MICROPROCESSOR AND MICROCONTROLLER

UNIT : 2

MICROPROCESSOR PROGRAMMING

2.1 INSTRUCTION SET OF INTEL 8085 :

The 8085 microprocessor instruction has 74 operation codes that results in 246 instructions.You are advised not to attempt to read all the instructions at one time. The following notations are used in the description of the instructions.

R	8085 8-bit register (A,B,C,D,E,H,L)
M	Memory register (location)
R_S	Register source
R_D	Register destination
R_P	Register pair (BC,DE,HL,SP)
( )	Contents of

Different types of instructions are given as follows:

1. DATA TRANSFER(COPY) INSTRUCTIONS: These instructions perform the following operations.

MNEMONICS	EXAMPLES	OPERATION
1.1 MVI R,8 bit	MVI B,4FH	Load 8-bit data byte (4FH)in a register B
1.2 MOV $R_{D,}$ R_S	MOV B, A	Copy data from source register R_S(A) to the destination register R_D (B)
1.3 LXI $R_{P\,,}$ 16 bit	LXI B, 2050H	Load 16-bit number(2050H) in a register pair(BC)
1.4 LDA 16-bit	LDA 2050H	Copy data byte into A from the memory specified by 16-bit address(2050H)
1.5 STA 16-bit	STA 2070H	Copy the data byte from A into a memory specified by 16-bit address(2070H

MNEMONICS	EXAMPLES	OPERATION
1.6 LDAX R_P	LDAX B	Copy the data byte into A from the memory specified by the address in the register pair.
1.7 STAX R_P	STAX D	Copy the data byte from A into the memory specified by the address in the register pair
1.8 IN 8-bit	IN 07H	Accept data byte from an input device(07H) and place it into the accumulator
1.9 OUT 8-bit	OUT 01H	Send data byte from the accumulator to an output device
1.10 MOV R,M	MOV B,M	Copy the data byte into register (B)from the memory specified by the address in HL register pair
1.11 MOV M,R	MOV M,C	Copy the data byte from the register(C) into the memory specified by the address in HL register pair

To realize the above instructions ,as a beginner you need to go through some small problem statement

Problem statement: Transfer(copy) the data byte from reg.B to accumulator(A) and then store the data byte in the memory location 9000H.

After going through the problem first impression will be in the mind of student as follows:

***** What does it mean ?

* How to think to proceed…..? See- the given problem means that some data byte (say, FFH) is already stored in the reg.B and you are asked to transfer the data byte to a new location accumulator(A) and a memory location 9000H.

Now it is clear my students that it is a data transfer operation and you need to go through the data transfer instructions.Now come to the steps of thinking given as follows:

Step-1: To transfer the data from reg.B to reg.A you might think it is MOV A,B. Yes, as given in instructions.

Step-2: Further to transfer the data to the memory address 9000H you might think it is STA 9000H as given in the instructions . Now I hope it is clear to you all my beloved students.

The questions arises how to write the program to solve the above problems.

I hope that you already know, each instruction has two parts: opcode and operand as given in unit-1, so you have to write the program in tabular form as follows:-

Memory address	Machine code	Label	Mnemonics	Operand	Comments
8000H			MOV	A,B	Copy the content of reg.B to reg.A
8001H			STA	9000H	Store the content A in the mem. Address(Location) 9000H.
8004H			HLT		End of program

Note:- In the above solution I know, you might think how suddenly memory address and machine codes are appearing in the above solution. Yes,your thinking is right. It will be clear to you when you will run the program in the laboratory.

Still I am trying to deliver some concept regarding memory address and machine code. As you know before running the program we have to store it in a certain memory location that is why memory location 8000H is selected to store the program and since the machine realizes only op-code in trainer kit that is why mnemonics are converted into machine codes which is available in your book and you need not memorize it. Machine codes are required only to serve the purpose during the execution of program in the laboratory. Hope that you might have gathered a little bit of confidence regarding instructions and programming in the above discussion.

