UNIT-I: Introduction to Microprocessors

1. Introduction and evolution of microprocessors
2. Architecture of 8085 processor
3. Pin configuration of 8085
4. Bus organization
5. Basic instruction sets.

1. INTRODUCTION TO MICROPROCESSOR AND MICROCOMPUTER ARCHITECTURE:

A microprocessor is a programmable electronics chip that has computing and decision making capabilities similar to central processing unit of a computer. Any microprocessor-based systems having limited number of resources are called microcomputers. Nowadays, microprocessor can be seen in almost all types of electronics devices like mobile phones, printers, washing machines etc.

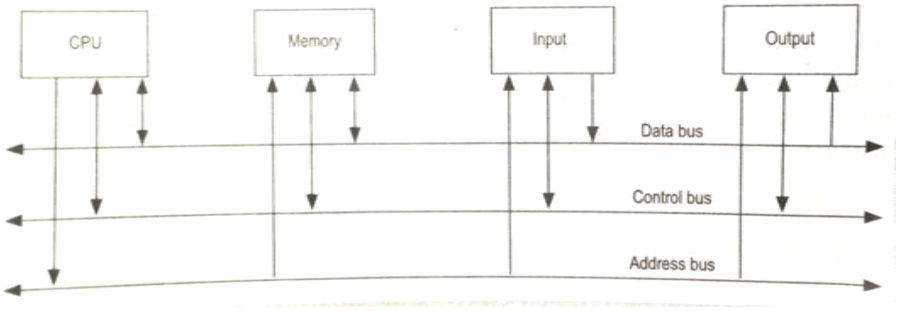

Fig.1 Microprocessor-based system

- **Address Bus**: It carries the address, which is a unique binary pattern used to identify a memory location or an I/O port. For example, an eight bit address bus has eight lines and thus it can address 2^8 = 256 different locations. The locations in hexadecimal format can be written as 00H – FFH.
- **Data Bus**: The data bus is used to transfer data between memory and processor or between I/O device and processor. For example, an 8-bit processor will generally have an 8-bit data bus and a 16-bit processor will have 16-bit data bus.
- **Control Bus**: The control bus carry control signals, which consists of signals for selection of memory or I/O device from the given address, direction of data transfer and synchronization of data transfer in case of slow devices.

A typical microprocessor consists of arithmetic and logic unit (ALU) in association with control unit to process the instruction execution. Almost all the microprocessors are based on the principle of store-program concept. In store-program concept, programs or instructions

are sequentially stored in the memory locations that are to be executed. To do any task using a microprocessor, it is to be programmed by the user. So the programmer must have idea about its internal resources, features and supported instructions. Each microprocessor has a set of instructions, a list which is provided by the microprocessor manufacturer. The instruction set of a microprocessor is provided in two forms: binary machine code and mnemonics.

Microprocessor communicates and operates in binary numbers 0 and 1. The set of instructions in the form of binary patterns is called a machine language and it is difficult for us to understand. Therefore, the binary patterns are given abbreviated names, called mnemonics, which forms the assembly language. The conversion of assembly-level language into binary machine-level language is done by using an application called assembler.

Evolution of Microprocessors

4-bit Microprocessors

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 and boolean 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 Microprocessors

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. Some other 8 bit processors are Zilog-80 and Motorola M6800.

16-bit Microprocessors

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

32-bit Microprocessors

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

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 multiple CPUs in a single system in order to achieve multiprocessing. The MMX extension was added to Pentium Pro and the result was Pentiuum II.

The Pentium III provided high performance floating point operations for certain types of computations by using the SIMD extensions to the instruction set. These new instructions makes the Pentium III faster than high-end RISC CPUs.

