Microprocessor ,Assembly Language & Computer Interfacing Sessional

EEE-3212

Fareha Akter Juha
Adjunct Faculty, Dep. of CSE,
Leading University, Sylhet
Email: farehaakter02@gmail.com

Lab-3: 8085 Microprocessor Addressing Mode and solving some programs.

Addressing mode

The way of specifying data to be operated by an instruction is called addressing mode.

Addressing modes in 8085 can be classified into 5 groups −

1. Immediate addressing mode
2. Register addressing mode
3. Direct addressing mode
4. Indirect addressing mode
5. Implied addressing mode

farehaakter02@gmail.com

Immediate addressing mode

In immediate addressing mode, the source operand is always data. If the data is 8-bit, then the instruction will be 2 bytes, if the data is 16-bit then the instruction will be of 3 bytes.

Examples:

MVI B 45 (move the data 45H immediately to register B)

LXI H 3050 (load the H-L pair with the operand 3050H immediately)

JMP address (jump to the operand address immediately)

FAREHAAKTER02@GMAIL.COM

Register Addressing Mode

In register addressing mode, the data to be operated is available inside the register(s) and register(s) are (are) operands. Therefore the operation is performed within various registers of the microprocessor.

Examples:

MOV A, B (move the contents of register B to register A)

ADD B (add contents of registers A and B and store the result in register A)

INR A (increment the contents of register A by one)

FAREHAAKTER02@GMAIL.COM

Direct Addressing Mode –

In direct addressing mode, the data to be operated is available inside a memory location and that memory location is directly specified as an operand. The operand is directly available in the instruction itself.

Examples:

LDA 2050 (load the contents of a memory location into accumulator A)

LHLD address (load contents of 16-bit memory location into H-L register pair)

IN 35 (read the data from port whose address is 35)

FAREHAAKTER02@GMAIL.COM

Register Indirect Addressing Mode –

In register indirect addressing mode, the data to be operated is available inside a memory location and that memory location is indirectly specified by a register pair.

Examples:

MOV A, M (move the contents of the memory location pointed by the H-L pair to the accumulator)

LDAX B (move contents of B-C register to the accumulator)

STAX B (store accumulator contents in memory pointed by register pair B-C)

FAREHAAKTER02@GMAIL.COM

Implied/Implicit Addressing Mode

In implied/implicit addressing mode the operand is hidden and the data to be operated is available in the instruction itself.

Examples:

CMA (finds and stores the 1’s complement of the contents of accumulator A in A)

RRC (rotate accumulator A right by one bit)

RLC (rotate accumulator A left by one bit)

FAREHAAKTER02@GMAIL.COM

8085 program to subtract two 8 bit numbers

Problem – To perform subtraction of two 8 bits numbers without borrowing using 8085 microprocessor.
Example:

```
INPUT:
2050:02
2051:04

OUTPUT:
2052:02
```

Program:

Memory Address	Mnemonics	Comment
2000	LDA 2050	A<-[2050]
2003	MOV H, A	H<-A
2004	LDA 2051	A<-[2051]
2007	SUB H	A<-A-H
2008	STA 2052	A → 2052
200B	HLT	STOP

FAREHAAKTER02@GMAIL.COM

Algorithm:

1.Load the accumulator with the first data.

2.Move data of accumulator to register B.

3.Load the second data into the accumulator.

4.Subtract the content of register B with the content of the accumulator.

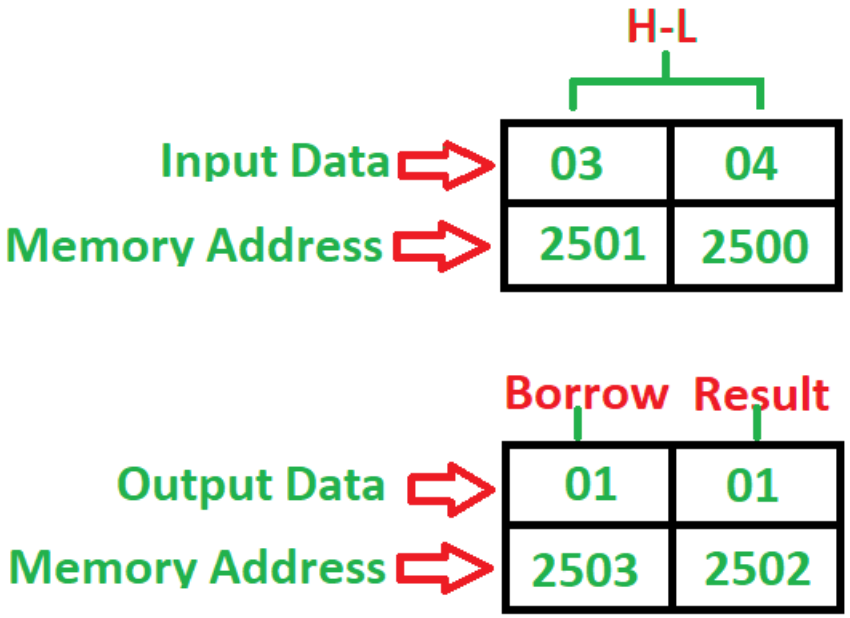
5.Now load the result value in a memory location.