2. ARITHMETIC INSTRUCTIONS

Mnemonics	Examples	Operation
2.1 ADD R	ADD B	Add the contents of the register B to the contents of the accumulator(A)
2.2 ADI 8-bit	ADI 37H	Add 8-bit data to the contents of A
2.3 ADD M	ADD M	Add the contents of memory to A; the Address of memory is in HL register pair
2.4 SUB R	SUB C	Subtract the contents of register C from The contents of A
2.5 SUI 8- bit	SUI 7FH	Subtract 8- bit data (7FH) from the Contents of A
2.6 SUB M	SUB M	Subtract the contents of memory from A; the address of memory is in HL regis-ter
2.7 INR R	INR D	Increment the contents of a register
2.8 INR M	INR M	Increment the contents of memory, the address of which is in HL
2.9 DCR R	DCR C	Decrement the contents of a register
2.10 DCR M	DCR M	Decrement the contents of memory, the address of which is HL

2.11 INX R_P	INX H	Increment the contents of HL register pair HL
2.12 DCX R_P	DCX B	Decrement the contents of BC reg.pair

3. LOGIC & BIT MANUPULATION INSTRUCTIONS.

MNEMONICS	EXAMPLES	OPERATIONS
3.1 ANA R	ANA B	Logically AND the content of reg.B With the content of register A
3.2 ANI 8-bit	ANI 2FH	Logically and 8-bit data with the con-Tents of A.
3.3 ANA M	ANA M	Logically AND the contents of memory With the contents of A; the address of Memory is in the HL register.
3.4 ORA R	ORA B	Logically OR the contents of register B With the contents of A.
3.5 ORI 8-bit	ORI 3FH	Logically OR the byte 3FH with the Contents of A.
3.6 ORA M	ORA M	Logically OR the contents of memory with the contents of A;the address of memory is in HL register pair.
3.7 XRA R	XRA B	Exclusive- OR the contents of register B with the contents of accumulator.
3.8 XRI 8-bit	XRI 3AH	Exclusive- OR the data byte 3AH with The contents of accumulator.
3.9 XRA M	XRA M	Exclusive-OR the contents of memory

		With the contents of A; the address of Memory is in HL register pair.

MNEMONICS	EXAMPLES	OPERATIONS
3.10 CMP R	CMP B	Compare the contents of register B With the contents of A for less than , Equal to or greater than.
3.11 CPI 8- bit	CPI 4FH	Compare data byte 4FH with the Contents of A for less than ,equal to Or greater than.

4. BRANCH INSTRUCTIONS: These instructions change the program sequence.

MNEMONICS	EXAMPLES	OPERATIONS
4.1 JMP 16-bit address	JMP 2050H	Change the program sequence to the specified address 2050H.
4.2 JZ 16-bit address	JZ 2080H	Change the program sequence to the specified address 2080H if the Zero flag is set.
4.3 JNZ 16-bit address	JNZ 2070H	Change the program sequence to the specified address 2070H if the Zero flag is reset.
4.4 JC 16-bit address	JC 2025H	Change the program sequence to the specified address 2025H if the Carry flag is set.

4.5 JNC 16-bit address	JNC 2030H	Change the program sequence to the specified address 2030H if the Carry flag is reset.

MNEMONICS	EXAMPLES	OPERATION
4.6 CALL 16-bit address	CALL 2075H	Change the program sequence To the location of a sub routine Specified by the address 2075H.
4.7 RET	RET	Return to the calling program After completing the sub routine.

5 MACHINE CONTROL INSTRUCTIONS

MNEMONICS	EXAMPLES	OPERATION
5.1 HLP	HLT	Stop processing and wait.
5.2 NOP	NOP	Do not perform any operation.

This set of instructions is a representative sample, further we have to include various instructions related to 16-bit data operation.

2.2 ADDRESSING MODES

It has already been explained that there are various techniques to specify data for instructions.These techniques are called addressing modes.

1. Direct Addressing

In this mode of addressing the address of the operand (data) is given in the instruction itself.

