LAB - 2

Aim of this lab: Analyzing and Evaluating the Branching operation in the 8086 Microprocessor.

OBJECTIVE - ①: Multiplication of two 16 bit numbers without using MUL instruction in direct addressing mode.

① Pseudocode :

1) Set DS = 0000H
2) Set SI = 4000H
3) Load CX with the 1st 16 bit number from memory at [SI].
4) Increment SI twice to point to the next 16-bit number
5) Load BX with the second 16 bit number from memory at [SI]
6) Initialise DX = 0000H (high part of result)
7) Initialise AX = 0000H (low part of result)

8) Loop L2: (CX times)
   a) Add the second number ([SI]) to AX
   b) If carry occurs (overflow in AX):
      - increment DX (propagating carry to higher bits)
      - Jump to L1 (carry handling)
   c) L1: (Return here if no carry).

d) Decrement CL (lower byte of CX)
e) If CL is not zero, repeat loop L2

9) Move to next memory location.
10) Store the lower 16 bits (AX) of the result in memory.
11) Move to the next memory location.
12) Store the higher 16 bits (DX) of the result in memory.
13) Halt execution.

② Assembly Code :

```
1) MOV AX,0000H ;      AX ← 0000H
2) MOV DS, AX   ;      DS ← 0000H
3) MOV SI, 4000H ;     SI ← 4000H
4) MOV CX, [SI] ;      CX ← [4000H]
5) INC SI       ;      SI ← SI +1
6) INC SI       ;      SI ← SI +1
7) MOV BX, [SI] ;      BX ← [4002 H]
8) MOV DX, 0000H ;     DX ← 0000H
9) MOV AX, 0000H ;     AX ← 0000 H
10) L2: ADD AX, [SI] ; AX ← AX +[4002H]
11) JNC L1 ;  {If no carry, jump to L1 (continue addition)}
12) INC DX ;  {if carry occurs, increment DX }
13) L1: DEC CL ;  CL ← CL -1
14) JNZ L2 ;  {if CL is not 0, repeat L2 (continue
                repeated addition) }
```

15) INC SI ; SI ← SI + 1

16) INC SI ; SI ← SI + 1

16) MOV [SI], AX ; {Store lower 16 bit result (AX) in memory at [4004H]}

18) INC SI ; SI ← SI + 1

19) INC SI ; SI ← SI + 1

20) MOV [SI], DX ; {Store upper 16 bit result (DX) in memory at [4006 H]}

21) HLT

③ INPUT / OUTPUT ANALYSIS:

(a) ~~After~~ loading the values into registers:

- ⊙ CX = 0005 (multiplier)
- ⊙ BX = [SI] = 0012 (multiplicand)

} INPUTS

- ⊙ AX = DX = 0

(b) Loop L2 : repeated addition

- ⊙ iteration 1: AX = 0012
- ⊙ iteration 2: AX = 0024
- ⊙ iteration 3: AX = 0036
- ⊙ iteration 4: AX = 0048
- ⊙ iteration 5: AX = 005A
- ⊙ iteration

(c) Store the Results in memory
- [4004H] = 005AH (lower 16 Bits of the product)
- [4006H] = 0000H (Higher 16 Bits, as there's no over flow)

④ OBSERVATION TABLE :

Input:

Sl.No	Memory location	operand (Data)
1	4000H	0005H
2	4002H	0012H

Output:

Sl.No	Memory location	operand (Data)
1	4004H	005AH
2	4006H	0000H

OBJECTIVE-② : Find the sum and average of N 16 Bit numbers.

① PSEUDOCODE :

1) Set DS = 0000H
2) Set SI = 4000H
3) Load the amount of Numbers (N) into CX.
4) Copy CX to BX (BX will be used for division)
5) Initialise AX = DX = 0 (Sum registers)
6) Loop L2 (Repeat N times) :
   a) Move to the Next Number in memory (SI += 2)

b) Add the Number to AX (Sum Calculation)
c) If Carry occurs, increment DX (handle overflow)
d) Decrement CL (lower byte of CX)
e) Repeat if CL is not zero.

