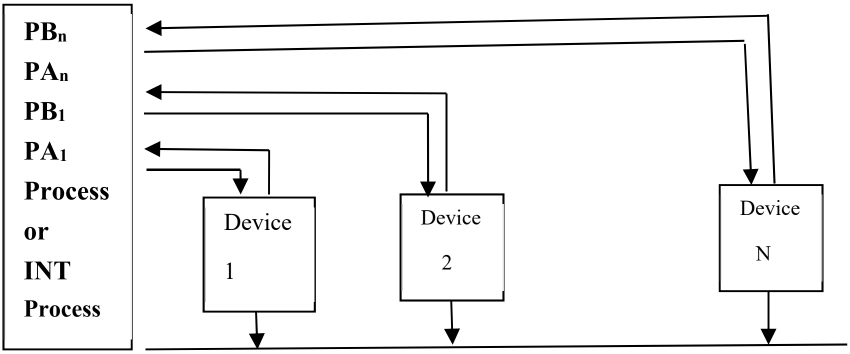


Fig6.5: Polled interrupt

2. Chained interrupt (vector)

This is a hardware concept of handling the multiple interrupt. In this technique the device are connected in a chain fashion as shown in figure. These devices with a higher priority place in the first position followed by the lower priority.

Address bus

Data Bus

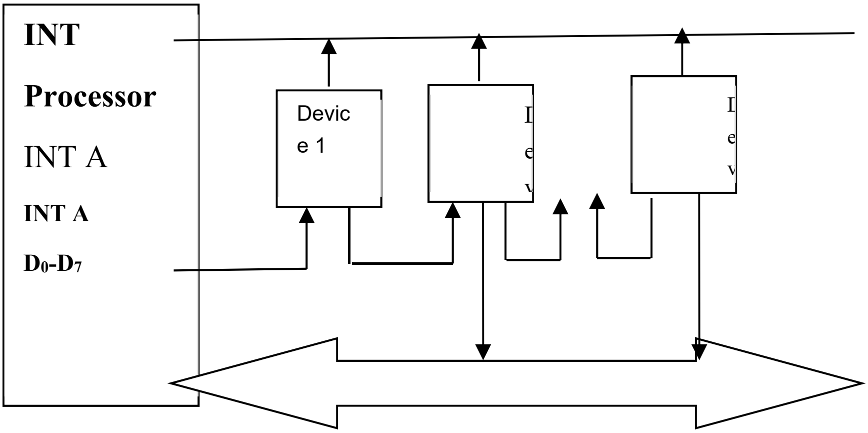


Fig 6.6: Chained interrupt

Interrupt pins and their priority

The processor has 5 interrupt pins in the order of ascending priority TRAP has highest priority and INTR has low priority.

- INTR (maskable interrupt)
- RST 5.5 (maskable interrupt)
- RST 6.5 (maskable interrupt)
- RST 7.5 (maskable interrupt)
- TRAP (non-maskable interrupt)

The 8085 has 5 interrupt inputs.

- The INTR input.
- The INTR input is the only non-vectored interrupt.
- INTR is maskable using the EI/DI instruction pair.
- RST 5.5, RST 6.5, RST 7.5 are all automatically vectored.

- RST 5.5, RST 6.5, and RST 7.5 are all maskable.
- TRAP is the only non-maskableinterrupt in the 8085
- TRAP is also automatically vectored

Interrupt name	Maskable	Vectored
INTR	Yes	No
RST 5.5	Yes	Yes
RST 6.5	Yes	Yes
RST 7.5	Yes	Yes
TRAP	No	Yes

Short note :

Write about TRAP

- TRAP is the only non-maskable interrupt.
- It does not need to be enabled because it cannot be disabled.
- It has the highest priority amongst interrupts.
- It is edge and level sensitive. – It needs to be high and stay high to be recognized.
- Once it is recognized, it won't be recognized again until it goes low, then high again.
- TRAP is usually used for power failure and emergency shutoff.

6.4 8259 Operations

The 8259A is a programmable interrupt controller designed to work with Intel Processors 8085, 8086 and 8088. The 8259A is upward-compatible with its predecessor 8259. The main difference between the two is that 8259A can be used with the Intel's 8086/8088 16-bit microprocessors. It also includes additional features such as the level triggered mode, buffered mode, and automatic-end of interrupt mode.

The 8259A interrupt controller can:

- Manage eight interrupts according to the instructions written into its control registers. This is equivalent to providing eight interrupts pin on the processor in place of one INTR (8085) pin.

- Vector an Interrupt request anywhere in the memory map. However, all eight interrupts are spaced at the interval of either four or eight locations. This eliminates the major drawback of the 8085 interrupts in which all interrupts are vectored to memory locations on page 00H.
- Resolve eight levels of interrupts priorities in variety of modes, such as fully nested mode, automatic rotation mode, and specific rotation mode (to be discussed later).
- Mask each interrupt request individually.
- Read the status of pending interrupts, in-service interrupts, and masked interrupts.
- Be expanded to 64 priority levels by cascading additional 8259As.
- To be setup to work with either the 8085 microprocessor mode or the 8086/8088 processor mode.
- **Block Diagram and Explanation**

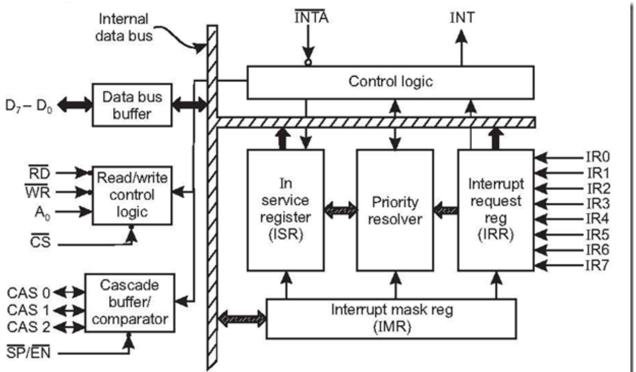


Fig 6.4(a): Block diagram of 8259A

The function of some of the blocks needs explanation, which is given below:

a. Read/Write Logic

This is a typical Read/Write control logic. When the address line A_0 is at logic 0, the controller is selected to write a command or read a status. The chip select logic and A_0 determines the port address of the controller.

b. Control Logic

This block has two pins: INT (Interrupt) as and output, and $\overline{\text{INTA}}$ (Interrupt

Acknowledge) as an input.

The INT is connected to the interrupt pin of MPU. Whenever a valid interrupt is asserted, the signal goes high. The $\overline{\text{INTA}}$ is the Interrupt Acknowledge signal from the MPU.

c. Interrupt Registers and Priority Resolver

The Interrupt Request Register (IRR) has eight input lines (IR_0-IR_7) for interrupts. When these lines go high, the requests are stored in the register. The In-service Register (ISR) stores all the levels that are currently being serviced, and the Interrupt Mask Register (IMR) stores the masking, bits of the interrupts lines to be masked. The Priority Resolver(PR) examines these three registers and determines whether INT should be sent to the MPU.

d. Cascade Buffer/Comparator

This block is used to expand the number of interrupt levels by cascading two or more 8259As.

- **Priority Modes and other features**

Many types of priority modes are available under software control in the 8259A, and they can be changed dynamically during the program by writing appropriate command words. Commonly used priority modes are discussed below.

1. Fully nested modes

This is a general purpose mode in which all IR_S are arranged from highest to lowest with IR_0 as the highest and IR_7 as the lowest.

In addition, any IR can be assigned the highest priority in this mode; the priority sequence will than begin at that IR. 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

2. Automatic rotation mode

In this mode, a device alters being service receives the lowest priority. Assuming that IR_2 just being serviced will receive the 7th priority.

IR0	IR1	IR2	IR3	IR4	IR5	IR6	IR7
5	6	7	0	1	2	3	4

3. 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 the other priority.

Unit: 7

Input/Output Interfaces

7.1 Parallel Communication: Introduction & Applications

Communication between two machines requires a transfer of signal from an output to an input. There needs to be a sender and receiver of the signal for the complete communication process to take place. Serial and parallel communications are both ways of transferring data over networks. Both systems have a unique way of operating, with differences between the two ranging from the mode of relaying signals to the level of connectivity they require.

1
0
1
1
1
0
1
0

Fig7.1(a): Data transmission Block format

Parallel communication is the method of transferring blocks, e.g. BYTEs, of data at the same time in same direction.

As you can appreciate parallel communication is faster than serial. For this reason, the internal connections in a computer, i.e. the busses, are linked together to allow parallel communication. However, the use of parallel communication for longer distance data communication is unfeasible for economic and practical reasons, e.g. amount of extra cable required and synchronization difficulties. Therefore, all long distance data communications takes place over serial connections.

7.2 Serial Communication

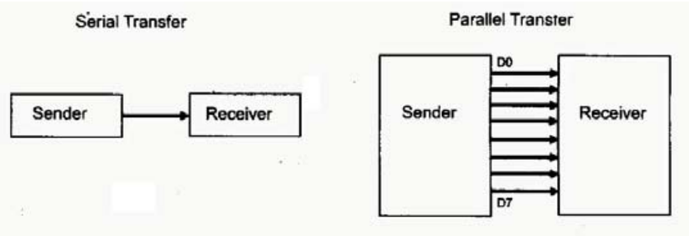


Fig 7.1(b): Serial communication

Introduction& applications

Serial communication is the method of transferring one bit at a time through a medium. A serial communication device sends data in bits, and at the end the bits harmonize to form a byte of data. Serial communication uses fewer connections and cables than parallel communication does. The use of fewer wires in serial communication makes its signals clearer, thus making it suitable for long distance communication.

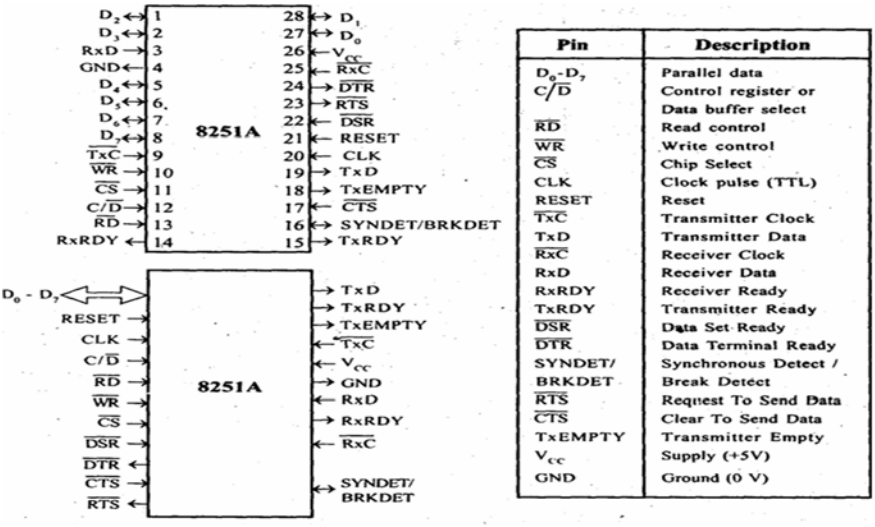
1					

Fig 7.1(c): Data sending in sequence serial communication

Serial communication has a single port with a connector while a parallel port is usually connected to eight wires. A serial port therefore, requires less investment for purchasing wires compared to parallel communication. In terms of space, parallel communication requires more space to accommodate its wires while serial communication utilizes minimal space for its single connector.

