AIM: - To study & Perform a Practical on Data Transfer Instructions.

THEORY:

An instruction is a binary pattern designed inside a microprocessor to perform a specific function. The entire group of instructions that a microprocessor supports is called ***Instruction Set***.

8085 has **246** instructions. Each instruction is represented by an 8-bit binary value. These 8-bits of binary value are called ***Op-Code*** or ***Instruction Byte.***

Classification of Instruction Set

- Data Transfer Instruction
- Arithmetic Instructions
- Logical Instructions
- Branching Instructions
- Control Instructions

Data Transfer Instruction:-

These instructions move data between registers, or between memory and registers. These instructions copy data from source to destination. While copying, the contents of source are not modified.

Opcode	Operand	Description
MOV	**Rd, Rs** **M, Rs** **Rd, M**	**Copy from source to destination.**

This instruction copies the contents of the source register into the destination register. The contents of the source register are not altered. If one of the operands is a memory location, its location is specified by the contents of the HL registers.

Example: MOV B, C or MOV B, M

Opcode	Operand	Description
MVI	**Rd, Data** **M, Data**	**Move immediate 8-bit**

The 8-bit data is stored in the destination register or memory. If the operand is a memory location, its location is specified by the contents of the H-L registers.
Example: MVI B, 57H or MVI M, 57H

Opcode	Operand	Description
LDA	**16-bit address**	**Load Accumulator**

The contents of a memory location, specified by a 16-bit address in the operand, are copied to the Accumulator. The contents of the source are not altered.
Example: LDA 2034H

Opcode	Operand	Description
LDAX	**B/D Register Pair**	**Load accumulator indirect**

The contents of the designated register pair point to a memory location. This instruction copies the contents of that memory location into the accumulator. The contents of either the register pair or the memory location are not altered.
Example: LDAX B

Opcode	Operand	Descr
LXI	**Reg. pair, 16-bit data**	**Load register pair immediate**

This instruction loads 16-bit data in the register pair.
Example: LXI H, 2034 H

Opcode	Operand	Description
LHLD	**16-bit address**	**Load H-L registers direct**

This instruction copies the contents of memory location pointed out by 16-bit address into register L. It copies the contents of next memory location into register H.
Example: LHLD 2040 H

Opcode	Operand	Description
STA	**16-bit address**	**Store accumulator direct**

The contents of accumulator are copied into the memory location specified by the operand.
Example: STA 2500 H

Opcode	Operand	Description
STAX	**Reg. pair**	**Store accumulator indirect**

The contents of accumulator are copied into the memory location specified by the contents of the register
Example: STAX B

Opcode	Operand	Description
SHLD	**16-bit address**	**Store H-L registers direct**

The contents of register L are stored into memory location specified by the 16-bit address. The contents of register H are stored into the next memory location.
Example: SHLD 2550 H

Opcode	Operand	Description
XCHG	**None**	**Exchange H-L with D-E**

The contents of register H are exchanged with the contents of register D. The contents of register L are exchanged with the contents of register E.
Example: XCHG

Opcode	Operand	Description
SPHL	**None**	**Copy H-L pair to the Stack Pointer (SP)**

This instruction loads the contents of H-L pair into SP.
Example: SPHL

Opcode	Operand	Description
XTHL	**None**	**Exchange H–L with top of stack**

The contents of L register are exchanged with the location pointed out by the contents of the SP. The contents of H register are exchanged with the next location (SP + 1).
Example: XTHL

Opcode	Operand	Description
PCHL	**None**	**Load program counter with H-L contents**

The contents of registers H and L are copied into the program counter (PC). The contents of H are placed as the high-order byte and the contents of L as the low-order byte.
Example: PCHL

Opcode	Operand	Description
PUSH	**Reg. pair**	**Push register pair onto stack**

The contents of register pair are copied onto stack. SP is decremented and the contents of high-order registers (B, D, H, A) are copied into stack. SP is again decremented and the contents of low-order registers (C, E, L, Flags) are copied into stack.
Example: PUSH B