Examples are :

A. STA 2400H	Store the content of the accumulator in the memory location 2400H.
B. IN 02H	02H is the address of an input port from where the data is to be read. The data will be available in accumulator.

2. Register addressing

In register addressing mode the operand is in one of the general purpose registers.The opcode specifies the address of the registers in addition to the operation to be performed.

Examples are :

A. MOV A,B	Move the content of register B to register A.
B. ADD B	Add the content of register B to the content of register A.

3. Register Indirect Addressing

A. LXIH, 2500H	Load H-L pair with 2500H.
B. MOV A, M	Move the content of memory location to the accumulator.Memory location is Specified by the content of H-L register pair.

4. Immediate Addressing

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

Examples are :

A. MVI 05H	Move 05H in register A.
B. ADI 06H	add 06h to the content of the accumulator.

5. Implicit Addressing

There are certain instructions which operate on the content of the accumulator and do not require the address of the operand.

Examples are :

A. CMA	Complement the content of accumulator.
B. RLC	Rotate accumulator left.
C. RAL	Rotate accumulator left through carry.

The above instructions will be clear to you when some program will be done in the next part.

2.**3Introducing to branch and subroutine**

There are three types of branching instructions:

JUMP instruction :

a. Unconditional JUMP : Transfers the program sequence to the described memory address.

e.g . JUMP 2050H : Described in the above instructions.

b. Conditional JUMP : Transfers the program sequence to the described memory address if the conditions is satisfied.

E g. JC 2050H, JNC 2050H : Described in the above instructions as well as in the program given below.

CALL instruction :

a. . Unconditional CALL : Transfers the program sequence to the described memory address given in the operand e.g. CALL 2050H. : Described in the above instructions.

b. Conditional CALL : Transfers the program sequence to the described memory address given in the operand if the conditions is satisfied e.g. CNC 2050H ,CC 2050H. Described in the above instructions.

RETURN instruction : The return instruction transfers the program sequence from the subroutine to the calling program.

a. Unconditional RETURN : The return instruction transfers the program sequence from the subroutine to the calling program unconditionally e.g RET.

b. . Conditional RETURN: The return instruction transfers the program sequence from the subroutine to the calling program conditionally e.g RC, RNC.

Sub Routine: Sub Routine is a group of instructions written separately from the main program to perform a function that occurs repeatedly in the main program . This will be explained during programming.

2.4 **SIMPLE PROGRAM SUCH AS ADDITION, SUBTRACTION,MULTI-BYTE ADDTION,MULTIPLICATIONS OF TWO NUMBER.**

Program for addition: Add 02H and 03H(data bytes) and store the result(sum) in the memory location 9000H

Memory address	Machine code	Label	Mnemonics	Operand	Comments
8000H	3EH		MVI	A,02H	Data byte 02H will be loaded Into the accumulator.
8002H			MVI	B,03H	Data byte 03H will be loaded Into the register B.
8004H			ADD	B	Content of register B will be Added with the content of Accumulator.
8005H			STA	9000H	Store the content of A in the memory specified by address 9000H.
8008			HLT		Stop

Program for addition: Add two data bytes already stored in the memory location 2501H and 2502H and store the result(sum) in the memory location 9000H

Let 49H in memory location 2501H and 56H in location 2502H. SUM will be stored in 9000H.

Mem.Add.	M/C code(H)	Mnemonics	Operand	Comments
8000H	21,01,25	LXI	H, 2501 H	Get address of 1st number in H-L pair.

8003H	7E	MOV	A, M	1st number in accumulator.
8004H	23	INX	H	Increment the content of H-L pair.
8005H	86	ADD	M	Add 1st and 2nd number . SUM will be stored In the accumulator(A).
8006H	32,00,90	STA	9000H	Store the SUM in 9000H.
8009H	76	HLT		Stop.