NAME	YEAR	TRANSISTORS	DATA WIDTH	CLOCK SPEED
8080	1974	6,000	8 bits	2 MHz
8085	1976	6,500	8 bits	5 MHz
8086	1978	29,000	16 bits	5 MHz
8088	1979	29,000	8 bits	5 MHz
80286	1982	134,000	16 bits	6 MHz
80386	1985	275,000	32 bits	16 MHz
80486	1989	1,200,000	32 bits	25 MHz
PENTIUM	1993	3,100,000	32/64 bits	60 MHz
PENTIUM II	1997	7,500,000	64 bits	233 MHz
PENTIUM III	1999	9,500,000	64 bits	450 MHz
PENTIUM IV	2000	42,000,000	64 bits	1.5 GHz

2. ARCHITECTURE OF 8085 MICROPROCESSOR

The 8085 microprocessor is an 8-bit processor available as a 40-pin IC package and uses +5 V for power. It can run at a maximum frequency of 3 MHz. Its data bus width is 8-bit and address bus width is 16-bit, thus it can address 2^{16} = 64 KB of memory. The internal architecture of 8085 is shown is Fig. 2.

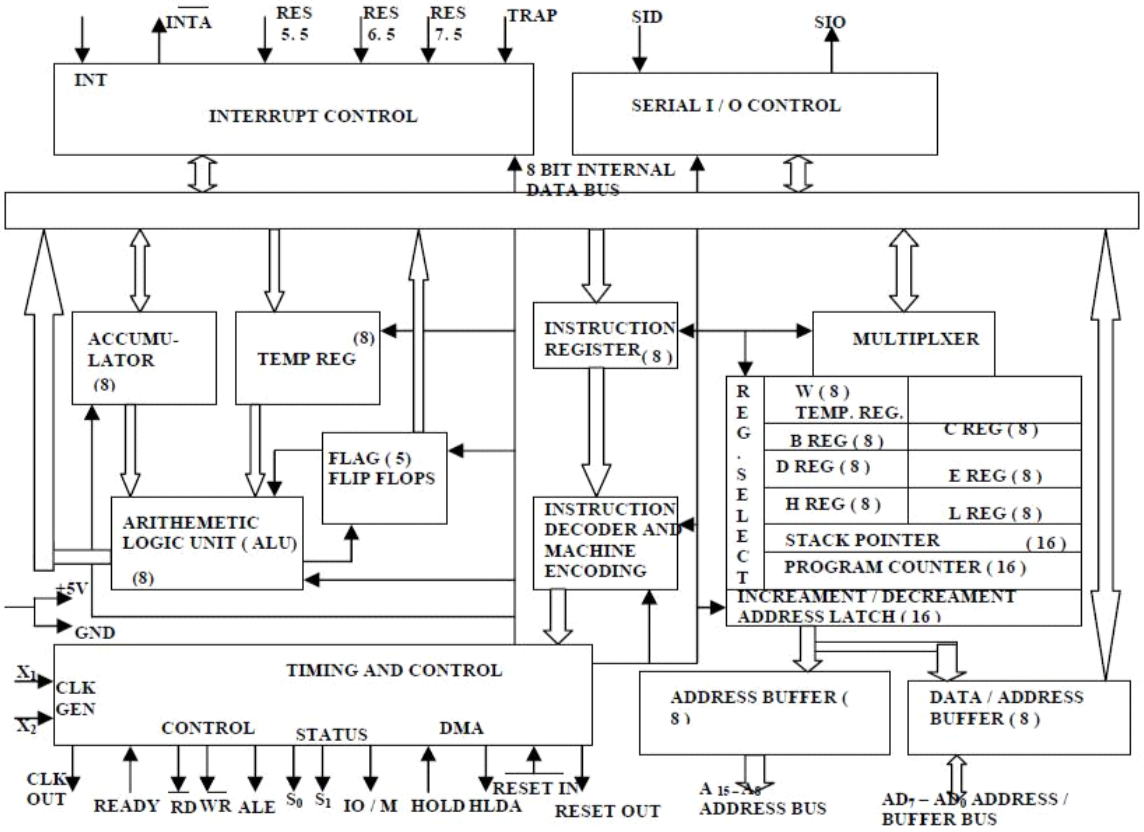

Fig. 2 Internal Architecture of 8085

Arithmetic and Logic Unit

The ALU performs the arithmetic and logical operations such as Addition (ADD), Subtraction (SUB), AND, OR etc. It uses data from memory and from Accumulator to perform operations. The results of the arithmetic and logical operations are stored in the accumulator.

Registers

The 8085 includes six registers, one accumulator and one flag register, as shown in Fig. 3. In addition, it has two 16-bit registers: stack pointer and program counter. They are briefly described as follows.