Introduction to programmable communication interface 8251A

The 8251A is the industry standard Universal Synchronous/Asynchronous Receiver/Transmitter (USART) designed for data communications with microprocessor families such as MCS-48, 80, 85, and iAPX 86, 88. The 8251A is used as a peripheral device and is programmed by the CPU to operate using virtually any serial data transmission technique presently in use (including IBM “bi-sync”). The USART accepts data characters from the CPU in parallel format and then converts them into a continuous serial data stream for transmission. Simultaneously, it can receive serial data streams and convert them into parallel data characters for the CPU. The USART will signal the CPU whenever it can accept a new character for transmission or whenever it has received a character for the CPU. The CPU can read the complete status of the USART at any time. These include data transmission errors and control signals such as SYNDET, TxEMPTY. The chip is fabricated using Intel's high performance HMOS technology.



Pin	Description
D_0-D_7	Parallel data
$C/\overline{D}$	Control register or Data buffer select
$\overline{RD}$	Read control
$\overline{WR}$	Write control
$\overline{CS}$	Chip Select
CLK	Clock pulse (TTL)
RESET	Reset
$\overline{TxC}$	Transmitter Clock
TxD	Transmitter Data
$\overline{RxC}$	Receiver Clock
RxD	Receiver Data
RxRDY	Receiver Ready
TxRDY	Transmitter Ready
$\overline{DSR}$	Data Set Ready
$\overline{DTR}$	Data Terminal Ready
SYNDET/ BRKDET	Synchronous Detect / Break Detect
$\overline{RTS}$	Request To Send Data
$\overline{CTS}$	Clear To Send Data
TxEMPTY	Transmitter Empty
V_{CC}	Supply (+5V)
GND	Ground (0 V)

Fig 7.2: Pin diagram of 8251A

Functional Block Diagram of 8251A

The functional block diagram of 8251A consists of five sections. They are:

- Read/Write control logic
- Transmitter
- Receiver
- Data Bus Buffer
- Modem Control

Read/Write Control Logic

The control logic interfaces the chip with the processor, determines the functions of the chip according to the control word in its register, and monitors the data flow

Transmitter

The transmitter section converts a parallel word received from the processor into serial bits and transmits them over the TxD line to a peripheral

Receiver

The receiver section receives serial bits from a peripheral, converts them into a parallel word, and transfers the word to the µP

Data Bus Buffer

This bidirectional register can be addressed as an input and an output port when the pin is low

Control/Data pin (C/$\overline{\boldsymbol{D}}$)

When this signal is high, the control register or the status register is addressed; when it is low, the data buffer is addressed. The control register and the status register are differentiated by $\overline{RD}$and $\overline{WR}$ signals respectively.

Modem Control

The modem control is used to establish data communication through modems over telephone lines

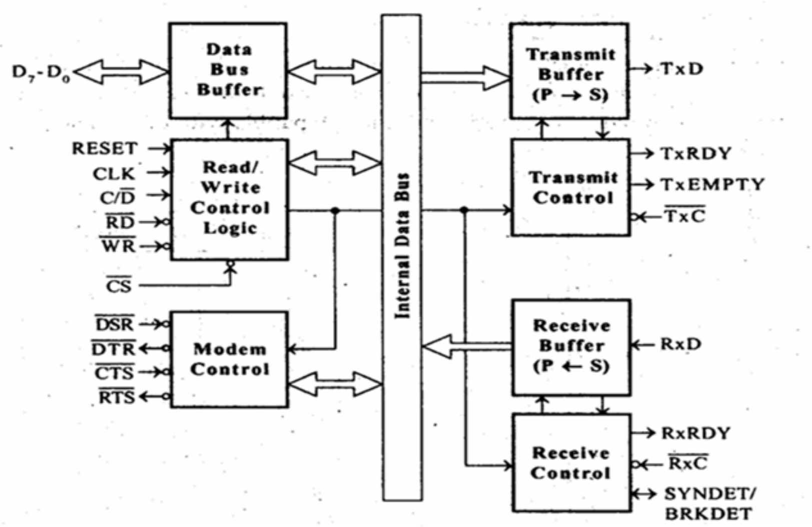


Figure7.3: Functional block diagram of 8251A

- **Basic concept of Synchronous and Asynchronous modes**

There are two operating modes of USART(8251A)

1. Synchronous Mode
2. Asynchronous Mode

Synchronous mode

In program-to-program communication, the synchronous mode requires that each end of an exchange respond in turn without initiating a new communication.

In this mode the Receiver (RX) clock and Transmitter (TX) clocks are synchronized with respect to each other. The data are transmitted at a fixed rate and in the form of blocks or frames.

Asynchronous mode

Operation means that a process operates independently of other processes. The Receiver (Rx) and Transmitter (Tx) clocks may not be synchronized with respect to each other. The Data doesn't have to be transmitted at fixed rate.

Apart from synchronous and Asynchronous mode of data transfer there are other modes also,

In terms of data in a line

a) Synchronous mode

b) Asynchronous mode

In terms of number of senders

a) Simplex

b) Half Duplex

c) Full Duplex

We will not be discussing about the Simplex, Half duplex and full Duplex in this class.

7.3 Simple I/O, Strobe I/O, Single handshake I/O, Double handshake I/O

Simple I/O

For simple I/O, the buffer switch and latch switches i.e. LED are always connected to the input and output ports. The devices are always ready to send or receive data.

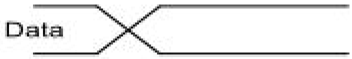


Fig: Simple I /O

Wait Interface (Simple strobe I/O)

In this technique, MP need to wait until the device is ready for the operation.

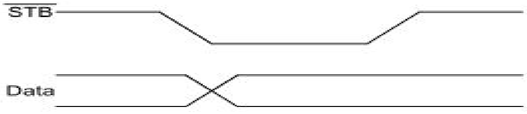


Fig: Simple Strobe I/O

Consider a simple keyboard consisting of 8 switches connected to a MP through a parallel interface CKT (Tri-state buffer). The switch is of dip switches. In order to use this keyboard as an input device the MP should be able to detect that a key has been activated. This can be done by observing that all the bits are in required order. The processor should repeatedly read the state of input port until it finds the right order of bits i.e. at least 1 bit of 8 bits should be 0. Consider the tri-state A/D converter.

- Used to convert analog to digital data which can be read by I/O unit of MP.
- When SOC appears 1, I/O unit should ready for reading binary data/digital data.
- When EOC's status is 1, then I/O unit should stop to read data.
- Strobe signal indicates the time at which data is being activated to transmit.

Single Handshaking

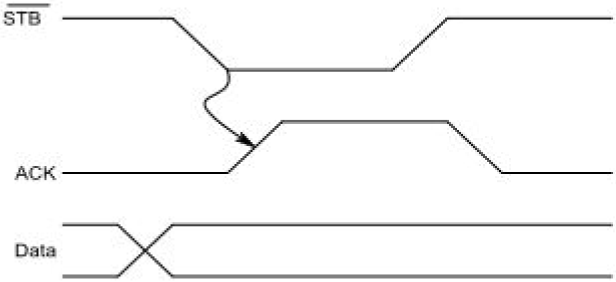


Fig: Single Handshaking

- The peripheral outputs some data and send STB signal to MP. "Here is the data for you."
- MP detects asserted STB
- signal, reads the data and sends an acknowledge signal (ACK) to indicate data

has been read and peripheral can send next data. “I got that one, send me another.”

- MP sends or receives data when peripheral is ready.

Double Handshaking:

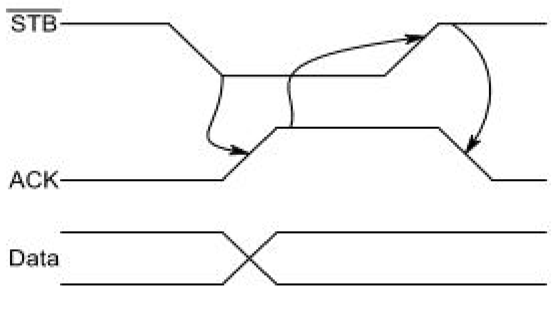


Fig: Double Handshaking

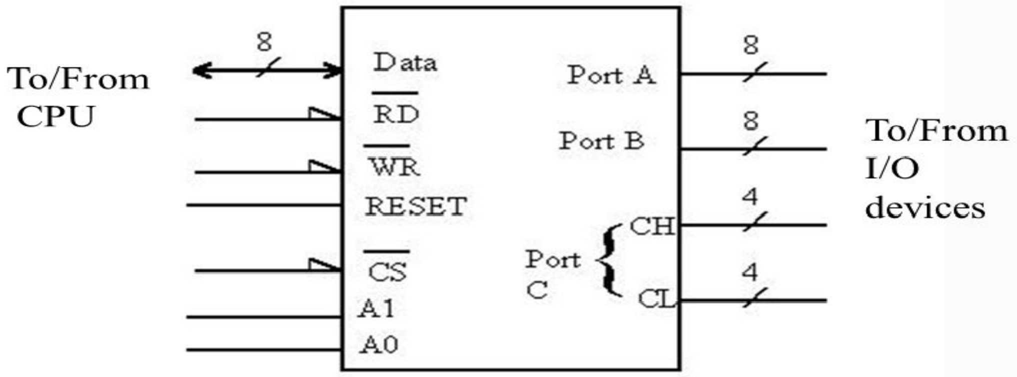
- The peripheral asserts its STB line low to ask MP “Are you ready?”
- The MP raises its ACK line high to say “I am ready”.
- Peripheral then sends data and raises its STB line low to say “Here is some valid data for you.”
- MP then reads the data and drops its ACK line to say, “I have the data, thank you, and I await your request to send the next byte of data.”

7.4 8255A (PPI) and its working principal

The 8255A is a widely used, programmable, parallel I/O device. It is a Programmable peripheral Interface (PPI) which acts as an interface between peripheral and microprocessor and controls the flow of data. It can be programmed to transfer data under various conditions, from simple I/O to Interrupt I/O. It is flexible versatile, and economical (When multiple I/O ports are required), but somewhat complex. It is an important general-purpose I/O device that can be used with almost any microprocessor.

The 8255A has 24I/O pins that can be grouped primarily into two 8-bit parallel ports: **A & B**, with the remaining eight bit as port **C**. The eight bits of port C can be used as individual bits or be grouped in two 4 bits ports: $\mathbf{C}_{upper}$ ($\mathbf{C}_{u}$) and $\mathbf{C}_{Lower}$($\mathbf{C}_{L}$), as in the given figure 7.4(a) The functions of these ports are defined by writing a control word in the control register. Figure 7.4(b) shows all the function of 8255A, classified according to two modes: the Bit Set/Reset (BSR) mode and the I/O mode. The BSR mode is used to set or reset the bits in port C. The I/O mode is further divided into three modes: Mode 0, Mode 1, and Mode 2. In Mode 0, all ports function as simple I/O ports. Mode 1 is a handshake mode whereby ports A and /or B use bits from port C as a handshake signal.

Fig7.4(a): 8255A I/O ports



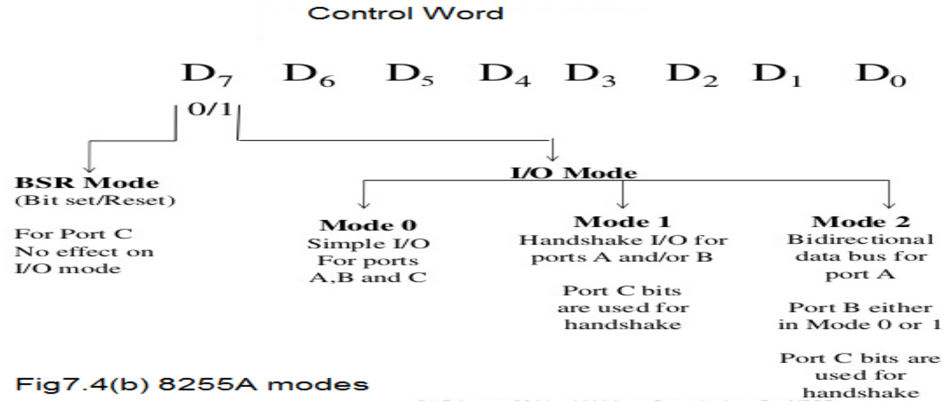


Fig7.4(b) 8255A modes

- **Block Diagram of 8255A**

Explain with block diagram working of 8255A PPI

Below Figure shows the internal block diagram of 8255A. It consists of data bus buffer, control logic, Group A and Group B controls, two 8-bit ports (A&B), two 4bit ports (C_u and C_L).