Opcode	Operand	Description
POP	**Reg. pair**	**Pop stack to register pair**

The contents of top of stack are copied into register pair. The contents of location pointed out by SP are copied to the low-order register (C, E, L, Flags). SP is incremented and the contents of location are copied to the high-order register (B, D, H, A).
Example: POP H

Opcode	Operand	Description
OUT	**8-bit port address**	**Copy data from accumulator to a port with 8- bit address**

The contents of accumulator are copied into the I/O port.
Example: OUT 78 H

Opcode	Operand	Description
IN	**8-bit port address**	**Copy data to accumulator from a port with 8- bit address**

The contents of I/O port are copied into accumulator.
Example: IN 8C H

EXERCISE:

Write and execute following programs in microprocessor kit

1. Write an ALP to store 32H in accumulator and store this data to all internal register.
2. Write an ALP to store 12H in accumulator and store this data to memory location 2800H.
3. Write an ALP to transfer a byte from memory location 2200H to 2250H.
4. Transfer the byte from memory location 2200H to the location 2250H using indirect addressing mode.
5. Write an ALP to exchange a byte of data between memory location 2200H & 2250H using indirect and indirect addressing mode.

AIM: - To Study & Perform Practical on Arithmetic Operations in 8085.

THEORY:

Arithmetic Instructions

These instructions perform the operations like:

- Addition
- Subtract
- Increment
- Decrement

Addition

Any 8-bit number, or the contents of register, or the contents of memory location can be added to the contents of accumulator. The result (sum) is stored in the accumulator. No two other 8-bit registers can be added directly. The contents of register B cannot be added directly to the contents of register C.

Subtraction

Any 8-bit number, or the contents of register, or the contents of memory location can be subtracted from the contents of accumulator. The result is stored in the accumulator. Subtraction is performed in 2's complement form. If the result is negative, it is stored in 2's complement form. No two other 8-bit registers can be subtracted directly.

Increment / Decrement

The 8-bit contents of a register or a memory location can be incremented or decremented by 1.The 16-bit contents of a register pair can be incremented or decremented by 1. Increment or decrement can be performed on any register or a memory location.

Opcode	Operand	Description
ADD	R M	Add register or memory to accumulator

The contents of register or memory are added to the contents of accumulator. he result is stored in accumulator. If the operand is memory location, its address is specified by H-L pair. All flags are modified to reflect the result of the addition.
Example: ADD B or ADD M

Opcode	Operand	Description
ADC	R M	Add register or memory to accumulator with carry

The contents of register or memory and Carry Flag (CY) are added to the contents of accumulator. The result is stored in accumulator. If the operand is memory location, its address is specified by H-L pair. All f lags are modified to ref lect the result of the addition.
Example: ADC B or ADC M

Opcode	Operand	Description
ADI	8-bit data	Add immediate to accumulator

The 8-bit data is added to the contents of accumulator. The result is stored in accumulator. All Flags are modified to reflect the result of the addition.

Example: ADI 45 H

Opcode	Operand	Description
ACI	8-bit data	Add immediate to accumulator with carry

The 8-bit data and the Carry Flag (CY) are added to the contents of accumulator. The result is stored in accumulator. All f lags are modified to ref lect the result of the addition.

Example: ACI 45 H

Opcode	Operand	Description
DAD	Reg. pair	Add register pair to H-L pair

The 16-bit contents of the register pair are added to the contents of H-L pair. The result is stored in H-L pair. If the result is larger than 16 bits, then CY is set. No other f lags are changed.

Example: DAD B

Opcode	Operand	Description
SUB	R M	Subtract register or memory from accumulator

The contents of the register or memory location are subtracted from the contents of the accumulator. The result is stored in accumulator. If the operand is memory location, its address is specified by H-L pair. All f lags are modified to ref lect the result of subtraction.

Example: SUB B or SUB M

Opcode	Operand	Description
SBB	R	Subtract register or memory from accumulator with borrow

The contents of the register or memory location and Borrow Flag (i.e. CY) are subtracted from the contents of the accumulator. The result is stored in accumulator. If the operand is memory location, its address is specified by H-L pair. All flags are modified to reflect the result of subtraction.

