BITS Pilani
Hyderabad Campus
Department of Electrical Engineering

Microprocessor Programming & Interfacing

CS/ECE/EEE/INSTR F241

Tutorial-6
(Arithmetic Instructions)

Date : 22/02/2021
24/02/2021

Problem-1

Write an assembly language program in 8086 microprocessor to find square root of a number.

Problem-1

Write an assembly language program in 8086 microprocessor to find square root of a number.

Solution:

Algorithm:

1. Move the input data in register AX
2. Move the data 0000 in CX and FFFF in BX
3. Add 0002 to the contents of BX
4. Increment the content of CX by 1
5. Subtract the contents of AX and BX
6. If Zero Flag(ZF) is not set go to step 3 else go to step 7
7. Store the data from CX to offset 600
8. Stop

❖N.B.: Will only work for perfect squares.

Problem-1

Write an assembly language program in 8086 microprocessor to find square root of a number.

Solution:

```
      MOV AX, [0500H]   //Place where the number is stored
      MOV CX, 0000H
      MOV BX, 0FFFFH
L1:   ADD BX, 02H
      INC CX
      SUB AX, BX
      JNZ L1
      MOV [0600H], CX //Place where the result is stored
      HLT
```

❖N.B.: Will only work for perfect squares.

Problem-2

Write a program to find the min value in a given array in assembly 8086 microprocessor.

Problem-2

Write a program to find the min value in a given array in assembly 8086 microprocessor

- **Solution:**

1. Assign value 500 in SI and 600 in DI
2. Move the contents of [SI] in CL and increment SI by 1
3. Assign the value 00 H to CH
4. Move the content of [SI] in AL
5. Decrease the value of CX by 1
6. Increase the value of SI by 1
7. Move the contents of [SI] in BL
8. Compare the value of BL with AL
9. Jump to step 11 if carry flag is set
10. Move the contents of BL in AL
11. Jump to step 6 until the value of CX becomes 0, and decrease CX by 1
12. Move the contents of AL in [DI]
13. Halt the program

Problem-2

Write a program to find the min value in a given array in assembly 8086 microprocessor

.

ADDRESS	MNEMONICS	COMMENTS
0400	MOV SI, 500H	SI ← 500
0403	MOV DI, 600H	DI ← 600
0406	MOV CL, [SI]	CL ← [SI]
0408	MOV CH, 00 H	CH ← 00
040A	INC SI	SI ← SI+1
040B	MOV AL, [SI]	AL ← [SI]
040D	DEC CX	CX ← CX-1
040E	INC SI	SI ← SI+1
040F	MOV BL, [SI]	BL ← [SI]
0411	CMP AL, BL	AL-BL
0413	JC 0418	Jump if carry is 1
0416	MOV AL, BL	AL ← BL
0418	LOOP 040E	Jump if CX not equal to 0
041B	MOV [DI], AL	[DI] ← AL
041D	HLT	End of the program

Problem-3

- Write an assembly language program in 8086 microprocessor to search a number in a string of 5 bytes, store the offset where the element is found in DX and the number of iterations used to find the number in BX.
- For e.g. if the numbers in the memory location starting from location 0600H is given as follows and we have to find the number 25:

0600	0601	0602	0603	0604
45	A5	25	78	9C

Output →

0602	0002
DX	BX

Problem-3

```
MOV AX, 2000      //starting location of ES
MOV ES, AX
MOV DI, 600 //starting location of first string
MOV AL, 25  //The number to search
MOV CX, 0005      //No. of items
MOV BX, CX
CLD
REPNE SCASB //   Repeat till ZF = 0. Scan value from [DI]
                 and compare with AL, Increment DI
DEC DI
MOV DX, DI
SUB BX, CX
DEC BX
INT 03H
```

SCAS

Compares the AL with a byte of data in memory
Compares the AX with a word of data in memory
Compares the EAX with a doubleword of data in memory

Memory is ES: DI
Operands not affected flags affected(subtraction)

SCASB
SCASW
SCASD

Can be used with prefix
REPNE SCASB

- Scanning is repeated as long as bytes are not equal or the end of the string not reached.

- SCAS uses direction flag (D) to select auto- increment or auto-decrement operation for DI.

Problem-4

Write an ALP to find 2’s complement of a string of 100 bytes.

NOT and NEG

- NOT and NEG can use any addressing mode except segment register addressing.
- The **NOT instruction inverts all bits** of a byte, word, or double word.
- **NEG two's complements a number**.
  - the arithmetic sign of a signed number changes from positive to negative or negative to positive

- The NOT function is considered logical
- NEG function is considered an arithmetic operation.

NOT and NEG

- NOT Destination ; Invert each bit of operand
- Ex: NOT BX ; complement the contents of BX register
- NOT BYTEPTR [SI]

- NEG Destination ; Form 2’s complement

Ex: NEG BX ; Replace the contents in BX with it’s 2’s complement

- ✓ This instruction forms the 2’s complement by subtracting the word or byte indicated in destination from zero.

Problem-4

Write an ALP to find 2’s complement of a string of 100 bytes.

```
2000    CLD                     : Clear direction flag
2001    MOV SI, 4000 H          : source address put in SI
2004    MOV DI, 5000 H          : Destination address put in DI
2007    MOV CX, 0064 H          : Put the number of bytes to be 2’s
                                   Complemented in CX
200A    LODSB                   : Data byte to AL and INC SI
200B    NEGAL                   : 2’s Complement of AL
200D    STOSB                   : Current AL value into DI and INC DI
200E    LOOP 200A H            : Loop till CX = 0.
2010 HLT                        : Stop.
```

Problem-5

- Write an ALP that will examine a set of 20 memory locations that have alphabets and count the number of vowels. The alphabets are store from memory location *3000H* and the count of the vowels must be stored in location *4000H*.

```
        MOV BX, 0000H
        MOV SI, 3000H
        MOV DI, 4000H
        MOV CX,0014H
x1:     MOV AL,[SI]
        CMP AL, ‘a'
        JNZ  x2
        INC BX
x2:     CMP AL, 'e'
        JNZ x3
        INC BX
x3:     CMP AL,'i'
        JNZ x4
        INC BX
x4:     CMP AL,'o'
        JNZ x5
        INC BX
x5:     CMP AL,'u'
        JNZ x6
        INC BX
x6:     INC SI
        DEC CX
        JNZ x1
        MOV [DI ], BX
        END
```