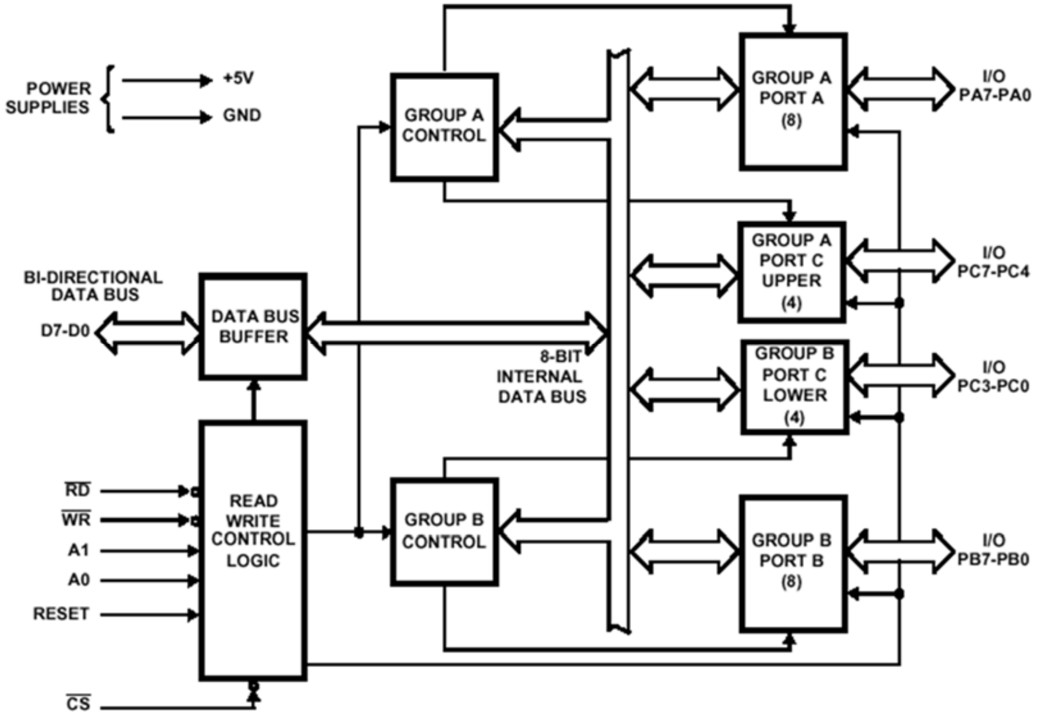


Fig:7.4 (a): Functional diagram of 8255A

1. **Data Bus Butter:** This tri-state bi-directional buffer is used to interface the internal data lines of 8255 to the system data bus. Input or Output instructions executed by the CPU either Read date from or Write data into the buffer. 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.

2. **Control Logic**: 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 (Group A control and Group B control). It issues appropriate enabling signals to access the required data/control words or status word. The input pins for the control logic section are described here.

3. **Group A and Group B Controls:** Each of the Group A and Group B control blocks receives control words from the CPU and issues appropriate commands to the ports associated with it. 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.

4. **Port A:** This has an 8-bit latched and buffered output and an 8-bit input latch. It can be programmed in three modes: mode 0, mode 1 and mode 2.

5. **Port B:** This has an 8-bit data I/O latch/ buffer and an 8-bit data input buffer. It can be programmed in mode 0 and mode 1.

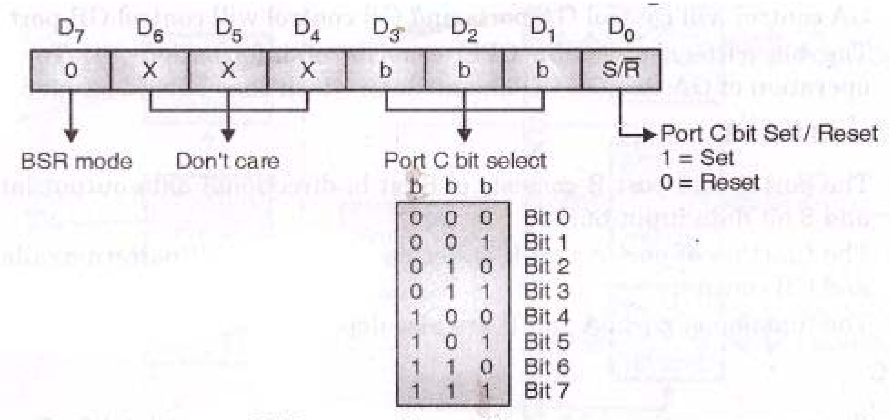
6. **Port C:** This has one 8-bit unlatched input buffer and an 8-bit output latch/buffer. Port C can be spitted into two parts and each can be used as control signals for ports A and B in the handshake mode. It can be programmed for bit set/reset operation.

- **Modes of Operations**

There are two basic modes of operations of 8255A:

BSR mode-

1. The BSR mode is a port C bit set/reset mode.
2. The individual bit of port C can be set or reset by writing control word in the control register.
3. The control word format of BSR mode is as shown in the figure below



b	b	b	
0	0	0	Bit 0
0	0	1	Bit 1
0	1	0	Bit 2
0	1	1	Bit 3
1	0	0	Bit 4
1	0	1	Bit 5
1	1	0	Bit 6
1	1	1	Bit 7

BSR control word format

Fig 7.4(b): Modes of operations

4. The pin of port C is selected using bit select bits [b b b] and set or reset is decided by bit S/R̄.
5. The BSR mode affects only one bit of port C at a time. The bit set using BSR

mode remains set unless and until you change the bit. So, to set any bit of port C, bit pattern is loaded in control register.

6. If a BSR mode is selected it will not affect I/O mode.

I/O modes

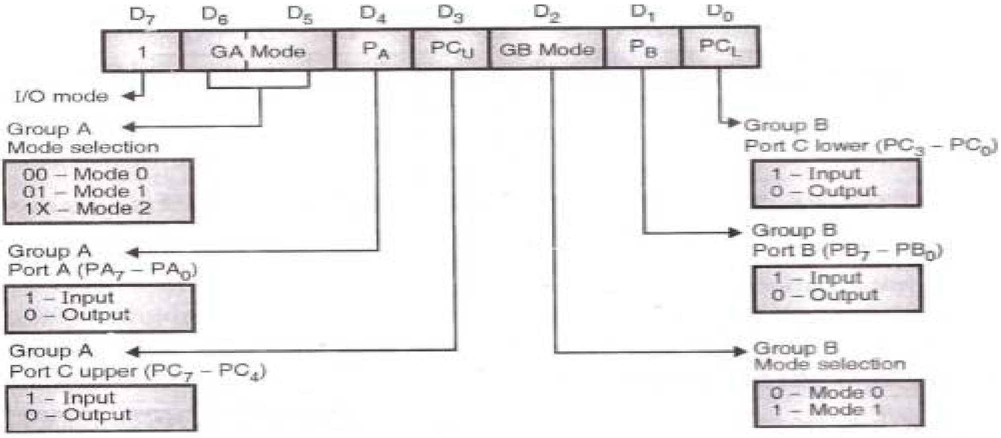
There are three I/O modes of operation:

Mode 0- Basic I/O

Mode 1- Strobed I/O

Mode 2- Bi-directional I/O

The I/O modes are programmed using control register. The control word format of I/O modes is as shown in the figure 7.4(e) below:



I/O modes control word format

Function of each bit is as follows:

1. D7 – When the bit D7 = 1 then I/O mode is selected, if D7=0, then BSR mode is selected. The function of bits D0to D6 is independent on mode (I/O mode or BSR mode).
2. D6 and D5 – In I/O mode the bits D6 and D5 specifies the different I/O modes for group A i.e. Mode 0, Mode 1 and Mode 2 for port A and port C upper.
3. D4 and D3 – In I/O mode the bits D4 and D3 selects the port function for group A. If these bits = 1 the respective port specified is used as input port. But if bit = 0, the port is used as output port.

4. D2 – In I/O mode the bit D2 specifies the different I/O modes for group B i.e. Mode 0 and Mode 1 for port B and port C lower.
5. D1 and D0 – In I/O mode the bits D1 and D0 selects the port function for group B. If these bits = 1 the respective port specified is used as input port. But if bit = 0, the port is used as output port.

All the 3 modes i.e. Mode 0, Mode 1 and Mode 2 are only for group A ports, but for group B only 2 modes i.e. Mode 0 and Mode 1 are provided. When 8255 is reset, it will clear control word register contents and all the ports are set to input mode. The ports of 8255 can be programmed for other modes by sending appropriate bit pattern to control register.

Control Word

The content of the control register is called control word, specify an I/O function for each port. This register can be accessed to write a control word when A_0 and A_1 are at logic 1. You can see the control word format below:

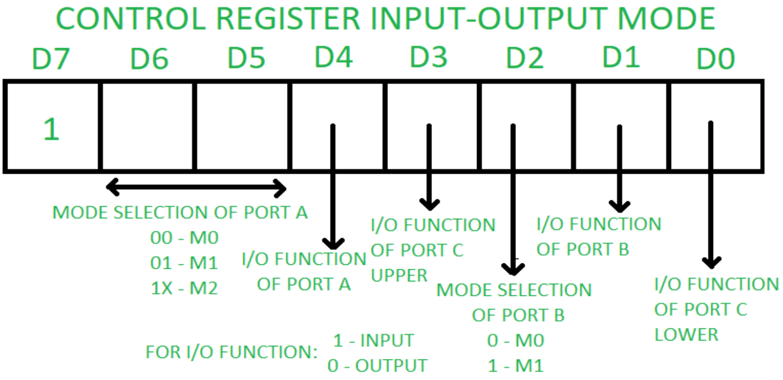


Fig 7.4(c): Control word format

7.5. RS-232: Introduction, pin configuration (9 pin and 25 pin)

In communications, RS-232 is a standard for serial communication transmission of data RS stands for Recommended Standards. It formally defines the signals connecting between a DTE (data terminal equipment) such as a computer terminal, and a DCE (data circuit-terminating equipment or data communication equipment), such as a modem. The RS-232 standard is commonly used in computer serial ports. The standard defines the electrical characteristics and timing of signals, the meaning of signals, and the physical size and pin out of connectors.

a. 9 pin RS-232 pin configurations

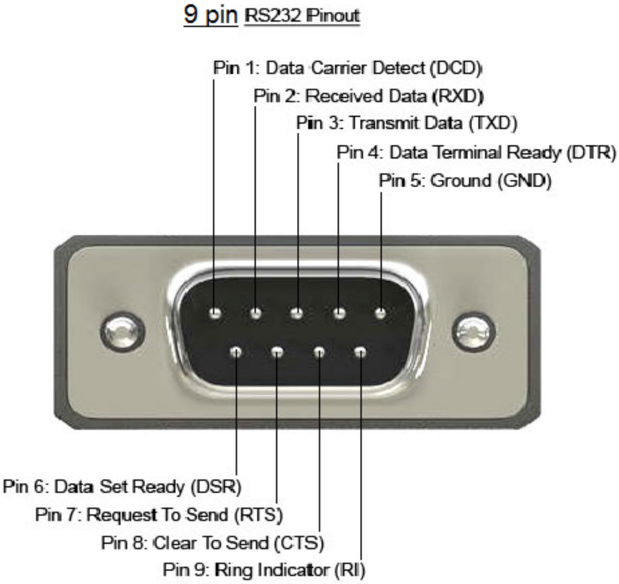

9 pin RS232 Pinout
Pin 1: Data Carrier Detect (DCD)
Pin 2: Received Data (RXD)
Pin 3: Transmit Data (TXD)
Pin 4: Data Terminal Ready (DTR)
Pin 5: Ground (GND)
Pin 6: Data Set Ready (DSR)
Pin 7: Request To Send (RTS)
Pin 8: Clear To Send (CTS)
Pin 9: Ring Indicator (RI)