The 8085 has six general-purpose registers to store 8-bit data; these are identified as B, C, D, E, H and L. they can be combined as register pairs - BC, DE and HL to perform some 16-bit operations. The programmer can use these registers to store or copy data into the register by using data copy instructions.

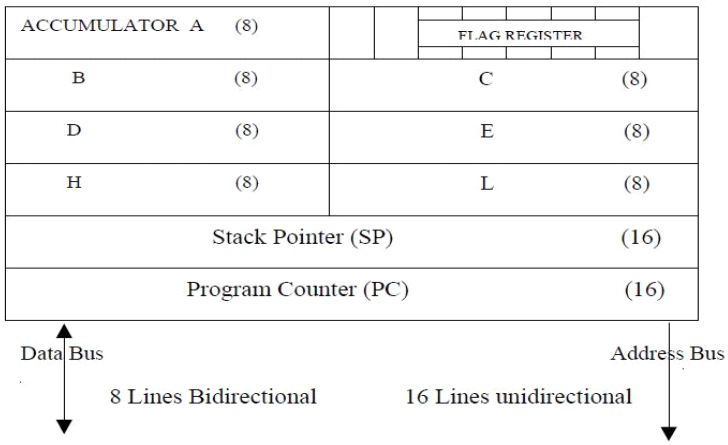

Fig. 3 Register organisation

Accumulator

The accumulator is an 8-bit register that is a part of ALU. This register is used to store 8-bit data and to perform arithmetic and logical operations. The result of an operation is stored in the accumulator. The accumulator is also identified as register A.

Flag register

The ALU includes five flip-flops, which are set or reset after an operation according to data condition of the result in the accumulator and other registers. They are called Zero (Z), Carry (CY), Sign (S), Parity (P) and Auxiliary Carry (AC) flags. Their bit positions in the flag register are shown in Fig. 4. The microprocessor uses these flags to test data conditions.

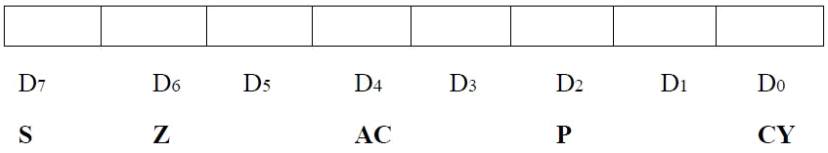

Fig. 4 Flag register

Program Counter (PC)

This 16-bit register deals with sequencing the execution of instructions. This register is a memory pointer. The microprocessor uses this register to sequence the execution of the instructions. The function of the program counter is to point to the memory address from which the next byte is to be fetched. When a byte is being fetched, the program counter is automatically incremented by one to point to the next memory location.

Stack Pointer (SP)

The stack pointer is also a 16-bit register, used as a memory pointer. It points to a memory location in R/W memory, called stack. The beginning of the stack is defined by loading 16-bit address in the stack pointer.

Instruction Register/Decoder

It is an 8-bit register that temporarily stores the current instruction of a program. Latest instruction sent here from memory prior to execution. Decoder then takes instruction and decodes or interprets the instruction. Decoded instruction then passed to next stage.

Control Unit

Generates signals on data bus, address bus and control bus within microprocessor to carry out the instruction, which has been decoded. Typical buses and their timing are described as follows:

- **Data Bus:** Data bus carries data in binary form between microprocessor and other external units such as memory. It is used to transmit data i.e. information, results of arithmetic etc between memory and the microprocessor. Data bus is bidirectional in nature. The data bus width of 8085 microprocessor is 8-bit.
- **Address Bus:** The address bus carries addresses and is one way bus from microprocessor to the memory or other devices. 8085 microprocessor contain 16-bit address bus and are generally identified as A0 - A15. The higher order address lines (A8 – A15) are unidirectional and the lower order lines (A0 – A7) are multiplexed (time-shared) with the eight data bits (D0 – D7) and hence, they are bidirectional.

