

DHAKA UNIVERSITY OF ENGINEERING & TECHNOLOGY, GAZIPUR-1707
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



Course No: CSE 3812

Course Title: Microprocessor and Interfacing Sessional

Lab 5

Report Name: Logic, Shift and Rotate Instructions & Multiplication and Division Instructions in EMU8086.

Date of Allocation: 10/11/2024

Date of Submission: 23/11/2024

Submitted To:

Dr. Md. Jakirul Islam

Associate Professor, Department of CSE

Dhaka University of Engineering & Technology, Gazipur

Sumaya Khatun

Lecturer, Department of CSE

Dhaka University of Engineering & Technology, Gazipur

Submitted By:

Name: Imran Raza

Student ID: 214097

Year/Semester: 3rd Year/1st Semester

Section: B

Objectives:

To understanding Logic, Shift and Rotate Instructions & Multiplication and Division Instructions in EMU8086.

Problem Statements:

1. Write a program to multiply AX by 27 using only Shift and Add instructions. You should not use the MUL instruction.

Recall that shifting left n bits multiplies the operand by 2^n .

If the multiplier is not an absolute power of 2,
then express the multiplier as a sum of terms which are absolute powers of 2.

For example, multiply AX by 7. ($7 = 4 + 2 + 1 = 2^2 + 2^1 + 1$)

Answer = AX shifted left by 2 + AX shifted left by 1 + AX.

Note: Only the original value of AX is used in each operation above.

2. Write a program to divide AX by 8 using Shift instructions. You should not use the DIV instruction. Assume AX is a multiple of 8.

Recall that shifting right n bits divides the operand by 2^n .

3. Write a program to check if a byte is a Palindrome. [Hint: Use Rotate instructions]. If the byte is a Palindrome, then move AAh into BL. Otherwise move 00h in BL.

A Palindrome looks the same when seen from the left or the right.

For example, 11011011 is a Palindrome but 11010011 is not a Palindrome

4. Write a program to display the bits of a register or memory location. Use the INT 21H interrupts to display data on the display monitor. [Hint: Use logical shift instruction to move data bit into the carry flag]

For example, if AL = 55H, then your program must display:

AL = 0 1 0 1 0 1 0 1

5. Write assembly code for each of the following high-level language assignment statements. Suppose that A, B, and C are word variables and all products will fit in 16 bits. Use IMUL for multiplication. It's not necessary to preserve the contents of variables A, B, and C.
 - a. $A = 5 \times A - 7$
 - b. $B = (A-B) \times (B-10)$

Problem 1:

ORG 0100H

.DATA

N DW ?

RESULT DW ?

.CODE

MAIN PROC

MOV AX,@DATA

MOV DS,AX

;INPUT

;FAST BX=0

XOR BX,BX

INPUT_LOOP:

;CHAR INPUT

MOV AH,1

INT 21H

;IF \N\R, STOP TAKING INPUT

CMP AL,10

JE END_INPUT_LOOP

CMP AL,13

JE END_INPUT_LOOP

;FAST CHAR TO DIGIT

;ALSO CLEARS AH

AND AX,000FH

;SAVE AX

MOV CX,AX

;BX=BX*10+AX

MOV AX,10

MUL BX

ADD AX,CX

MOV BX,AX

JMP INPUT_LOOP

END_INPUT_LOOP:

MOV N,BX

XOR AX,AX

;SHITHING BY 4

MOV BX,N

MOV CX,4

SHL BX,CL

ADD AX,BX

;SHIFTING BY 3

MOV BX,N

MOV CX,3

SHL BX,CL

ADD AX,BX

;SHIFTING BY 1

MOV BX,N

SHL BX,1

ADD AX,BX

;ADD ONE MORE

ADD AX,N

MOV RESULT,AX

MAIN ENDP

END MAIN

RET

edit: D:\CSE 3-1\Sessional\Microprocessor and Interfacing Sessional\lab

file edit bookmarks assembler emulator math ascii codes h

new open examples save compile emulate

```

01 ORG 0100H
02 .DATA
03 N DW ?
04 RESULT DW ?
05 .CODE
06 MAIN PROC
07     MOV AX,@DATA
08     MOV DS,AX
09     ;INPUT
10     ;FAST BX=0
11     XOR BX,BX
12 INPUT_LOOP:
13     ;CHAR INPUT
14     MOV AH,1
15     INT 21H
16     ;IF \N\R, STOP TAKING INPUT
17     CMP AL,10
18     JE END_INPUT_LOOP
19     CMP AL,13
20     JE END_INPUT_LOOP
21     ;FAST CHAR TO DIGIT
22     ;ALSO CLEARS AH
23     AND AX,000FH
24     ;SAVE AX
25     MOV CX,AX
26     ;BX=BX*10+AX
27     MOV AX,10
28     MUL BX
29     ADD AX,CX
30     MOV BX,AX
31     JMP INPUT_LOOP
32 END_INPUT_LOOP:
33     MOV N,BX
34     XOR AX,AX
35     ;SHITHING BY 4
36     MOV BX,N
37     MOV CX,4
38     SHL BX,CL
39     ADD AX,BX

```

line: 57 col: 1

edit: D:\CSE 3-1\Sessional\Microprocessor and Interfacing Sessional\lab 5\PROC

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calc

```

39     ADD AX,BX
40     ;SHIFTING BY 3
41     MOV BX,N
42     MOV CX,3
43     SHL BX,CL
44     ADD AX,BX
45     ;SHIFTING BY 1
46     MOV BX,N
47     SHL BX,1
48     ADD AX,BX
49     ;ADD ONE MORE
50     ADD AX,N
51
52     MOV RESULT,AX
53
54     MAIN ENDP
55 END MAIN
56 RET
57

```

variables

size: word elements: 1

edit show as: signed

N 10

RESULT 270

emulator: PROBL

file math debug

Load re

registers

	H	L
AX	01	0E
BX	00	14
CX	00	03
DX	00	00
CS	0700	
IP	0167	
SS	0700	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

emulator screen (80x25 chars)

```

10

```

line: 57

Problem 2:

ORG 0100H

.DATA

N DW ?

RESULT DW ?

.CODE

MAIN PROC

MOV AX,@DATA

```
MOV DS,AX
;INPUT
;FAST BX=0
XOR BX,BX
INPUT_LOOP:
;CHAR INPUT
MOV AH,1
INT 21H
;IF \N\R, STOP TAKING INPUT
CMP AL,10
JE END_INPUT_LOOP
CMP AL,13
JE END_INPUT_LOOP
;FAST CHAR TO DIGIT
;ALSO CLEARS AH
AND AX,000FH
;SAVE AX
MOV CX,AX
;BX=BX*10+AX
MOV AX,10
MUL BX
ADD AX,CX
MOV BX,AX
JMP INPUT_LOOP
END_INPUT_LOOP:
MOV N,BX
XOR AX,AX
;SHITHING BY 3
MOV BX,N
MOV CX,3
SHR BX,CL
ADD AX,BX
```

MOV RESULT,AX

MAIN ENDP

END MAIN

RET

edit: D:\CSE 3-1\Sessional\Microprocessor and Interfacing Sessional\lab 5\PROB3.ASM

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate

01 ORG 0100H
02 .DATA
03 N DW ?
04 RESULT DW ?
05 .CODE
06 MAIN PROC
07 MOV AX,@DATA
08 MOV DS,AX
09 ;INPUT
10 ;FAST BX=0
11 XOR BX,BX
12 INPUT_LOOP:
13 ;CHAR INPUT
14 MOV AH,1
15 INT 21H
16 ;IF \N\R, STOP TAKING INPUT
17 CMP AL,10
18 JE END_INPUT_LOOP
19 CMP AL,13
20 JE END_INPUT_LOOP
21 ;FAST CHAR TO DIGIT
22 ;ALSO CLEARS AH
23 AND AX,000FH
24 ;SAVE AX
25 MOV CX,AX
26 ;BX=BX*10+AX
27 MOV AX,10
28 MUL BX
29 ADD AX,CX
30 MOV BX,AX
31 JMP INPUT_LOOP
32 END_INPUT_LOOP:
33 MOV N,BX
34 XOR AX,AX
35 ;SHIFTHING BY 3
36 MOV BX,N
37 MOV CX,3
38 SHR BX,CL
39 ADD AX,BX
40
41 MOV RESULT,AX
42
43 MAIN ENDP
44 END MAIN
45 RET
46

edit: D:\CSE 3-1\Sessional\Microprocessor and Interfacing Sessional\lab 5\PROB3.ASM