Fig 7.5 (a) : 9 Pin configurations of RS-232

b. 25 pin RS-232

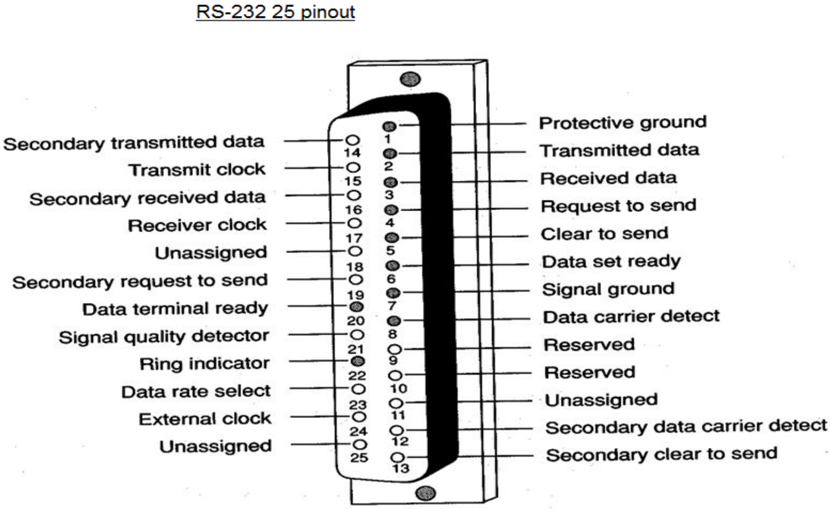
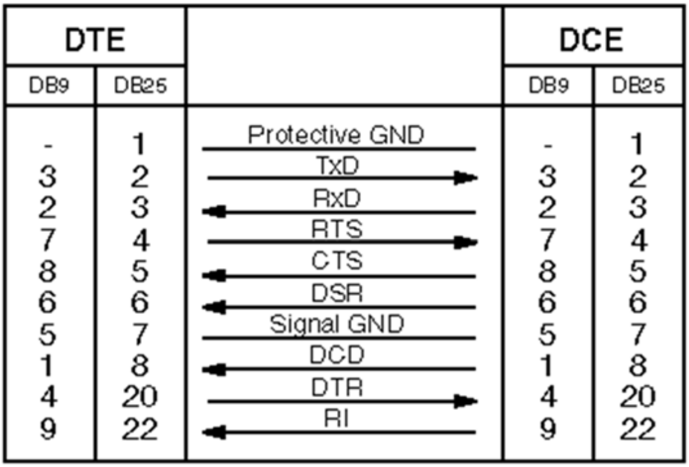


Fig 7.5(b): 25 Pin configurations of RS-232



DTE			DCE	
DB9	DB25		DB9	DB25
-	1	Protective GND	-	1
3	2	TxD →	3	2
2	3	← RxD	2	3
7	4	RTS →	7	4
8	5	← CTS	8	5
6	6	← DSR	6	6
5	7	Signal GND	5	7
1	8	← DCD	1	8
4	20	DTR →	4	20
9	22	← RI	9	22

Fig 7.5(c): RS-232 interface

2071(2015)

Microprocessor Questions & Solutions

Group-A

1. Define Microprocessor?

A microprocessor is an integrated computer circuit that performs all the functions of a CPU.

2. What are the three states of SAP-I fetch cycle?

The three states of SAP –I fetch cycle is T1, T2 and T3. And the rest T4, T5 and T6 are for execute cycle.

3. Define instruction cycle.

An ***instruction cycle*** (sometimes called a fetch–decode–execute ***cycle***) is the basic operational process of a computer. It is the process by which a computer retrieves a program ***instruction*** from its memory, determines what actions the ***instruction*** dictates, and carries out those actions.

4. List out all the valid registers pair of 8085.

Ans. The valid registers pairs are:

B-C, D-E & H-L registers pair.

5. For what purpose STA instruction is used in 8085 microprocessor?

Ans. STA instruction is used to store the content of Accumulator in the given address location.

6. What is DMA?

Ans. DMA stands for Direct Memory Access. It is a technique in which data is directly transfer from I/O devices to memory without the involvement of CPU.

7. What is meant by Synchronous Mode of communication?

Ans. ***Synchronous*** serial ***communication*** describes a serial ***communication*** protocol in which "data is sent in a continuous stream at a constant rate.

Group-B

9. List out two major differences between SAP-I and SAP-II architecture.

Ans. SAP-I: a.) It has 8-bit bus.

b) PC is 4-bit.

C) It has 5 instruction sets.

d) It does not have flags.

e) It has 16-byte memory

SAP-II a) It has a 16-bit bus

b) PC is 16-bit.

c) It has 42 instruction sets.

d) It has 2 flags.

e) It has 62 KB memory.

10. What is Microinstruction?

Ans. A single instruction in microcode. It is the most elementary instruction in the computer, such as moving the contents of a register to the arithmetic logic unit (ALU). It takes several ***microinstructions*** to carry out one complex machine instruction.

11. What is the function of Program Counter?

Ans. A ***program counter*** is a register in a computer processor that contains the address (location) of the instruction being executed at the current time. As each instruction gets fetched, the ***program counter*** increases its stored value by 1. In 8085 PC is of 16 bit.

12. Define T-States.

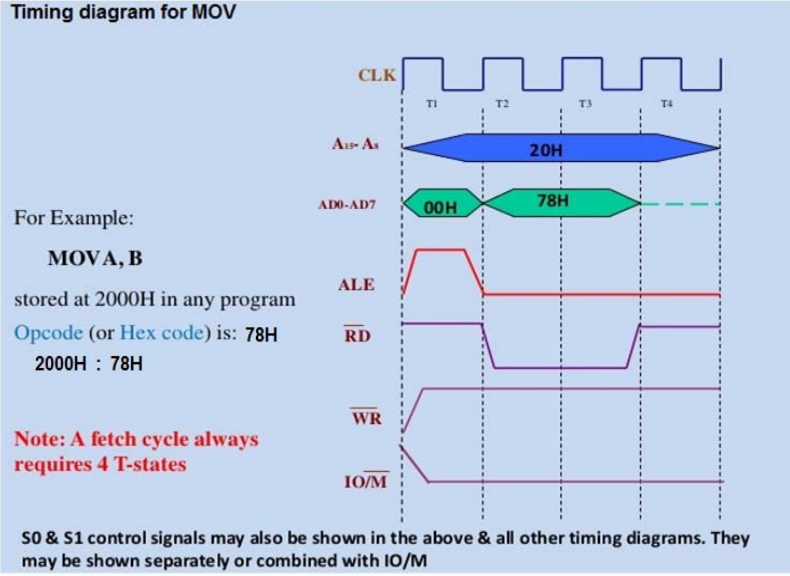
Ans. ***T-state*** is defined as one subdivision of operation performed in one clock period. These subdivisions are internal ***states*** synchronized with the system clock, and each ***T-state*** is precisely equal to one clock period. ***T-state*** is nothing but refers to one clock cycle.

13. Define Interrupt.

Ans. An ***interrupt*** is a signal from a device attached to a computer or from a program within the computer that requires the operating system to stop and figure out what to do next. Interrupts are served on the basis of their priority.

14. Sketch the timing diagram of MOV A, B.

Ans. MOV A, B has Opcode fetch machine cycle only



Timing diagram for MOV

For Example:

MOVA, B

stored at 2000H in any program

Opcode (or Hex code) is: 78H

2000H : 78H

Note: A fetch cycle always requires 4 T-states

S0 & S1 control signals may also be shown in the above & all other timing diagrams. They may be shown separately or combined with IO/M

15. List out the different addressing modes with examples.

Ans. The different addressing modes are:

a) Immediate addressing mode, example: MVI B, 06H

b) Register addressing mode, example: MOV A, B

c) Direct addressing mode, example: STA 2000H

d) Indirect addressing mode, example: MOV M, A

e) Implied addressing mode, example: CMP.

16. Write down any four types of branching instructions with their examples.

Ans. The four types of branching instruction are:

a) Jump instruction: e.g. JNZ, JNC

b) Callinstructions e.g. CP, CNC

c) Returns instructions, e.g. RNC, RNZ

d) Restart instructions e.g. RST.

17. WAP to load 1B H in register D and B5 H in register B, increment the content of B and decrement the content of D by 1 and subtract the content of D from the content of B, display the result in output port 01H.

Ans.

```
MVI D, 1BH     // load D register with 1B H
MVI B, B5H     // Load B with B5H
INR B          // increment the value of B
DCR D          // decrement the value of D
MOV A, B       // copy the content of B to A
SUB D          // subtract content of D from B
OUT 01H        //Store the result in output port no.1
HLT            // Halt, terminate the program
```

18. Write down two differences between Maskable and Non-maskable interrupts.

Ans. Maskable interrupts (a) These interrupts can be masked(disable) by writing proper control word in control registers.

(b) Example: RST 7.5, RST 6.5, RST 5.5, INTR

Non-Maskable Interrupts (a) These interrupts cannot be masked (disabled) without being serviced.

(c) Example: TRAP

19. Draw a block diagram of 8259 interrupt controller.

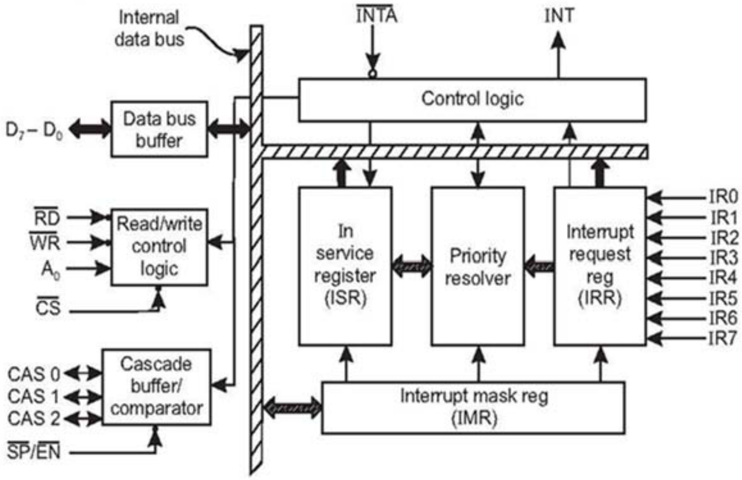


Fig. 9e.2: 8259 Functional block diagram (*Source:* Intel Corporation)

20. Write any two differences between Asynchronous and synchronous mode of Operations.

Ans. Synchronous Mode: (a) Sender and receiver uses the same clock signal

(b) Supports high data transfer rate

(c) Data is transmitted in the form of blocks or Frame.

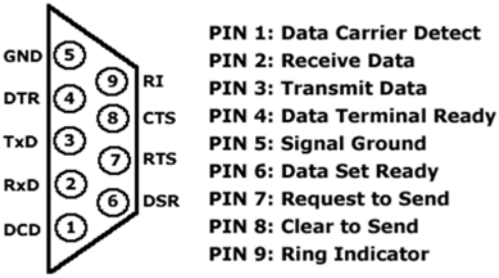
Asynchronous Mode: a.) Sender and receiver clock signal is random.