- **Control Bus:** Control bus are various lines which have specific functions for coordinating and controlling microprocessor operations. The control bus carries control signals partly unidirectional and partly bidirectional. The following control and status signals are used by 8085 processor:

    - ALE (output): Address Latch Enable is a pulse that is provided when an address appears on the AD0 – AD7 lines, after which it becomes 0.
    - RD (active low output): The Read signal indicates that data are being read from the selected I/O or memory device and that they are available on the data bus.
    - WR (active low output): The Write signal indicates that data on the data bus are to be written into a selected memory or I/O location.
    - IO/M (output): It is a signal that distinguished between a memory operation and an I/O operation. When IO/M = 0 it is a memory operation and IO/M = 1 it is an I/O operation.
    - S1 and S0 (output): These are status signals used to specify the type of operation being performed; they are listed in Table 1.

Table 1 Status signals and associated operations

S1	S0	States
0	0	Halt
0	1	Write
1	0	Read
1	1	Fetch

3. Bus organization

The schematic representation of the 8085 bus structure is as shown in Fig. 5. The microprocessor performs primarily four operations:

- Memory Read: Reads data (or instruction) from memory.
- Memory Write: Writes data (or instruction) into memory.
- I/O Read: Accepts data from input device.
- I/O Write: Sends data to output device.

The 8085 processor performs these functions using address bus, data bus and control bus as shown in Fig. 5.

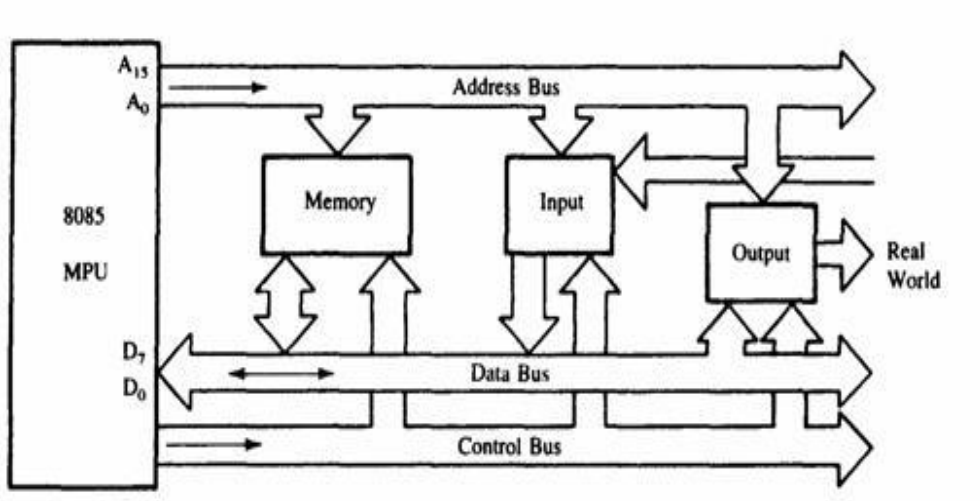

Fig. 5 The 8085 bus structure

4. 8085 PIN DESCRIPTION

Features:

- It is an 8-bit microprocessor
- Manufactured with N-MOS technology
- 40 pin IC package
- It has 16-bit address bus and thus has 2^{16} = 64 KB addressing capability.
- Operate with 3 MHz single-phase clock
- +5 V single power supply

The logic pin layout and signal groups of the 8085nmicroprocessor are shown in Fig. 6. All the signals are classified into six groups:

- Address bus
- Data bus
- Control & status signals
- Power supply and frequency signals
- Externally initiated signals
- Serial I/O signals

