Date :-

॥ नमामि देवि सरस्वति ॥

PRACTICAL - 1

*→ INSTRUCTIONS :

*→ For Data Transfer Instructions :

(1) MVI : Move Immediate, Loads an immediate value into register.
Ex, MVI C, [Value] , (C → register)

(2) MOV : Move, Copies the content of register C into the accumulator
Ex, MOV A, C (A, C → register)

(3) STA : Store Accumulator Direct, Stores the content of the accumulato
(A) into a specified memory address.
Ex, STA [Address] , STA 2002H.

(4) LXI : Load Register Pair Immediate, Loads a 16-bit address into the register pair.
Ex, LXI D, (Address) , LXI D, 2006H , Here it loads the immediate 16-bit address 2006H into the DE reg. pair.

(5) LDAX : Load An Accumulator Indirect - loads the byte from the memory location whose address is stored in the reg. pair into the accumulator (A).
Ex, LDAX D.

(6) STAX : Store Accumulator Indirect, Stores the content of the accumulator (A) into the memory location whose address is stored in the ~~DE~~ register pair.
Ex, STAX D

(7) LDA : Load Accumulator Direct, loads the byte from a specified memory address into the accumulator (A).

(8) PUSH : Push the contents of a register pair onto the stack.

(9) POP : Pop the contents from the stack into a register pair.

(10) IN : Read data from an input port into the accumulator.
Ex, IN 01H , Read data from input port 01H into the accumulator. 01H is the 8-bit address of the input port. It specifies which i/p device the microprocessor should read data from.

(11) OUT : Write data from the accumulator to an output port.
Ex, OUT 02H , Send the data from the accumulator to o/p port 02H.

(12) XCHG : Exchange the contents of the HL register pair with the DE register pair.

*→ Arithmetic Instructions :

(1) ADD : Add Register, Adds the content of register to the accumulator.
Ex, ADD B , adds the content of reg. B into the accumulator (A).

(2) ADI : Add immediate , Adds an immediate value to the content of the accumulator (A).
Ex, ADI [value] , ADI 05H , It adds the 05H value into the accumulator.

(3) ADC : Add register with carry, Adds the content of register to the accumulator (A) along with the carry flag.
Ex, ADC B, It adds content of reg. (B) to the accumulator along with carry flag.

(4) ACI : Add immediate with carry, Adds an immediate value to the accumulator (A) along with the carry flag.
Ex, ACI 03H, adds the immediate value 03H to the accumulator (A) along with the carry flag.

(5) DCR : Decrement the content of register or memory location by one.

(6) INR : Increment the content of register or memory location by one.

(7) SUB : Subtract the contents of a register or memory location from the accumulator
Ex, SUB B, it subtracts the value in register B from the accumulator.

(8) SBB : Substract with borrow, It substracts the value in register B & the borrow flag (carry flag) from the accumulator.

(9) SUI : It substract immediate data from the accumulator.
Ex, SUI 03H, substract 3 from accumulator.

(10) SBI : Substract immediate with borrow.
Ex, SBI 03H.

(11) INX : Increment a register pair.

(12) DCX : Decrement a register pair.

(13) DAD : Double add, it add a register pair to HL.
Ex, LXI H, 2000H ; load HL with 2000H
LXI B, 1000H ; load BC with 1000H
DAD B ; Add BC to HL.

(14) DAA : Decimal adjust accumulator, It adjusts the result of an addition operation in the accumulator to produce a valid Binary Coded Decimal (BCD) result.
Ex, MVI A, 48H ; load 48 (BCD) into A
ADI 29 H ; Add 29 (BCD) to A.
DAA ; Decimal adjust the result.
Since 71H is not valid BCD Number, DAA corrects it to 77H, which is the correct BCD representation of (48 + 29).

*→ Logical Instructions :

(1) ANA : AND operation, Logical AND with the Accumulator.
Ex, ANA B, performs a bitwise AND operation between the accumulator (A) & the value in register B.

(2) ANI : Logical AND immediate with the Accumulator.
Ex, ANI 0F0H, performs a bitwise AND operation between the accumulator (A) & the immediate value 0F0H.