Example: SBB B or SBB M

Opcode	Operand	Description
SUI	8-bit data	Subtract immediate from accumulator

The 8-bit data is subtracted from the contents of the accumulator .The result is stored in accumulator. All flags are modified to reflect the result of subtraction.
Example: SUI 45 H

Opcode	Operand	Description
SBI	8-bit data	Subtract immediate from accumulator with borrow

The 8-bit data and the Borrow Flag (i.e. CY) is subtracted from the contents of the accumulator. The result is stored in accumulator. All f lags are modified to ref lect the result of subtraction.
Example: SBI 45 H

Opcode	Operand	Description
INR	R M	Increment register or memory by 1

The contents of register or memory location are incremented by 1. The result is stored in the same place. If the operand is a memory location, its address is specified by the contents of H-L pair.
Example: INR B or INR M

Opcode	Operand	Description
INX	R	Increment register pair by 1

The contents of register pair are incremented by 1. The result is stored in the same place.
Example: INX H

Opcode	Operand	Description
DCR	R M	Decrement register or memory by 1

The contents of register or memory location are decremented by 1. The result is stored in the same place. If the operand is a memory location, its address is specified by the contents of H-L pair.
Example: DCR B or DCR M

Opcode	Operand	Description
DCX	R	Decrement register pair by 1

The contents of register pair are decremented by 1. The result is stored in the same place.
Example: DCX H

EXERCISE:

Write and execute following programs in microprocessor kit.

1. Load 56H in accumulator and B7H in reg. C. Add these two numbers and store answer at the memory location 2250H
2. Store two numbers in memory location 2200H and 2201H. Add these two numbers and store result on memory location 2202H (numbers are 59H and 77H).
3. Load the accumulator by 45H and get another number from memory location 2300H (F9H) and add the two numbers. Store the result to location 2350H. Check the carry flag.
4. Get two numbers from memory location 2200H and 2201H (47H and79H). Perform subtraction (47-79). Store the result to location 2202H. Check the carry flag.
5. Load a number 49H in reg. B and number 12H in reg. A. Perform subtraction and store result to memory location 2200H.

AIM: To Study & Perform Practical related to Logical instructions.

THEORY:

Logical Instructions

These instructions perform logical operations on data stored in registers, memory and status flags.

The logical operations are:

- AND
- OR
- XOR
- Rotate
- Compare
- Complement

AND, OR, XOR

Any 8-bit data or the contents of register, or memory location can logically have

- AND operation
- OR operation
- XOR

Operation with the contents of accumulator. The result is store in accumulator.

Rotate

E ach bit in the accumulator can be shifted either left or right to the next position.

Compare

Any 8-bit data or the contents of register, or memory location can be compares for:

- Equality
- Greater Than
- Less Than

With the contents of accumulator. The result is reflected in status flags.

Complement

The contents of accumulator can be complemented. Each 0 is replaced by 1 and each 1 is replaced by 0.

Opcode	**Operand**	**Description**
CMP	R M	Compare register or memory with accumulator

The contents of the operand (register or memory) are compared with the contents of the accumulator. Both contents are preserved .The result of the comparison is shown by setting the f lags of the PSW as follows:

Opcode	Operand	Description
CMP	R M	Compare register or memory with accumulator

- if (A) < (reg/mem): carry f lag is set
- if (A) = (reg/mem): zero f lag is set
- if (A) > (reg/mem): carry and zero f lags are reset.
- **Example:** CMP B or CMP M

Opcode	Operand	Description
CPI	8-bit data	Compare immediate with accumulator

The 8-bit data is compared with the contents of accumulator. The values being compared remain unchanged. The result of the comparison is shown by setting the f lags of the PSW as follows:

Opcode	Operand	Description
CPI	8-bit data	Compare immediate with accumulator

- if (A) < data: carry f lag is set
- if (A) = data: zero f lag is set
- if (A) > data: carry and zero f lags are reset
- Example: CPI 89H

Opcode	Operand	Description
ANA	R M	Logical AND register or memory with accumulator

The contents of the accumulator are logically ANDed with the contents of register or memory. The result is placed in the accumulator. If the operand is a memory location, its address is specified by the contents of H-L pair. S, Z, P are modified to ref lect the result of the operation. CY is reset and AC is set.
Example: ANA B or ANA M.