8) Move to next memory location.
9) Store the Sum:

a) Store the lower 16 Bits (AX) at [SI]
b) MovE SI forward
c) Store the upper 16 Bits (DX) at [SI]

10) Divide sum (AX : DX) by N (BX):
11) Store the quotient (average):

a) Store the lower 16 bits of the average at [SI]
b) MovE SI forward
c) Store the upper 16 bits of the average at [SI].

12) Halt execution.

② ASSEMBLY CODE:

1) MOV SI, 4000H ; SI ← 4000H
2) MOV CL, [SI] ; CL ←[4000H] {Load N, Count of Numbers, into CL}

3) MOV CH, 00H ; CH ← 00H {clear upper byte of CX, CX now holds N}

4) MOV BX, CX ; BX ← CX {Copy N into BX for later division}

sum by N, quotitent ~~by~~ in Ax, remainder in DX }

Store Average in memory

19) INC SI ; } SI = SI+2
20) INC SI ;
21) MOV [SI], AX ; {Store lower 16 Bits of average at [SI] }
22) INC SI } SI += 2
23) INC SI
24) MOV [SI] ,DX ; { Store ~~lower~~ upper 16 bits of average at memory [SI] }

25) HLT

③ INPUT / OUTPUT ANALYSIS :

(a) loading the values into 4000H as [0004 H]

b) input the 4 values into :

- 4002H ← ~~000~~ 2211 H
- 4004H ← 4433 H
- 4006H ← 6655 H
- 4008H ← 8877H

c) Now, computing the sum = 2211 + 4433 + 6655 + 8877

So, sum = 15510 H

d) Now, Storing sum :

5) MOV DX, 0000H ; DX ← 0000H {DX stores overflow}

6) MOV AX, 0000H ; AX ← 0000H {AX stores sum}

7) Loop L2 : Sum Calculation Loop (N times)

8) L2: INC SI ; SI = SI+1
INC SI ; SI ← SI+1
ADD AX, [SI] ; AX ← AX + [SI]
JNC L1 ; if NO carry jump to L1.
INC DX ; DX ← DX +1

8) Loop Control :

9) L1 : DEC CL ; CL ← CL-1
JNZ L2 ; {if CL ≠ 0 ; repeat the loop}

10) Store Sum in Memory

11) INC SI ; SI ← SI+1

12) INC SI ; SI ← SI+1

13) MOV [SI], AX ; {Store lower 16 Bits of Sum at memory [SI]}

14) INC SI ; } SI ← SI+2

15) INC SI ;

16) MOV [SI], DX ; {Stores upper 16 Bits of Sum at memory [SI]}

17) Compute Average (Sum/N)

18) DIV BX ; AX : DX ← AX : DX / BX {Divide

400AH ← 5510 H {lower 16 Bits}
400CH ← 0001 H {upper 16 Bits} - Carry

e) Now, Computing Average = $\frac{15510}{4}$ = 5544 H

f) Storing Average :

400EH ← 5544 H {lower 16 Bits}
4010 H ← 0000 H {upper 16 Bits} - No Remainder

④ OBSERVATION TABLE :

Input :

Sl.No	Memory location	Data
1	4000H	0004H
2	4002H	2211 H
3	4004H	4433H
4	4006H	6655H
5	4008H	8877H

Output :

Sl.No	Memory location	operand (Data)
1	400AH	5510H
2	400CH	0001 H
3	400EH	5544H
4	4010H	0000H

OBJECTIVE - ③ : Count No of 0's in an 8 bit Number.

① PSEUDOCODE :

1) Set AX = 0000H
2) Set DS = AX
3) Set SI = 2000H {memory Pointer}
4) Load the 8 bit number stored at BX into AL.

5) Set CL = 08H {loop counter for 8 bits}
6) Set BL = 00H {Counter for number of 0s}
7) Repeat until CL becomes 0 :
   a) Shift AL right by 1 bit (LSB moves to Carry flag)
   b) If Carry flag (CF) = 1, go do step d (skip increment)
   c) Increment BL (increase count of 0s)
   d) Decrement CL
   e) If CL ≠ 0, repeat from 7(a).
8) After the loop, store the count of 0s at memory SI+1.
9) HLT.