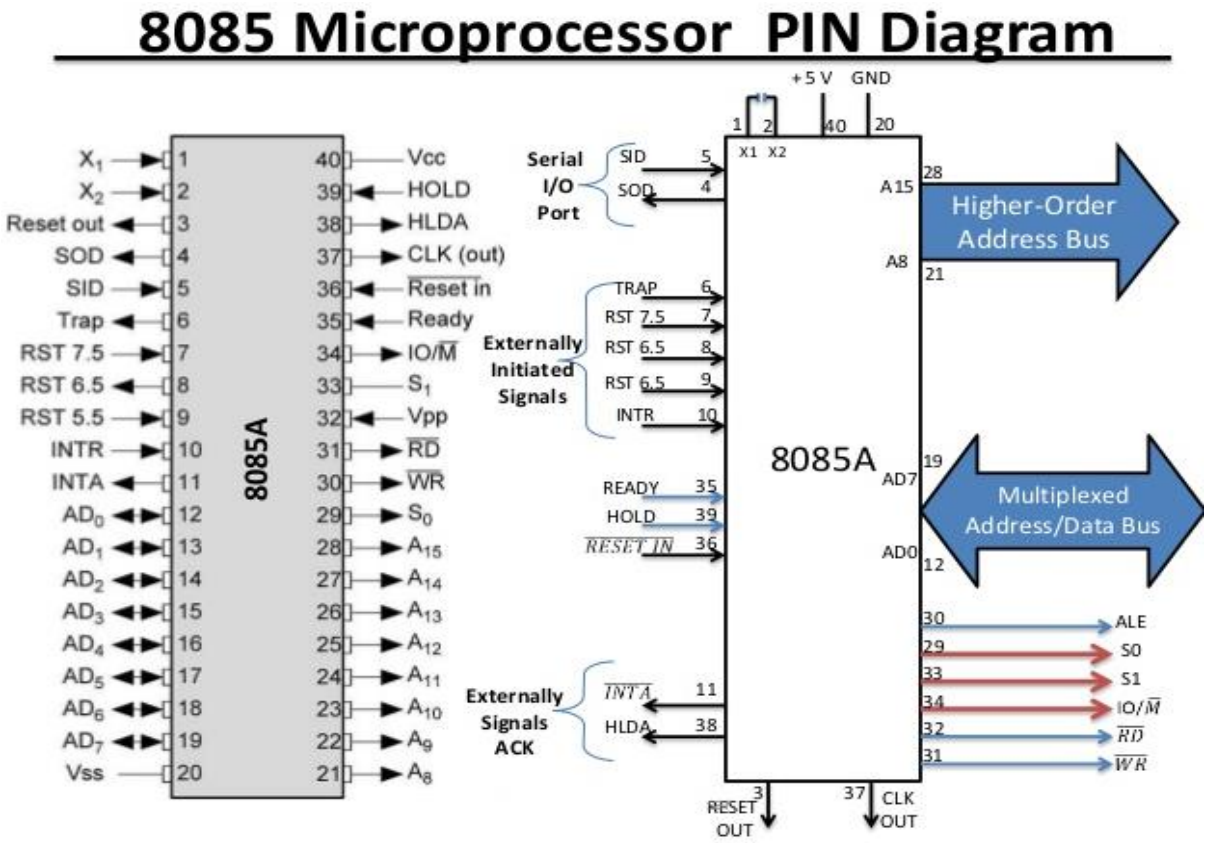

Fig. 6 8085 microprocessor pin layout and signal groups

Address and Data Buses:

- **A8 – A15 (output):** Most significant eight bits of memory addresses and the eight bits of the I/O addresses.

- **AD0 – AD7 (input/output):** Lower significant bits of memory addresses and the eight bits of the I/O addresses during first clock cycle. Behaves as data bus during third and fourth clock cycle.

Control & Status Signals:

- **ALE:** Address latch enable
- **RD** : Read control signal.
- **WR :** Write control signal.
- **IO/M, S1 and S0:** Status signals. Power Supply & Clock Frequency:
- **Vcc:** +5 V power supply
- **Vss:** Ground reference
- **X1, X2:** A crystal having frequency of 6 MHz is connected at these two pins
- **CLK:** Clock output

Externally Initiated and Interrupt Signals:

- **RESET IN**: When the signal on this pin is low, the PC is set to 0 and the processor is reset.
- **RESET OUT:** This signal indicates that the processor is being reset. The signal can be used to reset other devices.
- **READY:** When this signal is low, the processor waits for an integral number of clock cycles until it goes high.
- **HOLD:** This signal indicates that a peripheral like DMA (direct memory access) controller is requesting the use of address and data bus.
- **HLDA:** This signal acknowledges the HOLD request.
- **INTR:** Interrupt request is a general-purpose interrupt.
- **INTA:** This is used to acknowledge an interrupt.
- **RST 7.5, RST 6.5, RST 5,5** – restart interrupt: These are vectored interrupts and have highest priority than INTR interrupt.
- **TRAP:** This is a non-maskable interrupt and has the highest priority.