(b) Data transfer is slow

(c) Data is transmitted 1 byte or character at a time.

21. Sketch 9-pin RS-232 and write down function od each pin.

RS-232 DB-9 Male Pinout



PIN 1: Data Carrier Detect
PIN 2: Receive Data
PIN 3: Transmit Data
PIN 4: Data Terminal Ready
PIN 5: Signal Ground
PIN 6: Data Set Ready
PIN 7: Request to Send
PIN 8: Clear to Send
PIN 9: Ring Indicator

Group-C

22. Give a brief explanation on evolution of microprocessor

Ans. Evolution of Microprocessors

4-bit Processor:The first microprocessor was introduced in 1971 by Intel Corp. It was named Intel 4004 as it was a 4-bit processor. It was a processor on a single chip. It could perform simple arithmetic and logic operations such as addition, subtraction, boolean AND andboolean OR. It had a control unit capable of performing control functions like fetching an instruction from memory, decoding it, and generating control pulses to execute it. It was able to operate on 4 bits of data at a time.This first microprocessor was quite a success in industry. Soon other microprocessors were also introduced. Intel introduced the enhanced version of 4004, the 4040.

8-bit processor: The first 8 bit microprocessor which could perform arithmetic and logic operations on 8-bit words was introduced in 1973 again by Intel. This was Intel 8008 and was later followed by an improved version, Intel 8088.

16-bit processor: The 8-bit processors were followed by 16-bit processors. They are Intel 8086 and 80286.

32- bit processor: The 32-bit microprocessors were introduced by several companies but the most popular one is Intel 80386.

Intel Pentium series: Instead of 80586, Intel came out with a new processor namely Pentium processor. Its performance is closer to RISC performance. Pentium was followed by Pentium Pro CPU. Pentium Pro allows allow multiple CPUs in a single system in order to archive multiprocessing. The MMX extension was added to Pentium Pro and the result was Pentium II. The lowcost version of Pentium II isCeleron.

23. Sketch block diagram of SAP-II and explain each functional block.

Ans. Please refer to class notes.

24. List different types of flags in 8085 and explain it.

Ans. There are 5 different flags:

Carry Flag (CF): When two 8/16 bit number addition the result is generate 9/17 bit number then the carry flag is set.

Auxiliary flag (AF): This flag is active in BCD number. When Two 8 bit number addition the carry is generate by D3 bit to D4 bit theAuxiliary flag is set.

Parity Flag (PF): This flag is indicate the number is even or odd. The numbers of one's in result is even 0,2,4,6,8 then parity flag is set and the number of one's is odd 1,3,5,7 the parity flag is reset.

Zero Flag (ZF): This flag is indicate the result is zero or non-zero. when two 8 bit number addition then the result is zero then zero flag is set and the result is non-zero then the zero flag is reset.

Sign Flag (SF): This flag is indicate whether the number is negativeor positive. The MSB in result is one(1)then the sign flag isset or the result is negative And the MSB in result

Zero(0) then the sign flag is reset or the result is positive.

26. Write an APL to find the largest number in a data array between C001H to C003H and store the result in C0C0H.

Ans.

```
LXI H, C000H        // load HL as a pointer
MOV C, M            // copy the content of HL to C
DCR C               // Decrement the value of C
INX H               // Increment the value of HL
MOV A, M            // Copy the content of HL to A
INX H               // Increment the HL pair
L1:  CMP M          // Compare the content of HL with A
JNC L2              // jump if not carry to L2
```

```
MOV A, M          // Copy from HL to A
L2: INX H         // Increment HL
DCR C             // Decrement C
JNZ L1            // Jump if not zero to L1
STA C0C0H         // Store the content of A to C0C0H
HLT               // halt
```

27. Explain the DMA controller interfacing with 8085.

Ans. Please refer to class notes.

28. What is Handshaking? Explain its two types.

Ans. Handshaking is a I/O control method to synchronize I/O devices with the microprocessor. As many I/O devices accepts or release information at a much slower rate than the microprocessor, this method is used to control the microprocessor to work with a I/O device at the I/O devices data transfer rate.

Two types are:

a) **Single Handshake :** In single handshake, a peripheral device first sends a "Strobe Signal" to the microprocessor to indicate that it is ready to send a data.

* The microprocessor upon detecting the strobe signal, opens up its input port and receive the data.

*After receiving data it sends an "Acknowledge signal" to the peripheral indicating transmission has been completed.

b) **Double Handshake:**

* In double handshakefirst the peripheral sends a strobe signal, the microprocessor sends an Acknowledge signal to indicate it is ready to receive the data.

* After that data is received.

* After sending data, the peripheral sends a strobe signal to indicate data transmission completion.

2072(2016)

Group-A

1. Name the first 8-bit microprocessor.

Ans. Intel 8008

2. What is the primary purpose of SAP-I?

Ans. Its ***primary purpose*** is to develop a basic understanding of how a microprocessor works,

3. Define T-State.

Ans. ***T-state*** is ***defined*** as one subdivision of operation performed in one clock period.

4. Write any two valid register pair of 8085.

Ans. H-L & D-E.

5. Give an example of two byte and three-byte instruction of 8085.

Ans. Two byte: MVI B, 06H

Three byte: STA 4001H

One byte: MOV A, B

6. State Serial Communication.

Ans. S**erial communication** is the process of sending data one bit at a time, sequentially, over a **communication** channel or computer bus.

7. Define DMA.

Ans. Please refer to q.6(2071)

8. Why is binary display important in SAP-I.

Ans. The binary display is a row of eight light emitting diodes (LED's). Because each LED connects to one flip-flop of the output port, the binary display shows us the content of the output port. Therefore, after we transferred an answer from the accumulator to the output port, we can see the answer in binary form.

9. List out the instructions SET of SAP-I and define.

Ans. The five instructions set are LDA, ADD, SUB, OUT, HLT.

10. Draw a block diagram of 3 states of fetch cycle.

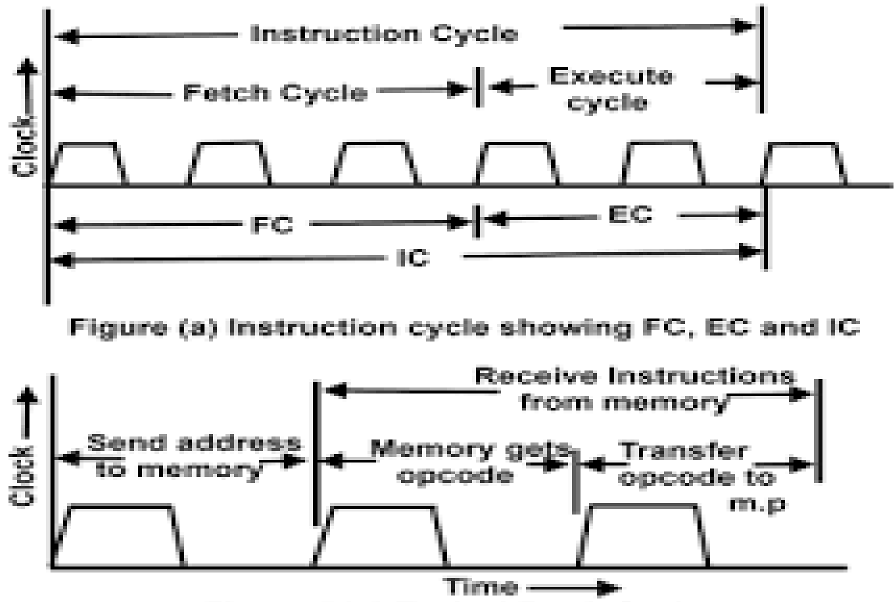
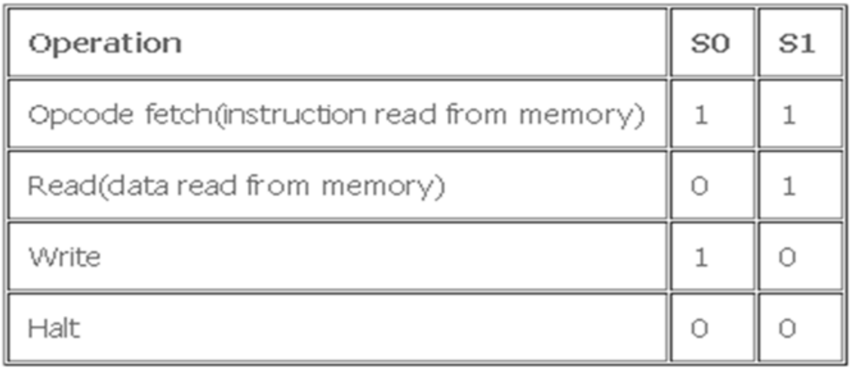


Figure (a) Instruction cycle showing FC, EC and IC

Figure (b) A Typical Fetch Cycle

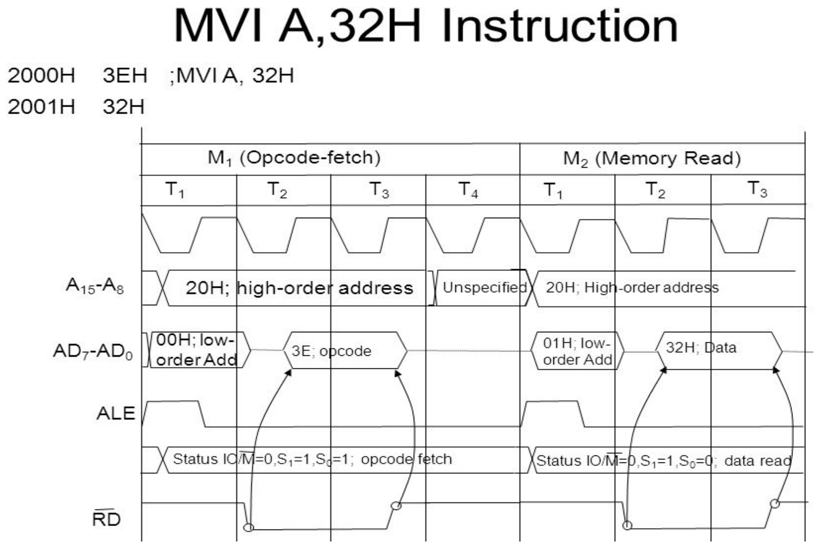
Figure 4

12. State the operation of status signal S_0 and S_1



Operation	S0	S1
Opcode fetch(instruction read from memory)	1	1
Read(data read from memory)	0	1
Write	1	0
Halt	0	0

11. Sketch the timing diagram of MVI A, 32H



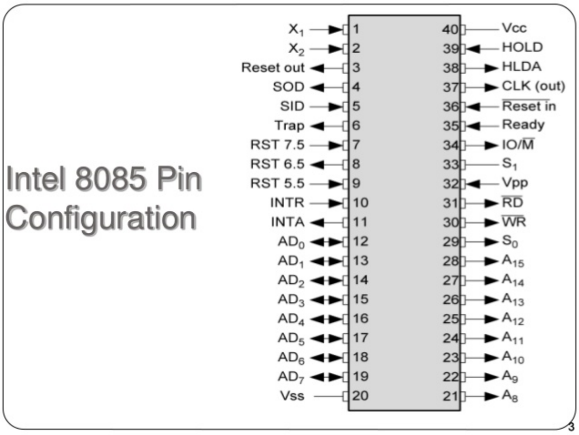
MVI A,32H Instruction

2000H 3EH ;MVI A, 32H
2001H 32H

13. Define Instruction Cycle and Machine Cycle

Ans. An **instruction cycle** (sometimes called a fetch–decode–execute **cycle**) is the basic operational process of a computer. It is the process by which a computer retrieves a program **instruction** from its memory, determines what actions the **instruction** dictates, and carries out those actions.The steps performed by the computer processor for each machine language instruction received. The **machine cycle** is a 4 process cycle that includes reading and interpreting the machine language, executing the code and then storing that code.