<u>Program for addition with carry: Add two data bytes already stored in the memory location 2501H and 2502H and store the result(sum) in the memory location 2503HandCARRY IN 2504 H.</u>

Mem.Add.	M/C code(H)	LABEL	Mnemonics	Operand	Comments
2000H	21,01,25		LXI	H,2501 H	Get address of 1st number in H-L pair.
2003H	0E,00		MVI	C,00	MSBs of sum in register C. Initial Value=00
2005H	7E		MOV	A,M	1ST number in accumulator
2006H	23		INX	H	Address of 2Nd no. 2502 in H-L pair.
2007H	86		ADD	M	Add 1st and 2nd numbers.
2008H	D2,OC,20		JNC	AHEAD	Is carry? No, go to level AHEAD
200B	0C		INR	C	Yes, increment C.
200C	32,03,25	AHEAD	STA	2503H	LSBs of sum in 2503H
200F	79		MOV	A,C	MSBs of sum in accumulator
2010H	32,04,25		STA	2504H	MSBs(CARRY) of sum in 2504H.
2013H	76		HLT		Stop.

<u>MULTI –BYTE ADDITION</u>

Program for multi-byte addition: Let us have to add 6 data bytes A2,FA,DF,E5,98,8B already stored in the memory locations with starting address 9000H. After addition the result will be stored in the memory location 9000H and carry will be stored in the memory 9001H.

Mem.Add.	M/C code(H)	LABEL	Mnemonics	Operand	Comments
8000H	AF		XRA	A	Clear the content of accumulator.
8001H	47		MOV	B, A	Clear the content of reg.B
8002H	0E		MVI	C, 06H	Set up reg .C as a counter
8003H	06				
8004H	21, 00, 90		LXI	H, 9000H	Set up H-Lreg.as a memory index.
8007H	86	LOOP	ADD	M	Add (M) to (A)
8008H	D2 ,0C, 80		JNC	NEXT	If no carry,do not increment carry r
					Jump to increment the index
800BH	04		INR	B	If carry,save carry bit
800CH	23	NEXT	INX	H	Point to next memory location.
800DH	0D		DCR	C	One addition is completed. Decrement C
800EH	C2, 07, 80		JNZ	LOOP	If all bytes are not yet added,go back to loop.
8011H	32, 00, 90		STA	9000H	Store the SUM in the mem.location 9000H.
8014H	78		MOV	B, A	Transfer carry to accumulator.
8015H	32, 01, 90		STA	9001H	Store the carry in mem. Location 9001H
8018H	76		HLT		End of program.

8-BIT SUBTRACTION

Problem:- Subtract 32H from 49H which are already stored in the memory location 2501H and 2502H respectively. Store the result in the memory location 2503H.

Mem.Add.	M/C code(H)	LABEL	Mnemonics	Operand	Comments
2000H	21,01,25		LXI	H,2501H	Get address of 1st number in H-L Pair.
2003H	7E		MOV	A,M	1ST number in accumulator.
2004H	23		INX	H	Content of H-L pair increased By 1(From 2501H to 2502H).
2005H	96		SUB	M	1ST Number-2nd number
2006H	23		INX	H	Content of H-L pair becomes 2503H.
2007H	77		MOV	M,A	Store result in 2503H.
2008H	76		HLT		End of program.

.

2.5 Interrupt and Interrupt Service Routine

When Interrupt is initiated, the microprocessor stops the current execution of instruction- Performs the Interrupt function and then resumes its operation.

There are two kinds of Interrupts:-

A. **Hardware Interrupt:**

In 8085MPU, we have five Interrupts as given in the table below

Sl no.	Name	Priority	Vector address	Masking	Type of triggering
1	TRAP	Highest	0024H	Non maskable interrupt	Edge & Level triggered
2	RST 7.5		003CH	Maskable	Edge triggered
3	RST 6.5		0034H	Maskable	Level triggered
4	RST 5.5		002CH	Maskable	Level triggered
5	INTR	Lowest	Non vectored interrupt	Maskable	Level triggered

Now let us see the meaning of Vector interrupt, non Vector interrupt, Maskable interrupt(MI) non Maskable interrupt(NMI).