Opcode	Operand	Description
ANI	8-bit data	Logical AND immediate with accumulator

The contents of the accumulator are logically ANDed with the 8-bit data. The result is placed in the accumulator. S, Z, P are modified to reflect the result. CY is reset, AC is set.

Example: ANI 86H.

Opcode	Operand	Description
ORA	R M	Logical OR register or memory with accumulator

The contents of the accumulator are logically ORed with the contents of the register or memory. The result is placed in the accumulator. If the operand is a memory location, its address is specified by the contents of H-L pair. S, Z, P are modified to ref lect the result. CY and AC are reset.
Example: ORA B or ORA M.

Opcode	Operand	Description
ORI	8-bit data	Logical OR immediate with accumulator

The contents of the accumulator are logically ORed with the 8-bit data. The result is placed in the accumulator. S, Z, P are modified to ref lect the result. CY and AC are reset.

Example: ORI 86H.

Opcode	Operand	Description
XRA	R M	Logical XOR register or memory with accumulator

The contents of the accumulator are XORed with the contents of the register or memory. The result is placed in the accumulator. If the operand is a memory location, its address is specified by the contents of H-L pair. S, Z, P are modified to reflect the result of the operation. CY and AC are reset.

Example: XRA B or XRA M.

Opcode	Operand	Description
XRI	8-bit data	XOR immediate with accumulator

The contents of the accumulator are XORed with the 8-bit data. The result is placed in the accumulator. S, Z, P are modified to reflect the result. CY and AC are reset.

Example: XRI 86H.

Opcode	Operand	Description
RLC	None	Rotate accumulator left

Each binary bit of the accumulator is rotated left by one position. Bit D7 is placed in the position of D0 as well as in the Carry f lag. CY is modified according to bit D7. S, Z, P, AC are not affected.

Example: RLC.

Opcode	Operand	Description
RRC	None	Rotate accumulator right

Each binary bit of the accumulator is rotated right by one position. Bit D0 is placed in the position of D7 as well as in the Carry flag.CY is modified according to bit D0. S, Z, P, AC are not affected.

Example: RRC.

Opcode	Operand	Description
RAL	None	Rotate accumulator left through carry

Each binary bit of the accumulator is rotated left by one position through the Carry flag. Bit D7 is placed in the Carry flag, and the Carry f lag is placed in the least significant position D0. CY is modified according to bit D7. S, Z, P, AC are not affected.

Example: RAL

Opcode	Operand	Description
RAR	None	Rotate accumulator right through carry

Each binary bit of the accumulator is rotated right by one position through the Carry f lag. Bit D0 is placed in the Carry f lag, and the Carry f lag is placed in the most significant position D7.CY is modified according to bit D0. S, Z, P, AC are not affected.

Example: RAR.

Opcode	Operand	Description
CMA	None	Complement accumulator

The contents of the accumulator are complemented. No f lags are affected.

Example: CMA.

Opcode	Operand	Description
CMC	None	Complement carry

The Carry f lag is complemented. No other f lags are affected.

Example: CMC.

Opcode	Operand	Description
STC	None	Set carry

The Carry f lag is set to 1.No other f lags are affected.
Example: STC.

EXERCISE:

Write and execute following programs in microprocessor kit.

1. A memory location 2202H contains 80H data. Write an ALP to convert this data to 08H. Store the modified number to location 2221H.
2. Load 93H in register A and 6CH in register B and perform the EX-OR operation with the numbers and store result on memory location 2200H. (Note: use only AND, OR and compliment operation).
3. Memory locations 2200H contain 88H data. Mask the bit D2, D3, D6 and D7 of the byte and store result on memory location 2201H.
4. A memory location 2200H contains 70H data. Find out the compliment of this number
5. Write an ALP to set a number from memory location 2040H and convert the number to 2's compliment number and store result in memory location 2041H.
6. Write an ALP to perform X-OR operation with two bytes 6CH and 9FH. Store result in register B.

AIM: - To Study & Perform Practical on Branch and Control Instructions.