14. Draw a clean pin configuration of 8085



Model Question Paper

Group A

Very Short Answer Questions (Attempt any ten) 10x1=10

1. Name the first Microprocessor?
2. Define Registers.
3. How data flow between MPU and Memory?
4. What is Instruction Cycle?
5. Name the first machine cycle for execution of any instructions.
6. Define Op-code and Operands.
7. Name the flags associated with 8085.
8. Define the instruction CMP.
9. What is the size of the instruction STA 2050H?
10. Define Interrupts.
11. Which hardware interrupt has the highest priority?
12. In which mode of transmission the data is transferred in frames?
13. Define Control Word.

Group B

Short Answer Questions (Attempt any five) 5x2=10

1. What are the Basic organization of microprocessor?
2. Draw the Bus Architecture of 8085.
3. Calculate the execution time for STA 2050H, considering the clock frequency of microprocessor be 3.125 MHZ.
4. What are the different addressing modes in 8085?
5. WAP to multiply two 8-bit numbers.
6. What are the different priority modes of 8259A.
7. Draw the labelled pin out diagram of 9 pin RS-232.

Group C

Long Questions(attempt any five) 5x4=20

1. Explain different processor unit.
2. Draw timing diagram for STA 4001H.
3. Draw the functional block diagram of 8085.
4. Explain different types of instructions.
5. Explain the working of DMA along with advantage and applications.
6. Sixteen byte of data is stored in memory locations at 2250H to 225FH. Transfer the entire block of data to new locations starting at 2270H

Sample Programs for practice

Introduction

In all Experimental Laboratory, student will follow following procedural steps:

Step 1: Analyzing/Defining the Problem

The first step in writing a program is to think very carefully about the problem that you want the program to solve. In other words, ask yourself many times, “what do I really want this program to do? “It is good idea to write down exactly what you want the program to do and the order in which you want the program to do it. At this point student should not write down program statements, should write the operations in general terms.

Step 2: Designing the solution/Representing Program Operations:

The formula or sequence of operations used to solve a programming problem is often called the algorithm of the program. Draw flowchart or use pseudo code to represent program which you want to write to solve your problem. In EXPERIMENT it is better to use flowchart.

Step 3: Implementing the Solution

3.1 Define your Constant, Variables & Pointers for the program.

3.2 Finding the right instruction:

After student prepare flowchart of a program, the next step is to determine the instruction statements required to do each part of the program. Student has to remember the complete instruction set for 8085.Each statement in flow chart could be implemented using one or more instructions.

Standard program format for assembly language program:

Student should finally write the program in following format:

Memory Address Label Mnemonics Label Hex Code Comments

2100 start :MVI A,55 3E,55H ; Acc=55

3.3 Constructing the machine codes from 8085 instructions:

Student will refer the table and manually convert its assembly language program into machine code.

Step 4: Loading/Running the solution:

Student will use trainer kit to load his machine code program in RAM and run his program. Student will use debugging tools availed in EXPERIMENT kit to find out the software bugs in the program.

Step 5: Testing the Solution

Test the solution by observing the content of various registers and memory addresses.

This is the standard procedure for all experiments, hence not required to write separate procedure for all experiments. The list of experiments is attached.

Example: Move a block of 8-bit numbers from one place to other place.

Data(H): 37, A2, F2, 82, 57, 5A, 7F, DA, E5, 8B, A7, C2, B8, 10, 19, 98

Step 1: By analyzing the problem statement, you require following things.

i) Block size (How many number of 8-bit numbers you want to move)

ii) Source Memory Pointer

iii) Destination Memory Pointer

Step 2: Designing the solution/Representing Program Operations.

(Flow Chart for block transfer)

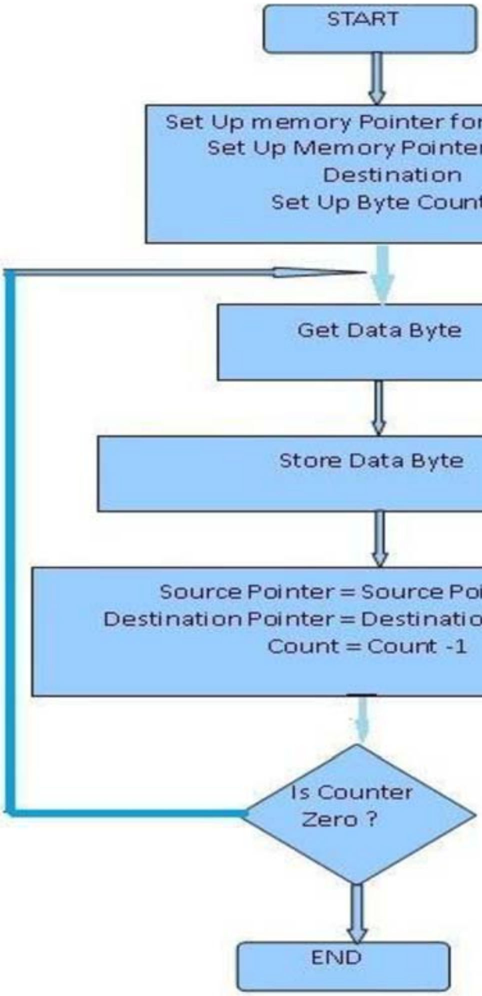

START
Set Up memory Pointer for the Source
Set Up Memory Pointer for the Destination
Set Up Byte Counter
Get Data Byte
Store Data Byte
Source Pointer = Source Pointer + 1
Destination Pointer = Destination Pointer + 1
Count = Count -1
Is Counter Zero ?
END

Fig1 : source: PTU(practical course file for Microprocessor)

Step 3: Implementing the Solution

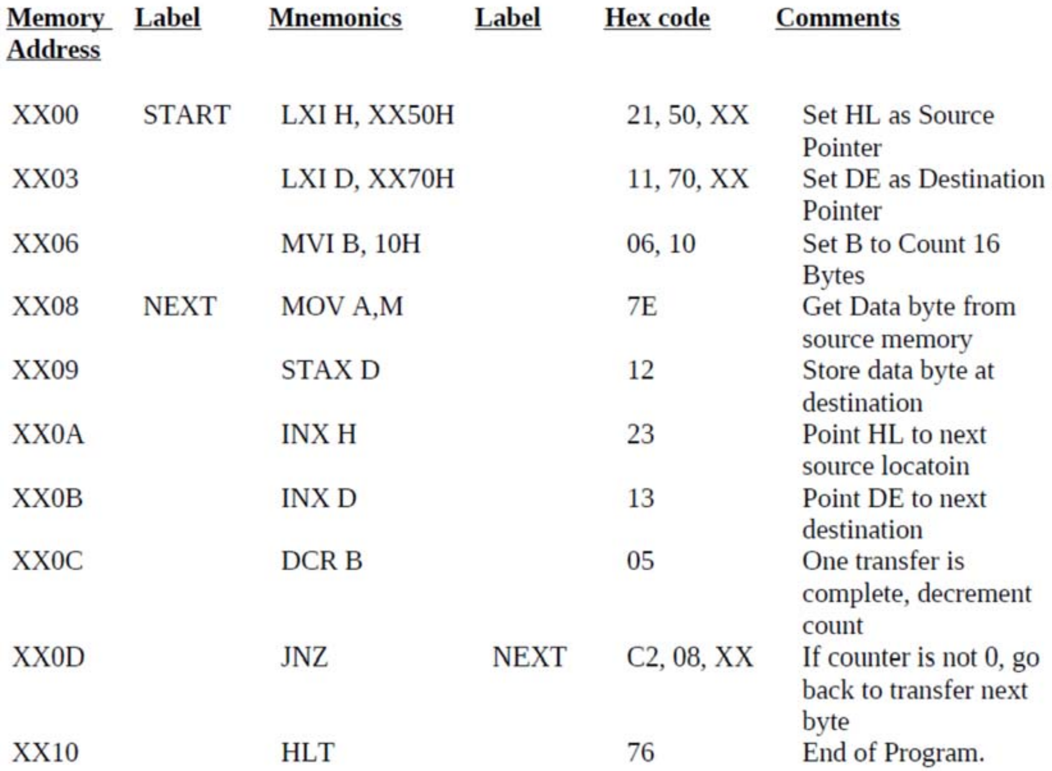
3.1 Define your Constant, Variables & Pointers

3.2 Finding the right instruction:

LXI H, XX50	Set HL pointer for source memory
LXI D, XX70	Set DE pointer for destination memory
MVI B,10	Set B as a byte counter
NEXT:	MOV A,M Get data from source memory
STAX D	Store data in destination memory

INX H	Increment the address value of HL pair by 1
INX D	Get ready for next byte
DCR B	Decrement the count value of register B
JNZ NEXT	Go back to next byte if counter is not equal to 0
HLT	End of program

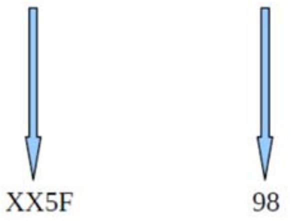
3.3 Constructing the machine codes from 8085 instructions:



Memory Address	Label	Mnemonics	Label	Hex code	Comments
XX00	START	LXI H, XX50H		21, 50, XX	Set HL as Source Pointer
XX03		LXI D, XX70H		11, 70, XX	Set DE as Destination Pointer
XX06		MVI B, 10H		06, 10	Set B to Count 16 Bytes
XX08	NEXT	MOV A,M		7E	Get Data byte from source memory
XX09		STAX D		12	Store data byte at destination
XX0A		INX H		23	Point HL to next source locatoin
XX0B		INX D		13	Point DE to next destination
XX0C		DCR B		05	One transfer is complete, decrement count
XX0D		JNZ	NEXT	C2, 08, XX	If counter is not 0, go back to transfer next byte
XX10		HLT		76	End of Program.

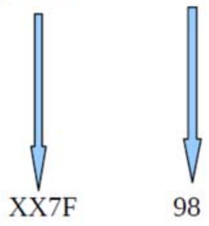
Step 4: Loading/Running the solution:

Data: XX50 37



Step 5: Testing the Solution

A.E (After Execution) XX70 37



Introduction to Microprocessor Trainer Kit

- **Study of HEX Keypad**

Study the functions of the following Keys

SAVE: This command is used to save the contents of specified block ontoa audio cassette for permanent storage.

LOAD: This command is opposite to save command. The contents of audio cassette block is loaded (retrieved back) in the system RAM from a given DS (file name)

CODE: When this command key is pressed the address field remains blank, Data field shows a dot indicating that it expects a code. User is provided with a table of codes, indicating the meaning and Prerequisites of each code. User loads the appropriate code and executes it by pressing EXEC. The monitor branches to the appropriate sub-routines pointed by the code.

STEP: Mere running a program with RUN is done whenever the program development is complete i.e. to *run* a final working program. During the program development stage some sort of aid to execute the part of program at a time and then test its success is required. The STEP command helps you to do the above.

There are two ways of stepping the program.

- ***SINGLE STEPPING****:* to execute single instruction at a time, using STEP command. The STEP command requires a start address, break address and the no.of times the break should occur.
- ***BREAK POINT:***set a software breakpoint *RST1.* This software breakpoint can be done using the *RUN* command. It requires *RST1 (CFH)* to be inserted to a location where you want to break. The disadvantage of this method is that you have to insert and remove *'CF'* and you have to operate in the *RAM* area only.