(3) ORA : logical OR with the accumulator.
Ex, ORA B, performs a bitwise OR operation between

Date :-

|| नमामि देवि सरस्वति ||

the accumulator (A) & value in register B.

(4) ORI : logical OR immediate with the accumulator.
Ex, ORI 0F0H, performs a bitwise OR operation between the accumulator (A) & the immediate value 0F0H.

(5) XRA : logical Exclusive-OR with the accumulator.
Ex, XRA B, performs a bitwise XOR operation between the accumulator (A) & the value in reg. B.

(6) XRI : logical Exclusive-OR immediate with the accumulator.
Ex, XRI 0F0H, performs a bitwise XOR operation between the accumulator (A) & the immediate value 0F0H.

(7) CMP : Compare the contents of register or memory location with the accumulator.
Ex, CMP B, Compares the value in register B with the accumulator (A) & sets the flags accordingly (Carry & zero flags in this case).

(8) CPI : Compare immediate data with the accumulator.
ex, CPI 0FFH, Compare the immediate value 0FFH with the accumulator (A) & sets the flags (zero flag in this case).

(9) CMA : Complement the accumulator.

(10) CMC : Complement the carry flag.

(11) STC : Set the Carry flag.

*→ Shift & Rotate Instructions :

(1) RLC : Rotate accumulator left (carry included)

(2) RRC : Rotate accumulator right (carry included), Rotates the bits in the accumulator one position to the right, with the rightmost bit moving into both the carry flag & the leftmost bit position.

(3) RAL : Rotate accumulator left through carry.
Rotates the bits in the accumulator one position to the left, with the carry flag moving into the rightmost bit and the leftmost bit moving into the carry flag.

(4) RAR : Rotate accumulator right through carry.
Rotates the bits in the accumulator one position to the right, with the carry flag moving into the leftmost bit & the rightmost bit moving into the carry flag.

(5) SLA : Shift left arithmetic.
It shifts all the bits in the accumulator one position to the left.
Each bit is moved one position to the left.
The least significant bit (LSB) is filled with a 0.

(6) SRA : Shift right arithmetic.
It shifts all the bits in the accumulator one position to the right.
The most significant bit remains unchanged (It's copied into the bit to its right). This is the key difference between SRA & SRL.

Date :-

|| नमामि देवि सरस्वति ||

Ex, MVI A, 8AH ; load 1000 1010 (Binary) into A.
SRA ; shift right arithmetic.

The result in A becomes 1100 0101 (binary).

(7) SRL : Shift right logical.
It shifts all the bits in the accumulator one position to the right.
Each bit is moved one position to the right
The most significant bit (MSB) is filled with a 0. This is key difference between SRL & SRA.

Ex, MVI, 8AH ; load 10001010 (binary) into A.
SRL ; shift right logical.

The result in A becomes 01000101 (Binary).

Date :-

|| नमामि देवि सरस्वति ||

PRACTICAL - 2

*→ INSTRUCTIONS :

(1) MOV : Move, copies the content of register into Accumulator.
Ex, mov A, C (A, C ← register).

Here, content of register C is copied into the accumulator.

(2) LDA : load Accumulator Direct → loads the byte from a specified memory address into the accumulator (A).
Ex, LDA 3000H.

Here, Accumulator content the value which is present in the memory location 3000H.

(3) ADD : Add register, Adds the content of register to the accumulator.
Ex, ADD B, adds the content of reg. B into the accumulator (A).

(4) STA : Store Accumulator Direct → Stores the content of the accumulator (A) into a specified memory address.
Ex, STA 2002H.

Here, The content of accumulator stores into the specified memory location.

(5) LXI : load register pair Immediate → loads a 16-bit address into the register pair.
Ex, LXI D, [Address] → LXI D, 2006H

Here it loads the immediate 16-bit address 2006H into the DE register pair.

(6) INX : Increment a register pair, the value will be incremented by one.

(7) HLT : It is used to usefull for stop the 8085 microprocessor from executing any further instructions.