② ASSEMBLY CODE :

1) MOV SI, 2000H ; SI ← 2000H
2) MOV AL, [BX] ; AL ← [BX] {load 8 bit number from memory at BX.}
3) MOV CL, 08 H ; CL ← 08 H {Set loop Counter to 8}
4) MOV BL, 00H ; BL ← 00H {Set 0-bit Counter to 0}
5) L2 : SHR AL, 01H ; AL ← AL >> 1 {Right Shift - LSB moves to Carry flag}

6) JC L1 : {If CF = 1 (bit = 1), jump to L1 (skip increment) }

7) INC BL : { BL ← BL + 1 (increment count of 0s) }

8) L1 : DEC CL ; CL ← CL - 1

9) JNC L2 ; if CL ≠ 0, repeat loop

10) INC SI ; SI ← SI + 1

11) mov [SI], BL ; Store count of 0s at memory [SI]

12) HLT

③ INPUT / OUTPUT ANALYSIS : {input = 38}

Step	Instruction	AL (Binary)	CF (Carry flag)	BL (Zero count)	CL (Loop Counters)
1	MOV AL, [BX]	00100110	-	0 (00H)	8 (08H)
2	SHR AL, 01H	00010011	0	1 (01H)	7 (07H)
3	SHR AL, 01H	00001011	1	1 (01H)	6 (06H)
4	SHR AL, 01H	00000100	1	1 (01H)	5 (05H)
5	SHR AL, 01H	00000010	0	2 (02H)	4 (04H)
6	SHR AL, 01H	00000001	0	3 (03H)	3 (03H)
7	SHR AL, 01H	000000000	1	3 (03H)	2 (02H)
8	SHR AL, 01H	000000000	0	4 (04H)	1 (01H)
9	SHR AL, 01H	00000000	0	5 (05H)	0 (00H)

At this point, loop ends since CL = 0.

(4) OBSERVATION TABLE :

Input :

S.No	Memory location	Data
1	2000H	38

Output :

Sl No	Memory location	Data
1	2001 H	05

OBJECTIVE (4) : Move a block of 16 bit data from one location to another.

(1) PSEUDOCODE :

1) Set SI = 4000 H { Source memory address }
2) Set DI = 5000 H { Destination memory address }
3) Set CL = 03 { Loop Counter for Number of 16 bit values to copy }
4) Repeat until CL becomes 0 :
   a) Load BX with the 16 bit value from memory at SI
   b) Store BX at memory location DI
   c) Increment SI by 2 { Move to next 16 bit value }
   d) Increment DI by 2 { move to next destination location }
   e) Decrement CL
   f) If CL ≠ 0 , repeat from 6(a)
5) HLT execution.

② ASSEMBLY CODE :

1) MOV SI, 4000H ; SI ← 4000H {Set Source index to location 4000H}
2) MOV DI, 5000H ; DI ← 5000H {Set Destination index to memory location 5000H}
3) MOV CL, 03H ; CL ← 03H {loop Counter for 3 iterations}
4) L1: MOV BX, [SI] ; BX ← [SI] {load 16 Bit data from Source}
5) MOV [DI], BX ; [DI] ← BX {Store 16 bit data to destination}
6) INC SI
   INC SI } Increment Source Index, Move to next 16 bit data
7) INC DI
   INC DI } Increment Destination Index, Move to next 16 bit destination.
8) DEC CL ; CL ← CL-1
9) JNZ ~~L~~ L1; if CL ≠ 0 , repeat loop
10) HLT

③ INPUT / OUTPUT ANALYSIS :

1) The program Copies 3 16 Bit values from memory locations 4000H, 4002H, 4004H to 5000H, 5002H, 5004H respectively.
2) Loop executes 3 times { CL = 03H }
3) SI & DI are incremented by 2 after each transfer.

4) OBSERVATION TABLE:

Input:

Sl.No	Memory location	operand
1	4000H	0012
2	4002H	0034
3	4004H	0056

Output:

Sl.No	Memory location	operand
1	5000H	0012
2	5002H	0034
3	5004H	0056

CONCLUSION:

In this lab, we implemented fundamental 8086 processor assembly programs like multiplication, addition, division, counting zeros, data transfer etc and we learned how data processing is done using registers and memory, improving our understanding of assembly programming.