VI: This key causes immediate recognition of *RST 7.5* interrupt and control passes to location

003C in the monitor. This location has a jump to location 20CE in user *RAM*. You can put any instruction or jump in 20CE to 20D0

- Interrupts must be enabled (EI) instruction.

- RST 7.5 must be unmasked (mask reset by SIM instruction)

RUN: This command is used to execute your programs or subroutines from a Specified address.

EXEC: Pressing EXEC will place the data field contents into the named register and terminate the command.

REG: This command allows you to display and optionally modify the contents of 8085 CPU registers. The various registers are A, B, C, D, E, F, I, H, L, SPH, SPL, PCH, PCL. (H – higher byte, L – lower byte)

RES: On RES, the display shows *MP – 85* as a sign on message, indicating that the monitor is ready to accept a command. Pressing any non-command key generates "Err" message. After "- Err" user can immediately give a valid command.

SET: It is used to SET the address of a required memory location. A dot in the address field of display indicated that the entry will be displayed in the address field.

INC: Pressing INC, first time will shift the dot to the data field of display. Data field will show the contents of the memory location pointed by the address set. One can modify or retain the data field to any value.

DEC: DEC acts as similar to INC, except the address field is decremented, pointing to previous memory locations.

SPH: Stack pointer Register (Higher byte)

SPL: Stack pointer Register (Lower byte)

PCH: Program Counter Register (Higher byte)

PCL: Program Counter Register (Lower byte)

0 – F: Hex Keypad

H, L: Registers H & L

- **Study of following Devices**

  1. IC – 8251 (Programmable Synchronous and asynchronous serial data transmitter)
  2. IC – 8253 (Programmable interval Timer /Counter)
  3. IC – 8255 (Programmable Parallel IO Device)
  4. IC – 8279 (Keyboard Display Interface)
  5. IC – 6264 (RAM)
  6. IC – 2764 (EPROM)
  7. ADC
  8. DAC
  9. IC – 8085 (Microprocessor)

- **Study of Memory Address Space**

  ROM :0000 – 1FFF

  RAM :2000 – 3FFF

Memory Address Space Used by Firmware Program: 2000 – 20FF

Students should not use this address range for their program or do not modify the content of these locations. Memory is expandable to 64K Bytes by interfacing appropriate RAM IC in the empty sockets.

- **Crystal Frequency**

  Crystal Frequency = 6.144 MHz

- **Study of Onboard Interfaces**

  The kit has following onboard Interfaces:

  - Parallel I/O using 8255
  - Serial I/O using 8251/8253
  - Keyboard/Display using 8279
  - ADC/DAC using 8255 / Latch -373

- **Study of Interrupts**

The Kit uses following interrupts

- RST 7.5 - VI

- RST 5.5 - 8279

- NMI - Counter 0 output

- RST 6.5 - Used to implement Single Step

- INTR - Used to implement Single Step

How to run/execute the program on kits

SI-Single step execution

. (stop) .(stop)

GO (means run) SI

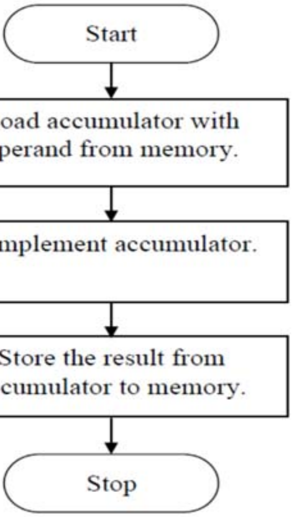
Enter the beginning address of prog Enter the beginning address of prog

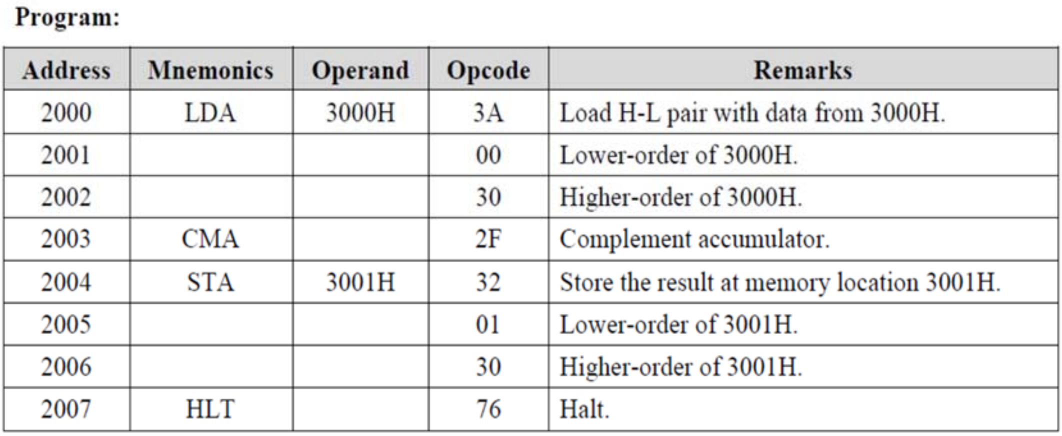
Press next, next, next……… tills the end of prog.

(stop running) (stop running)

Examine mem/reg. Examine mem/reg.

Program 1: WAP to find 1's complement of an 8-bit number


Start
Load accumulator with operand from memory.
Complement accumulator.
Store the result from accumulator to memory.
Stop



Program:

Address	Mnemonics	Operand	Opcode	Remarks
2000	LDA	3000H	3A	Load H-L pair with data from 3000H.
2001			00	Lower-order of 3000H.
2002			30	Higher-order of 3000H.
2003	CMA		2F	Complement accumulator.
2004	STA	3001H	32	Store the result at memory location 3001H.
2005			01	Lower-order of 3001H.
2006			30	Higher-order of 3001H.
2007	HLT		76	Halt.

Explanation: Let us assume that the operand stored at memory location 3000H is 85H.

- The operand is moved to accumulator from memory location 3000H.
- Then, its complement is found by using CMA instruction.
- The result is stored at memory location 3001H.

Output:

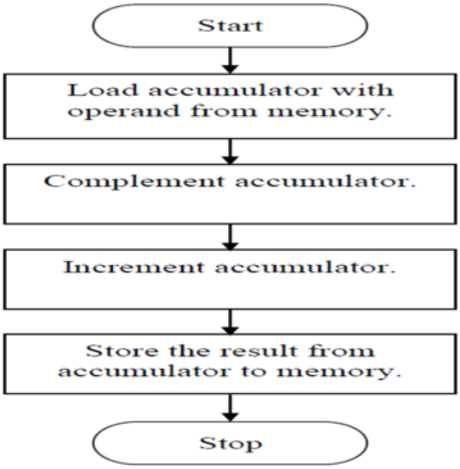
Before Execution:

3000H: 85H

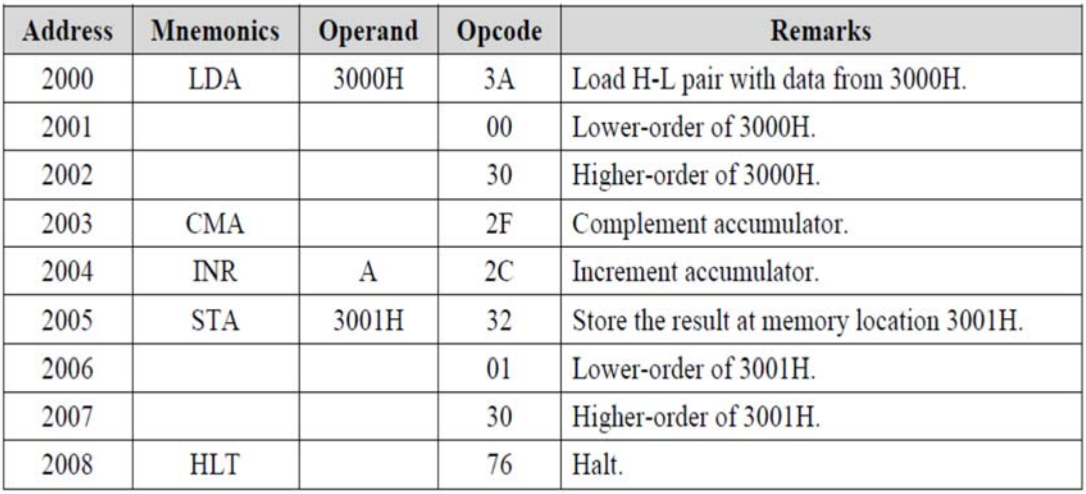
After Execution:

3001H: 7AH

Program 2: WAP to find 2's Complement of 8-bit number


Start
Load accumulator with operand from memory.
Complement accumulator.
Increment accumulator.
Store the result from accumulator to memory.
Stop

Program:



Address	Mnemonics	Operand	Opcode	Remarks
2000	LDA	3000H	3A	Load H-L pair with data from 3000H.
2001			00	Lower-order of 3000H.
2002			30	Higher-order of 3000H.
2003	CMA		2F	Complement accumulator.
2004	INR	A	2C	Increment accumulator.
2005	STA	3001H	32	Store the result at memory location 3001H.
2006			01	Lower-order of 3001H.
2007			30	Higher-order of 3001H.
2008	HLT		76	Halt.

Explanation

- This program finds the 2's complement of an 8-bit number stored in memory location 3000H.
- Let us assume that the operand stored at memory location 3000H is 85H.
- The operand is moved to accumulator from memory location 3000H.
- Then, its complement is found by using CMA instruction.
- One is added to accumulator by incrementing it to find its 2's complement.
- The result is stored at memory location 3001H.

Output

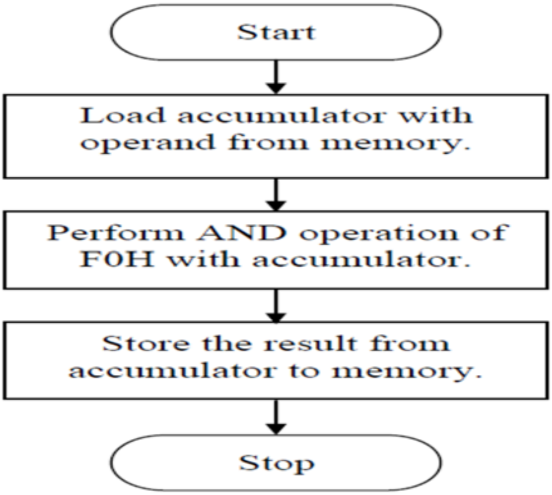
Before Execution

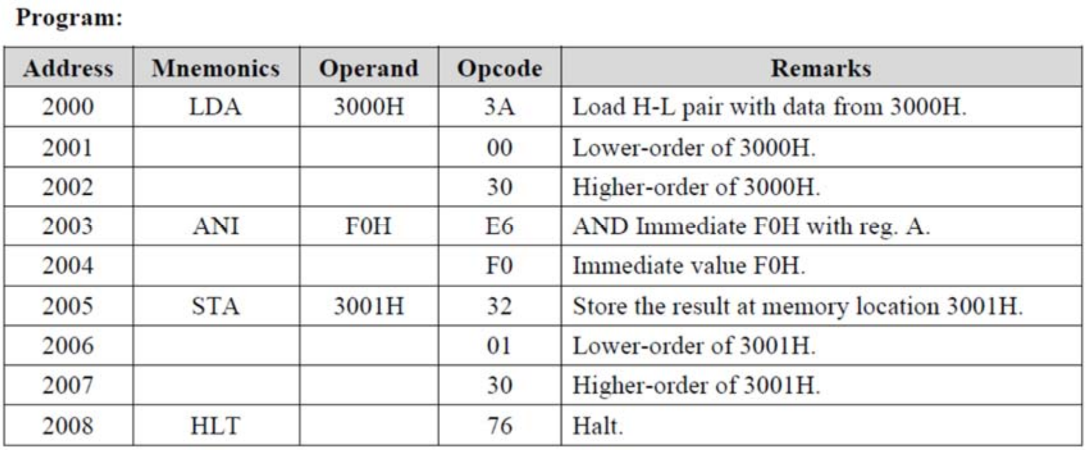
3000H: 85H

After Execution

3001H: 7BH

Program 3: WAP to mask lower nibble of an 8-bit number


Start
Load accumulator with operand from memory.
Perform AND operation of F0H with accumulator.
Store the result from accumulator to memory.
Stop



Program:

Address	Mnemonics	Operand	Opcode	Remarks
2000	LDA	3000H	3A	Load H-L pair with data from 3000H.
2001			00	Lower-order of 3000H.
2002			30	Higher-order of 3000H.
2003	ANI	F0H	E6	AND Immediate F0H with reg. A.
2004			F0	Immediate value F0H.
2005	STA	3001H	32	Store the result at memory location 3001H.
2006			01	Lower-order of 3001H.
2007			30	Higher-order of 3001H.
2008	HLT		76	Halt.