Vector Interrupt	: It means that interrupt address is known to the Microprocessor.Eg-RST 7.5,6.5,5.5,TRAP
Non Vector interrupt	: It means that the Add is not known to the Microprocessor. Eg- INTR
Maskable interrupt	: We can disable the interrupt by writting some instruction(DI) to the program. Eg- RST 7.5,6.5,5.5
Non Maskable interrupt	: We cannot disable the interrupt by writting instructions to the program.

Software Interrupt:

- Interrupt which can be introduced through instruction.
- RST n instruction is software interrupt.
- Vector address is fixed.
  Software interrupts are given in the table below:

Interrupt	Op-code	Vector address
RST 0	C7H	0000H
RST 1	CFH	0008H
RST 2	D7H	0010H
RST 3	DFH	0018H
RST 4	E7H	0020H
RST 5	EFH	0028H
RST 6	F7H	0030H
RST 7	FFH	0038H

Instructions for 8085 interrupts:

RIM, SIM, EI, DI.

RIM: It stands for Read Interrupt Mask.It is used to check status of all maskable interrupt.

When RIM instruction is executed, then the current status of interrupts & SID goes to the accumulator.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SID	I7.5	I 6.5	I 5.5	IE	M 7.5	M 6.5	M 5.5

Fig shows accumulator contents after the execution of instruction RIM.

BITS0-2(M 5.5,M 6.5,M 7.5): Interrupt masks; 1=masked

Bit 3(IE): Interrupt enable flag; 1=enabled

Bits 4-6(I 5.5,I 6.5,I 7.5): Pending interrupts; 1= pending

Bit 7(SID): Serial input data, if any.

After executing the interrupt service sub routine the processor checks wheather any other interrupt is pending using RIM instruction. If an interrupt is pending the processor executes its interrupt service sub routine before it returns to the main program.

SIM: This instruction reads the contents of the accumulator and enables or disables the interrupts according to the contents of the accumulator.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SOD	SDE	XXX	R 7.5	MSE	M 7.5	M 6.5	M 5.5

Bit 0-2(M 5.5, M 6.5, M 7.5): 0-Enable, 1-Disable(mask)

Bit D3(MSE): Mask set enable. If 0,bits 0-2 ignored.If 1 , mask is set.

R7.5(RESET RST 7.5) : If 1, RST 7.5 FF is reset OFF.

Bit 5: Ignored.

Bit 6(SDE): If 1, bit 7 is output to Serial output Data Latch.

Bit 7(SOD): Serial output Data : Ignored if bit 6=0.

Interrupt Service Routine(ISR) in 8085 MPU

Interrupt process in 8085 is explained in the following steps:-

Step-1: The interrupt process should be enabled by writing the EI instruction and disabled by DI instruction.

- EI(Enable Interrupt):
  -1 byte instruction

  -It is used to enable interrupt
- DI(Disable Interrupt):
  -1 byte instruction
  -It is used to disable interrupt

Step-2: During execution of program in 8085, the microprocessor checks the INTR line during execution of all instructions.

Step-3: If the INTR is high and Interrupt is enabled, the processor completes current instruction and then disable interrupt flip-flop and then sends INTA(Interrupt acknowledge, active low). The processor can not accept any interrupt request until the Interrupt flip-flop is enabled again

Step-4: The INTA is used to insert RST instruction through external hardware.

The RST instruction is 1-byte instruction which transfers control to specific location 0000H

Page.

Step-5: When microprocessor receives RST instruction, it saves the memory address of next Instruction on the stack.Then program control gets transferred to new location 00H page.

Step-6: After performing interrupt task, the processor again jumps to original program. That Sub Routine is known as ISR(Interrupt Service Routine).

Step-7: The ISR should include EI at the beginning.

Step-8: At the end of ISR, RET instruction resumes the execution of instruction from the original address of the program from where it responded to interrupt signal.

*IF ANY TYPOLOGICAL ERROR OCCURS, PLEASE BEAR WITH ME.