THEORY:

BRANCHING INSTRUCTION

The branching instruction alters the normal sequential flow. These instructions alter either unconditionally or conditionally.

Opcode	Operand	Description
JMP	16-bit address	Jump unconditionally

The program sequence is transferred to the memory location specified by the 16-bit address given in the operand.

Example: JMP 2034 H.

Opcode	Operand	Description
Jx	16-bit address	Jump conditionally

The program sequence is transferred to the memory location specified by the 16-bit address given in the operand based on the specified flag of the PSW.

Example: JZ 2034 H.

JUMP CONDITIONALLY

Opcode	Description	Status Flags
JC	Jump if Carry	CY = 1
JNC	Jump if No Carry	CY = 0
JP	Jump if Positive	S = 0
JM	Jump if Minus	S = 1
JZ	Jump if Zero	Z = 1
JNZ	Jump if No Zero	Z = 0
JPE	Jump if Parity Even	P = 1
JPO	Jump if Parity Odd	P = 0

Opcode	Operand	Description
CALL	16-bit address	Call unconditionally

The program sequence is transferred to the memory location specified by the 16-bit address given in the operand. Before the transfer, the address of the next instruction after CALL (the contents of the program counter) is pushed onto the stack.

Example: CALL 2034 H.

Opcode	Operand	Description
Cx	16-bit address	Call conditionally

The program sequence is transferred to the memory location specified by the 16-bit address given in the operand based on the specified f lag of the PSW. Before the transfer, the address of the next instruction after the call (the contents of the program counter) is pushed onto the stack.

Example: CZ 2034 H.

CALL CONDITIONALLY

Opcode	Description	Status Flags
CC	Call if Carry	CY = 1
CNC	Call if No Carry	CY = 0
CP	Call if Positive	S = 0
CM	Call if Minus	S = 1
CZ	Call if Zero	Z = 1
CNZ	Call if No Zero	Z = 0
CPE	Call if Parity Even	P = 1
CPO	Call if Parity Odd	P = 0

Opcode	Operand	Description
RET	None	Return unconditionally

The program sequence is transferred from the subroutine to the calling program. The two bytes from the top of the stack are copied into the program counter, and program execution begins at the new address.

Example: RET.

Opcode	Operand	Description
Rx	None	Call conditionally

The program sequence is transferred from the subroutine to the calling program based on the specified flag of the PSW. The two bytes from the top of the stack are copied into the program counter, and program execution begins at the new address.

Example: RZ.

RETURN CONDITIONALLY

Opcode	Description	Status Flags
RC	Return if Carry	CY = 1
RNC	Return if No Carry	CY = 0
RP	Return if Positive	S = 0
RM	Return if Minus	S = 1
RZ	Return if Zero	Z = 1
RNZ	Return if No Zero	Z = 0
RPE	Return if Parity Even	P = 1
RPO	Return if Parity Odd	P = 0

Opcode	Operand	Description
RST	0 – 7	Restart (Software Interrupts)

The RST instruction jumps the control to one of eight memory locations depending upon the number. These are used as software instructions in a program to transfer program execution to one of the eight locations.

Example: RST 3.

RESTART ADDRESS TABLE

Instructions	Restart Address
RST 0	0000 H
RST 1	0008 H
RST 2	0010 H
RST 3	0018 H
RST 4	0020 H
RST 5	0028 H
RST 6	0030 H
RST 7	0038 H

- **CONTROL INSTRUCTIONS**

The control instructions control the operation of microprocessor.

Opcode	Operand	Description
NOP	None	No operation

No operation is performed. The instruction is fetched and decoded but no operation is executed.

Example: NOP

Opcode	Operand	Description
HLT	None	Halt

The CPU finishes executing the current instruction and halts any further execution. An interrupt or reset is necessary to exit from the halt state.

Example: HLT

Opcode	Operand	Description
DI	None	Disable interrupt

The interrupt enable f lip-f lop is reset and all the interrupts except the TRAP are disabled No flags are affected.