FAREHAAKTER02@GMAIL.COM

8085 program to subtract two 8-bit numbers with or without borrow

Problem – Write a program to subtract two 8-bit numbers with or without borrowing where the first number is at 2500 memory address and the second number is at 2501 memory address and store the result into 2502 and borrow into 2503 memory address.

Example –



Algorithm –

1.Load 00 in a register C (for borrow)

2.Load two 8-bit number from memory into registers

3.Move one number to accumulator

4.Subtract the second number with accumulator

5.If borrow is not equal to 1, go to step 7

6.Increment register for borrow by 1

7.Store accumulator content in memory

8.Move content of register into accumulator

9.Store content of accumulator in other memory location

10.Stop

FAREHAAKTER02@GMAIL.COM

Memory	Mnemonics	Operands	Comment
2000	MVI	C, 00	[C] <- 00
2002	LHLD	2500	[H-L] <- [2500]
2005	MOV	A, H	[A] <- [H]
2006	SUB	L	[A] <- [A] – [L]
2007	JNC	200B	Jump If no borrow
200A	INR	C	[C] <- [C] + 1
200B	STA	2502	[A] -> [2502], Result
200E	MOV	A, C	[A] <- [C]
2010	STA	2503	[A] -> [2503], Borrow
2013	HLT		Stop

FAREHAAKTER02@GMAIL.COM

Explanation – Registers A, H, L, and C are used for general purposes:

1. **MOV** is used to transfer the data from memory to the accumulator (1 Byte)
2. **LHLD** is used to load register pair directly using a 16-bit address (3 Byte instruction)
3. **MVI** is used to move data immediately into any of the registers (2 Byte)
4. **STA** is used to store the content of the accumulator into memory(3 Byte instruction)
5. **INR** is used to increase the register by 1 (1 Byte instruction)
6. **JNC** is used to jump if no borrow (3 Byte instruction)
7. **SUB** is used to subtract two numbers where one number is in the accumulator(1 Byte)
8. **HLT** is used to halt the program

FAREHAAKTER02@GMAIL.COM

8085 program to multiply two 8 bit numbers

Problem – Multiply two 8 bit numbers stored at address 2050 and 2051. Result is stored at address 3050 and 3051. Starting address of program is taken as 2000.

Input Data ⇨	07	43
Memory Address ⇨	2051	2050

Output Data ⇨	01	D5
Memory Address ⇨	3051	3050

Algorithm –

1.We are taking adding the number 43 seven(7) times in this example.

2.As the multiplication of two 8-bit numbers can be maximum of 16 bits so we need to register pair to store the result.

Program –

Memory Address	Mnemonics	Comment
2000	LHLD 2050	H←2051, L←2050
2003	XCHG	H↔D, L↔E
2004	MOV C, D	C←D
2005	MVI D 00	D←00
2007	LXI H 0000	H←00, L←00
200A	DAD D	HL←HL+DE
200B	DCR C	C←C-1
200C	JNZ 200A	If Zero Flag=0, goto 200A
200F	SHLD 3050	H→3051, L→3050
2012	HLT	

FAREHAAKTER02@GMAIL.COM

Explanation – Registers used: A, H, L, C, D, E

1. LHLD 2050 loads content of 2051 in H and content of 2050 in L
2. XCHG exchanges contents of H with D and contents of L with E
3. MOV C, D copies content of D in C
4. MVI D 00 assigns 00 to D
5. LXI H 0000 assigns 00 to H and 00 to L
6. DAD D adds HL and DE and assigns the result to HL
7. DCR C decrements C by 1
8. JNZ 200A jumps program counter to 200A if zero flag = 0
9. SHLD stores value of H at memory location 3051 and L at 3050
10. HLT stops executing the program and halts any further execution

Thank You

Farehaakter02@gmail.Com