Date :-

|| नमामि देवि सरस्वति ||

PRACTICAL - 3

*→ INSTRUCTIONS :

(1) ADD : Add register, Adds the content of register to the accumulator.
Ex, ADD B, adds the content of reg. B into the accumulator (A).

(2) SUB : Substruct the contents of a register or memory location from the accumulator.
Ex, SUB B. → it substructs the value in register B from the accumulator.

(3) ADC : Add register with carry, Adds the content of register to the accumulator (A) along with the carry flag.
Ex, ADC B, It adds content of reg. (B) to the accumulator along with carry flag.

(4) SBB : Substruct with borrow, It substructs the value in register & the borrow flag (carry flag → c) from the accumulator.

(5) CMA : Compliment, It is used for compliment the value of Accumulator.

By using ~~this~~ this ~~cts & a~~ ...
This instruction is used for finding 1's or 2's compliment.

(6) INR : Increment , It is usefull for increment the value by 1 in register.
ex, INR A

Here, the value which is present in Accumulator, will incremented by 1.

(7) MVI : Move Immediate, loads an immediate value into register.
Ex, MVI C, (value) → MVI C, 05H.

Here, the value 05H will be stored in register C.

Date :-

|| नमामि देवि सरस्वति ||

PRACTICAL - 4

*→ INSTRUCTIONS :

(1) DAA : Decimal adjust accumulator, It adjusts the result of an addition operation in the accumulator to produce a valid Binary coded Decimal (BCD) result.

```
Ex, MVI A, 48H      ; load 48 (BCD) into A.
    ADI 29H         ; Add 29 (BCD) to A.
    DAA             ; Decimal adjust the result.
```

→ since 71H is not valid BCD Number, DAA corrects it to 77H, which is the correct BCD representation of (48 + 29).

(2) LHLD : load HL Direct, It loads the contents of a 16-bit memory location directly into the HL register pair.

→ It retrieves two consecutive bytes from memory & places them into the H (High-order) & L (low-order) registers.

→ Ex, LHLD 2050H ; This instruction would load the byte at memory location 2050H into the L register & the byte at memory location 2051H into the H register.

(3) XCHG : Exchange HL with DE, It exchanges the content of the HL register pair with the contents of the DE register pair.

Date :-
|| नमामि देवि सरस्वति ||

→ Specifically, the content of the H register is exchanged with the content of the D register, & the content of the L register is exchanged with the content of the E register.

→ If the HL register pair contains 1000H & the DE reg. pair contains 2000H, after executing XCHG instruction, HL will contain 2000H & DE will contain 1000H.

(4) DAD : Double Add, it add a register pair to HL

```
Ex, LXI H, 2000H     ; load HL with 2000H
    LXI B, 1000H     ; load BC with 1000H
    DAD B            ; Add BC to HL.
```

(5) SHLD : Store HL Direct, it stores the contents of the HL register pair into two consecutive memory locations.

→ The content of the L register is stored in the memory location specified by the 16-bit address in the instruction.

→ The content of the H register is stored in the next ~~[illegible]~~ consecutive memory location.

→ Ex, SHLD 2010H ; This instruction would store the content of the L register into memory location 2010H & the content of the H register into memory location 2011H.

→ SHLD takes the 16-bit number that is held within the HL register pair, & places that number into two consecutive memory locations.

(6) JNZ : Jump if Not zero, this instruction causes the program to jump to a specified 16-bit memory address if the zero flag (Z) in the 8085's flag register is 0 (meaning the result of the previous operation was not zero).

→ If the zero flag is 1 (meaning the result was zero), the program continues to the next sequential instruction.

(7) ~~loop kab~~ ANA : AND Accumulator, This instruction performs a logical AND operation between the contents of the accumulator (A) & the contents of a specified register or memory location.

Ex, ANA B, performs a bitwise AND operation between the accumulator (A) & the value in register B.

(8) ORA : OR Accumulator, This instruction performs a bitwise logical OR operation between the contents of the accumulator (A) & the contents of a specified register or memory location.

Ex, ORA B, performs a bitwise OR operation between the accumulator (A) & value in register B.