Example: DI

Opcode	Operand	Description
EI	None	Enable interrupt

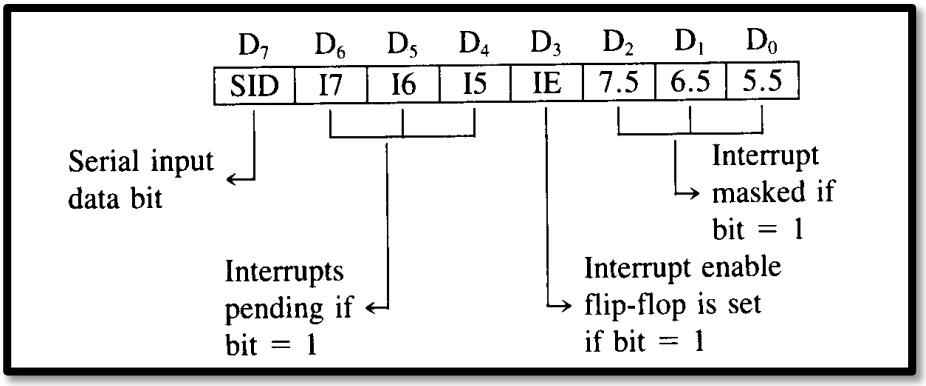
The interrupt enable f lip-f lop is set and all interrupts are enabled. No f lags are affected. This instruction is necessary to re-enable the interrupts (except TRAP).

Example: EI

Opcode	Operand	Description
RIM	None	Read Interrupt Mask

This is a multipurpose instruction used to read the status of interrupts 7.5, 6.5, 5.5 and read serial data Input bit. The instruction loads eight bits in the accumulator with the following interpretations.
Example: RIM

RIM INSTRUCTION

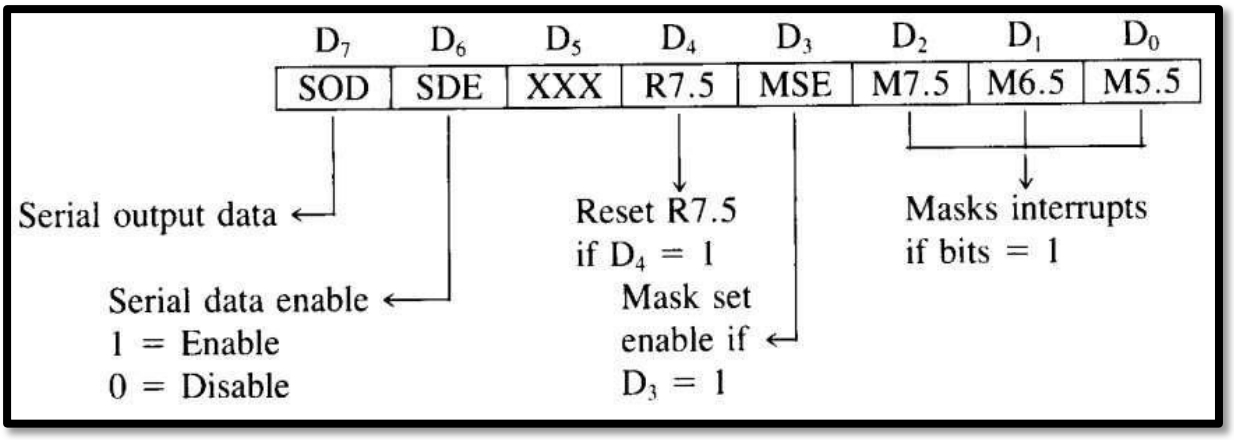


Opcode	Operand	Description
SIM	None	Set Interrupt Mask

This is a multipurpose instruction and used to implement the 8085 interrupts 7.5, 6.5, 5.5, and serial data output. The instruction interprets the accumulator contents as follows.

Example: SIM

SIM INSTRUCTION



EXERCISE:

Write and execute following programs in microprocessor kit.

1. Write an ALP to find out the square root of the given number. Store the result into the B register.
2. Write an ALP to find out the square of the given number. Store the result into the B register.
3. Write an ALP to load one 8bit number in D register increment it until AC flag is set. (Check the AC- auxiliary carry).
4. Write an ALP to decrement the content of DE register pair. Decrement it until lower 8 bit is set to 00.