# Microprocessor

BCA 3rd Semester 2020

# Lecture- 12

**Subhadip MUKHERJEE**

Department of computer science

Kharagpur college

# Assembly Language Programming

**Exchange the contents of two memory locations.**

```
LDA 2000H ; "Get the contents of memory location 2000H into accumulator"
MOV B, A ; "Save the contents into B register"
LDA 4000H ; "Get the contents of memory location 4000Hinto accumulator"
STA 2000H ; "Store the contents of accumulator at address 2000H"
MOV A, B ; "Get the saved contents back into A register"
STA 4000H ; "Store the contents of accumulator at address 4000H"
```

# Assembly Language Programming

## Multiply two 8 bit data using successive addition method

```
        MVI D,00 Initialize register D to 00
        MVI A,00 Initialize Accumulator content to 00
        LXI H,4150
        MOV B,M Get the first number in B - reg
        INX H
        MOV C,M Get the second number in C- reg.
LOOP:   ADD B Add content of A - reg to register B.
        JNC NEXT Jump on no carry to NEXT.
        INR D Increment content of register D
NEXT:   DCR C Decrement content of register C.
        JNZ LOOP Jump on no zero to address
        STA 4152 Store the result in Memory
        MOV A, D
        STA 4153 Store the MSB of result in Memory
        HLT Terminate the program.
```

# Assembly Language Programming

## One’s complement of a 8 bit number

LDA 4400B : "Get the number"

CMA : "Complement number"

STA 4300H : "Store the result"

HLT : "Terminate program execution"

(4400H) = 55H

Result = (4300B) = AAB

# Assembly Language Programming

**Two’s complement of a 8 bit number.**

LDA 4200H : "Get the number"

CMA : "Complement the number"

ADI, 01 H : "Add one in the number"

STA 4300H : "Store the result"

HLT : "Terminate program execution"

(4200H) = 55H

Result = (4300H) = AAH + 1 = ABH

# Assembly Language Programming

**Evaluate X (Y + Z), where X = 10 H, Y = 20 H and Z = 30 H.**

2000 MOV AL, 20 H : 20 H put in AL

2002 MOV CL, 30 H : 30 H put in CL

2004 ADD AL, CL : AL and CL are added up and result in AL

2006 MOV CL, AL : AL transferred in CL

2008 MOV AL, 10 H : 10 H put in AL

200A MUL CL : AL and CL are multiplied and result in AL

200C MOV SI, 4000 H : Source address in SI

200F MOV SI, AL : AL put in SI

2011 HLT : Stop.

# Assembly Language Programming

**Find the minimum value of a byte from a string of bytes**

2000 MOV SI, 3000 H : Source address put in SI

2003 MOV CX, 0100 H : Count value of bytes put in CX

2006 MOV AH, 00 H : AH initialized with 00H

2008 CMP AH, [SI] : AH compared with data pointed to by SI

200A JB 200E H : Jump if (AH) < (SI) to 200E H

200C MOV AH, [SI] : Otherwise move (SI) to AH

200E INC SI : Increment SI

200F LOOPNZ 2008 : Loop unless CX ¯ 0

2011 MOV [SI], AH : (AH) transferred to the memory location pointed to by SI

2013 HLT : Stop.

# Assembly Language Programming

**Sort an array of 8 bit data in ascending order.**

```
LXI H,5000 ;Set pointer for array
MOV C,M ;Load the Count
DCR C ;Decrement Count
REPEAT: MOV D,C
LXI H,5001
LOOP: MOV A,M ;copy content of memory location to Accumulator
INX H
CMP M
```

# Assembly Language Programming

**Sort an array of 8 bit data in ascending order (Continued)**

```
JC SKIP ;jump to skip if carry generated
MOV B,M ;copy content of memory location to B - Register
MOV M,A ;copy content of Accumulator to memory location
DCX H ;Decrement content of HL pair of registers
MOV M,B ;copy content of B - Register to memory location
INX H ;Increment content of HL pair of registers
SKIP: DCR D ;Decrement content of Register - D
JNZ LOOP ;jump to loop if not equal to zero
DCR C ;Decrement count
JNZ REPEAT ;jump to repeat if not equal to zero
HLT ;Terminate Program
```

# Assembly Language Programming

## Square of a Number Using Look Up Table.

### Algorithm

1. Initialize HL pair to point Look up table
2. Get the data
3. Check whether the given input is less than 9
4. If yes go to next step else halt the program
5. Add the desired address with the accumulator content
6. Store the result

# Assembly Language Programming

## Square of a Number Using Look Up Table.

```
LXI H,5000 ;Initialize Look up table address
LDA 5050 ;Get the data
CPI 0A ;Check input > 9
JC AFTER ;if yes error
VI A,FF ;Error Indication
STA 5051
HLT
AFTER: MOV C,A ;Add the desired Address
MVI B,00
DAD B
MOV A,M
STA 5051 ;Store the result
HLT ;Terminate the program
```

# Assembly Language Programming

## Square of a Number Using Look Up Table.

**LOOKUP TABLE:**

5000 01

5001 04

5002 09

5003 16

5004 25

5005 36

5006 49

5007 64

5008 81

**Result**

Input:
Data: 05H in memory location 5050

Output:
Data: 25H (Square of 5) in memory location 5051

# Thank You