Explanation

- This program masks the lower nibble of an 8-bit number stored in memory location 3000H.

- Let us assume that the operand stored at memory location 3000H is 45H.
- The operand is moved to accumulator from memory location 3000H.
- Then, AND operation of F0H is performed with accumulator. This results in the masking of lower nibble.
- The result is stored at memory location 3001H.

Output:

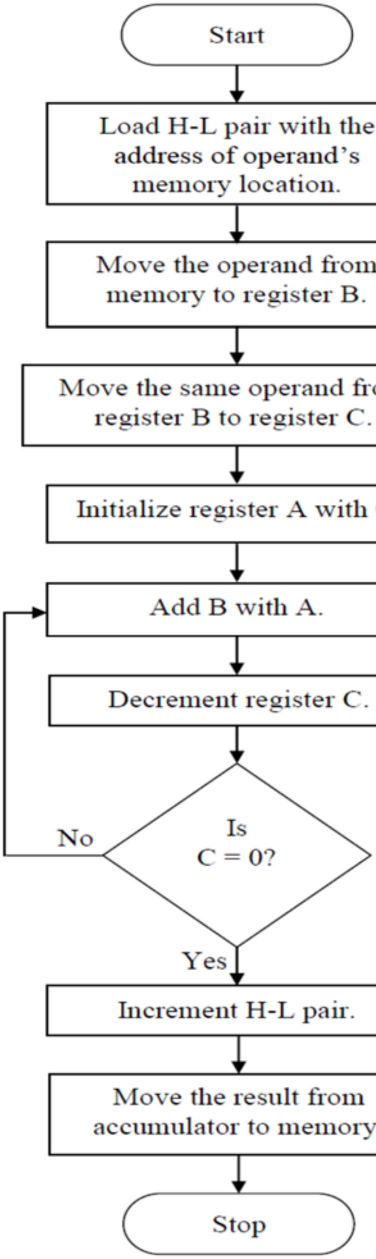
Before Execution

3000H: 45H

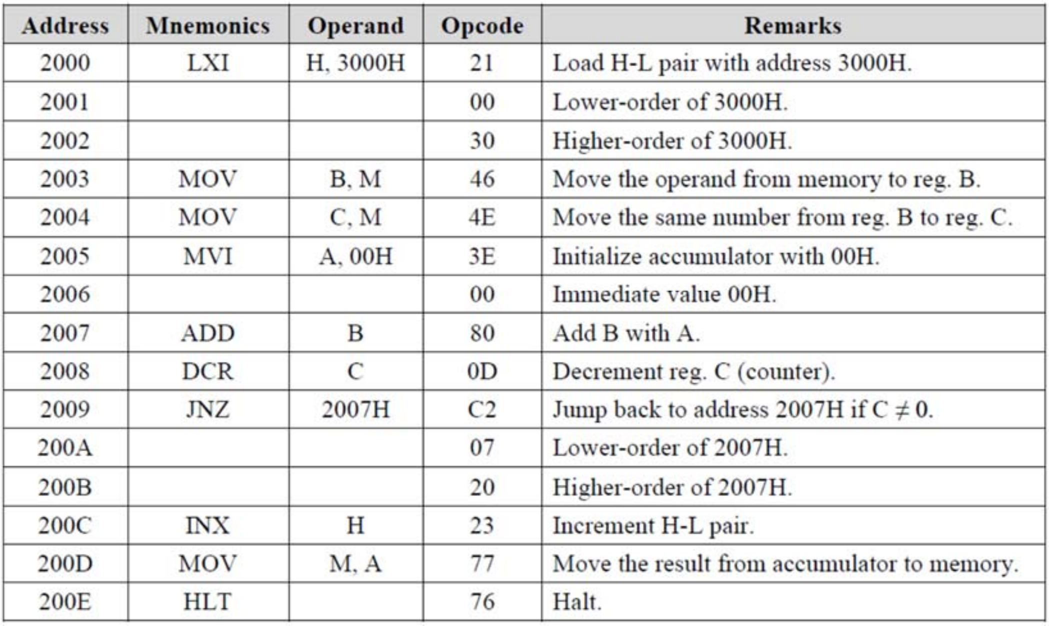
After Execution

3001H: 40H

Program 4: WAP to find the square of an 8-bit number


Start
Load H-L pair with the address of operand's memory location.
Move the operand from memory to register B.
Move the same operand from register B to register C.
Initialize register A with 0.
Add B with A.
Decrement register C.
Is C = 0?
No
Yes
Increment H-L pair.
Move the result from accumulator to memory.
Stop

Program:



Address	Mnemonics	Operand	Opcode	Remarks
2000	LXI	H, 3000H	21	Load H-L pair with address 3000H.
2001			00	Lower-order of 3000H.
2002			30	Higher-order of 3000H.
2003	MOV	B, M	46	Move the operand from memory to reg. B.
2004	MOV	C, M	4E	Move the same number from reg. B to reg. C.
2005	MVI	A, 00H	3E	Initialize accumulator with 00H.
2006			00	Immediate value 00H.
2007	ADD	B	80	Add B with A.
2008	DCR	C	0D	Decrement reg. C (counter).
2009	JNZ	2007H	C2	Jump back to address 2007H if C ≠ 0.
200A			07	Lower-order of 2007H.
200B			20	Higher-order of 2007H.
200C	INX	H	23	Increment H-L pair.
200D	MOV	M, A	77	Move the result from accumulator to memory.
200E	HLT		76	Halt.

Explanation

- This program finds the square of an 8-bit number stored in memory location 3000H.
- The square of a number is found by multiplying it by itself.
- Therefore, the number is added with itself and is also used as counter.
- Let us assume that the operands stored at memory location 3000H is 03H.
- Then, by using successive addition method, we get 03H + 03H + 03H = 09H.
- Initially, H-L pair is loaded with the address of the operand.
- The operand is moved to register B from memory location 3000H and then it is copied to register C.
- Accumulator is initialized to 00H.
- Register B is added with accumulator and the result is stored in accumulator.
- Register C (counter) is decremented by 1.
- Then, counter is checked for zero. If it hasn't become zero yet, then register B is again added with accumulator, and counter is again checked for zero.
- If counter becomes zero, then H-L pair is incremented and the result is moved

from accumulator to memory location 3001H.

Output

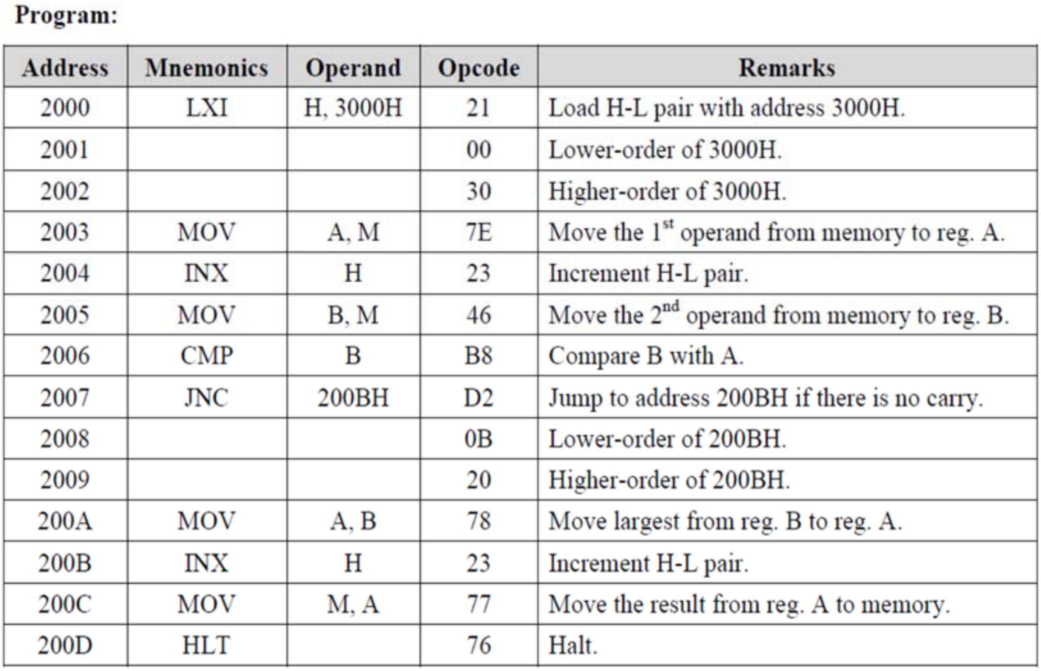
Before Execution

3000H: 03H

After Execution

3001H: 09H

Program 5: WAP to find larger of two 8-bit number



Program:

Address	Mnemonics	Operand	Opcode	Remarks
2000	LXI	H, 3000H	21	Load H-L pair with address 3000H.
2001			00	Lower-order of 3000H.
2002			30	Higher-order of 3000H.
2003	MOV	A, M	7E	Move the 1st operand from memory to reg. A.
2004	INX	H	23	Increment H-L pair.
2005	MOV	B, M	46	Move the 2nd operand from memory to reg. B.
2006	CMP	B	B8	Compare B with A.
2007	JNC	200BH	D2	Jump to address 200BH if there is no carry.
2008			0B	Lower-order of 200BH.
2009			20	Higher-order of 200BH.
200A	MOV	A, B	78	Move largest from reg. B to reg. A.
200B	INX	H	23	Increment H-L pair.
200C	MOV	M, A	77	Move the result from reg. A to memory.
200D	HLT		76	Halt.

Explanation

- This program compares the two operands to find the largest out of them.
- After comparison, the largest of two must be in accumulator. If it is already in accumulator, then it is moved to memory.
- If it is not in accumulator, then first it is moved to accumulator and then from there, it is moved to memory.
- Let us assume that the operands stored at memory location 3000H is 25H and 3001H is 15H.

- Initially, H-L pair is loaded with the address of first memory location.
- The first operand is moved to accumulator from memory location 3000H and H-L pair is incremented to point to next memory location.
- The second operand is moved to register B from memory location 3001H.
- The two operands are compared.
- After comparison, if A > B, then CF = 0, and if A < B, then CF = 1.
- Carry flag is checked for carry. If there is a carry, it means B is greater than A and it is moved to accumulator.
- At last, H-L pair is incremented and the largest number is moved from accumulator to memory location 3002H.

Output

Before Execution

3000H: 25H

3001H: 15H

After Execution

3002H: 25H

References

www.slideshare.net

http://weknownyourdreamz.com

http://www.8085projects.info

www.tutorialspoint.com

Microprocessor Architecture, programming and applications by Ramesh S.Gaonkar

Embedded System Design A unified hardware/software introduction by Frank Vahid/Tony Givargis