Serial I/O Signals:

- **SID:** Serial input signal. Bit on this line is loaded to D7 bit of register A using RIM instruction.
- **SOD:** Serial output signal. Output SOD is set or reset by using SIM instruction.

5. Basic instruction set

- **Data transfer operations:** This group of instructions copies data from source to destination. The content of the source is not altered.

- **Arithmetic operations:** Instructions of this group perform operations like addition, subtraction, increment & decrement. One of the data used in arithmetic operation is stored in accumulator and the result is also stored in accumulator.

- **Logical operations:** Logical operations include AND, OR, EXOR, NOT. The operations like AND, OR and EXOR uses two operands, one is stored in accumulator and other can be any register or memory location. The result is stored in accumulator. NOT operation requires single operand, which is stored in accumulator.

- **Branching operations:** Instructions in this group can be used to transfer program sequence from one memory location to another either conditionally or unconditionally.

- **Stack, I/O and Machine control instructions:** Instruction in this group control execution of other instructions and control operations like interrupt, halt etc.

Data transfer instructions:

Instructions, which are used to transfer data from one register to another register, from memory to register or register to memory, come under this group.

EXAMPLES:

1. MOV r1, r2 (Move Data; Move the content of the one register to another). [r1] <-- [r2]

2. MOV r, m (Move the content of memory register). r <-- [M]

3. MOV M, r. (Move the content of register to memory). M <-- [r]

4. MVI r, data. (Move immediate data to register). [r] <-- data.

5. MVI M, data. (Move immediate data to memory). M <-- data.

6. LXI rp, data 16. (Load register pair immediate). [rp] <-- data 16 bits, [rh] <-- 8 LSBs of data.

7. LDA addr. (Load Accumulator direct). [A] <-- [addr].

8. STA addr. (Store accumulator direct). [addr] <-- [A].

9. LHLD addr. (Load H-L pair direct). [L] <-- [addr], [H] <-- [addr+1].

10.SHLD addr. (Store H-L pair direct) [addr] <-- [L], [addr+1] <-- [H].

11.LDAX rp. (LOAD accumulator indirect) [A] <-- [[rp]]

12.STAX rp. (Store accumulator indirect) [[rp]] <-- [A].

13.XCHG. (Exchange the contents of H-L with D-E pair) [H-L] <--> [D-E].

2) Arithmetic instructions:

The instructions of this group perform arithmetic operations such as addition, subtraction; increment or decrement of the content of a register or memory.

Examples:

1. ADD r. (Add register to accumulator) [A] <-- [A] + [r].

2 .ADD M. (Add memory to accumulator) [A] <-- [A] + [[H-L]].

3.ADC r. (Add register with carry to accumulator). [A] <-- [A] + [r] + [CS].

4. ADC M. (Add memory with carry to accumulator) [A] <-- [A] + [[H-L]] [CS].

5 .ADI data (Add immediate data to accumulator) [A] <-- [A] + data.

6 .ACI data (Add with carry immediate data to accumulator). [A] <-- [A] + data + [CS].

7.DAD rp. (Add register pair to H-L pair). [H-L] <-- [H-L] + [rp].

8.SUB r. (Subtract register from accumulator). [A] <-- [A] – [r].

9.SUB M. (Subtract memory from accumulator). [A] <-- [A] – [[H-L]].

10.SBB r. (Subtract register from accumulator with borrow). [A] <-- [A] – [r] – [CS].

11.SBB M. (Subtract memory from accumulator with borrow). [A] <-- [A] – [[H-L]] – [CS].

12.SUI data. (Subtract immediate data from accumulator) [A] <-- [A] – data.

13.SBI data. (Subtract immediate data from accumulator with borrow). [A] <-- [A] – data – [CS].

14.INR r (Increment register content) [r] <-- [r] +1.

15. INR M. (Increment memory content) [[H-L]] <-- [[H-L]] + 1.

16.DCR r. (Decrement register content). [r] <-- [r] – 1.

17.DCR M. (Decrement memory content) [[H-L]] <-- [[H-L]] – 1.

18.INX rp. (Increment register pair) [rp] <-- [rp] – 1.

19.DCX rp (Decrement register pair) [rp] <-- [rp] -1.

20.DAA (Decimal adjust accumulator).

3) Logical instructions:

The Instructions under this group perform logical operation such as AND, OR, compare, rotate etc.

Examples:

1. ANA r. (AND register with accumulator) [A] <-- [A] ^ [r].

2.ANA M. (AND memory with accumulator). [A] <-- [A] ^ [[H-L]].

3.ANI data. (AND immediate data with accumulator) [A] <-- [A] ^ data.

4).ORA r. (OR register with accumulator) [A] <-- [A] ∨[r].

5.ORA M. (OR memory with accumulator) [A] <-- [A] ∨ [[H-L]]

6.ORI data. (OR immediate data with accumulator) [A] <-- [A] ∨ data.

7.XRA r. (EXCLUSIVE – OR register with accumulator) [A] <-- [A] xor [r]

8.XRA M. (EXCLUSIVE-OR memory with accumulator) [A] <-- [A] xor [[H-L]]

9.XRI data. (EXCLUSIVE-OR immediate data with accumulator) [A] <-- [A] xor data.

10.CMA. (Complement the accumulator) [A] <-- [A]'

11.CMC. (Complement the carry status) [CS] <-- [CS]'

12.STC. (Set carry status) [CS] <-- 1.

13.CMP r. (Compare register with accumulator) [A] – [r]

14.CMP M. (Compare memory with accumulator) [A] – [[H-L]]

15. CPI data. (Compare immediate data with accumulator) [A] – data.

16.RAL (Rotate accumulator left) [An+1] <-- [An], [A0] <-- [A7],[CS] <-- [A7].
The content of the accumulator is rotated left by one bit. The seventh bit of the accumulator is moved to carry bit as well as to the zero bit of the accumulator. Only CS flag is affected.

17.RAR. (Rotate accumulator right) [A7] <-- [A0], [CS] <-- [A0], [An] <-- [An+1].
The content of the accumulator is rotated right by one bit. The zero bit of the accumulator is moved to the seventh bit as well as to carry bit. Only CS flag is affected.

18. RLC. (Rotate accumulator left through carry) [An+1] <-- [An], [CS] <-- [A7], [A0] <-- [CS].

19.RRC. (Rotate accumulator right through carry) [An] <-- [An+1], [CS] <-- [A0], [A7] <-- [CS].

4) Branching Instructions:

This group includes the instructions for conditional and unconditional jump, subroutine call and return, and restart.

Examples:
Unconditional jump:
MP addr (label). (Unconditional jump: jump to the instruction specified by the address). [PC] <-- Label.
Conditional Jump:
Conditional Jump addr (label): After the execution of the conditional jump instruction the program jumps to the instruction specified by the address (label) if the specified condition is true. The program proceeds further in the normal sequence if the specified condition is not true.
1.JZ addr (label). (Jump if ZF=1)
2.JNZ addr (label) (Jump if ZF=0)
3.JC addr (label). (Jump if CF=1)
4.JNC addr (label). (Jump if CF=0)
5.JP addr (label). (Jump if the result is plus)
6.JM addr (label). (Jump if the result is minus)
7.JPE addr (label) (Jump if even parity)
8.JPO addr (label) (Jump if odd parity)
9.CALL addr (label) (Unconditional CALL: call the subroutine identified by the operand)
10.CALL instruction is used to call a subroutine.
11.RET (Return from subroutine).
12.RST n (Restart) Restart is a one-word CALL instruction. The content of the program counter is saved in the stack. The program jumps to the instruction starting at restart location.

5). Stack, I/O and Machine control instructions:

1.IN port-address. (Input to accumulator from I/O port) [A] <-- [Port]
2.OUT port-address (Output from accumulator to I/O port) [Port] <-- [A]
3.PUSH rp (Push the content of register pair to stack)
4.PUSH PSW (PUSH Processor Status Word)
5.POP rp (Pop the content of register pair, which was saved, from the stack)
6.POP PSW (Pop Processor Status Word)
7.HLT (Halt)
8.XTHL (Exchange stack-top with H-L)
9.SPHL (Move the contents of H-L pair to stack pointer)
10.EI (Enable Interrupts)
11.DI (Disable Interrupts)

12.SIM (Set Interrupt Masks)
13.RIM (Read Interrupt Masks)
14.NOP (No Operation).