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate

35 ;SHIFTHING BY 3
36 MOV BX,N
37 MOV CX,3
38 SHR BX,CL
39 ADD AX,BX
40
41 MOV RESULT,AX
42
43 MAIN ENDP
44 END MAIN
45 RET
46

emulator: PROB3

file math debug

Load

registers

	H	L
AX	00	00
BX	00	00
CX	00	03
DX	00	00
CS	0700	
IP	0150	
SS	0700	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

variables

size: word elements: 1

edit show as: signed

N	104
RESULT	13

emulator screen (80x25 chars)

104

line: 46 col: 1

line: 46

Problem 3:

ORG 0100H

.DATA

N DB 11011011B

RESULT DB ?

.CODE

MAIN PROC

MOV AX,@DATA

MOV DS,AX

MOV AL,N

MOV BL,N

MOV CL,4

CHECK:

ROL AL,1

JC ONEL

MOV AH,0

ZEROL:

ROR BL,1

JC ONER

MOV BH,0

ZEROR:

CMP AH,BH

JNE NOT_PALINDROME

DEC CL

JNZ CHECK

JMP PALINDROME

ONEL:

MOV AH,1

JMP ZEROL

ONER:

MOV BH,1

JMP ZEROR

PALINDROME:

MOV BL,0AAH

JMP END

NOT_PALINDROME:

MOV BL,00H

END:

MAIN ENDP

END MAIN

RET

edit: D:\CSE 3-1\Sessional\Microprocessor and Interfacing Sessional\lab 5\PROE

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calcul

```
01 ORG 0100H
02 .DATA
03 N DB 11011011B
04 RESULT DB ?
05 .CODE
06 MAIN PROC
07     MOV AX,@DATA
08     MOV DS,AX
09     MOV AL,N
10     MOV BL,N
11     MOV CL,4
12 CHECK:
13     ROL AL,1
14     JC ONEL
15     MOV AH,0
16 ZEROL:
17     ROR BL,1
18     JC ONER
19     MOV BH,0
20 ZEROR:
21     CMP AH,BH
22     JNE NOT_PALINDROME
23     DEC CL
24     JNZ CHECK
25     JMP PALINDROME
26 ONEL:
27     MOV AH,1
28     JMP ZEROL
29 ONER:
30     MOV BH,1
31     JMP ZEROR
32 PALINDROME:
33     MOV BL,0AAH
34     JMP END
35 NOT_PALINDROME:
36     MOV BL,00H
37 END:
38     MAIN ENDP
39 END MAIN
```

emulator: PROBL

file math debug

Load re

registers

	H	L
AX	01	BD
BX	01	AA
CX	00	00
DX	00	00
CS	0700	
IP	0149	
SS	0700	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

line: 41 col: 1 dr

Problem 4:

ORG 0100H

.DATA

M DW 'AL = \$'

.CODE

MAIN PROC

MOV AX,@DATA

MOV DS,AX

;MSG

MOV AH,9

LEA DX,M

INT 21H

MOV CL,8

MOV AL,55H

MOV BL,AL

AGAIN:

ROL BL,1

JC ONE

MOV DL,'0'

JMP DISPLAY

ONE:

MOV DL,'1'

DISPLAY:

MOV AH,2

INT 21H

DEC CL

JNZ AGAIN

MAIN ENDP

END MAIN

RET

edit: D:\CSE 3-1\Sessional\Microprocessor and Interfacing Sessional

```
file edit bookmarks assembler emulator math ascii co
new open examples save compile er

01 ORG 0100H
02 .DATA
03 M DW 'AL = $';
04 .CODE
05 MAIN PROC
06     MOV AX,@DATA
07     MOV DS,AX
08
09     ;MSG
10     MOV AH,9
11     LEA DX,M
12     INT 21H
13
14     MOV CL,8
15     MOV AL,55H
16     MOV BL,AL
17 AGAIN:
18     ROL BL,1
19     JC ONE
20     MOV DL,'0'
21     JMP DISPLAY
22 ONE:
23     MOV DL,'1'
24 DISPLAY:
25     MOV AH,2
26     INT 21H
27     DEC CL
28     JNZ AGAIN
29
30     MAIN ENDP
31 END MAIN
32 RET
33
```

emulator: PROB

file math debug

Load

registers

	H	L
AX	02	31
BX	00	55
CX	00	00
DX	01	31
CS	0700	
IP	013F	
SS	0700	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

emulator screen (80x25 chars)

```
AL = 01010101
```

line: 33

Problem 5:

ORG 0100H

.DATA

A DW ?

B DW ?

C DW ?

AM DW 'VALUE OF A: \$'

BM DW 'VALUE OF B: \$'

.CODE

MAIN PROC

MOV AX,@DATA

MOV DS,AX

;A MSG

MOV AH,9

LEA DX,AM

INT 21H

;A INPUT

;FAST BX=0

XOR BX,BX

INPUT_LOOP:

;CHAR INPUT

MOV AH,1

INT 21H

;IF \n\r, STOP TAKING INPUT

CMP AL,10

JE END_INPUT_LOOP

CMP AL,13

JE END_INPUT_LOOP

;FAST CHAR TO DIGIT

;ALSO CLEARS AH

AND AX,000FH

;SAVE AX

MOV CX,AX

;BX=BX*10+AX

MOV AX,10

MUL BX

ADD AX,CX

MOV BX,AX

JMP INPUT_LOOP

END_INPUT_LOOP:

MOV A,BX

;NEW LINE

MOV AH,2

MOV DL,0AH

INT 21H

MOV DL,0DH

INT 21H

;B MSG

MOV AH,9

LEA DX,BM

INT 21H

;B INPUT

;FAST BX=0

XOR BX,BX

INPUT_LOOP2:

;CHAR INPUT

MOV AH,1

INT 21H

;IF \NR, STOP TAKING INPUT

CMP AL,10

JE END_INPUT_LOOP2

CMP AL,13

JE END_INPUT_LOOP2

;FAST CHAR TO DIGIT

;ALSO CLEARS AH

AND AX,000FH

;SAVE AX

MOV CX,AX

;BX=BX*10+AX

MOV AX,10

MUL BX

ADD AX,CX

```
    MOV BX,AX
    JMP INPUT_LOOP2
END_INPUT_LOOP2:
    MOV B,BX

    MOV AX,5
    MOV BX,A
    IMUL BX
    SUB AX,7
    MOV A,AX

    MOV AX,A
    SUB AX,B
    MOV BX,B
    SUB BX,10
    IMUL BX
    MOV B,AX

    MAIN ENDP
END MAIN
RET
```

```

edit: D:\CSE 3-1\Sessional\Microprocessor and Interfacing Sessional\lab:
file edit bookmarks assembler emulator math ascii codes h
new open examples save compile emulate

01 ORG 0100H
02 .DATA
03 A DW ?
04 B DW ?
05 C DW ?
06 AM DW 'VALUE OF A: $'
07 BM DW 'VALUE OF B: $'
08 .CODE
09 MAIN PROC
10     MOV AX,@DATA
11     MOV DS,AX
12     ;A MSG
13     MOV AH,9
14     LEA DX,AM
15     INT 21H
16     ;A INPUT
17     ;FAST BX=0
18     XOR BX,BX
19 INPUT_LOOP:
20     ;CHAR INPUT
21     MOV AH,1
22     INT 21H
23     ;IF \N\R, STOP TAKING INPUT
24     CMP AL,10
25     JE END_INPUT_LOOP
26     CMP AL,13
27     JE END_INPUT_LOOP
28     ;FAST CHAR TO DIGIT
29     ;ALSO CLEARS AH
30     AND AX,000FH
31     ;SAVE AX
32     MOV CX,AX
33     ;BX=BX*10+AX
34     MOV AX,10
35     MUL BX
36     ADD AX,CX
37     MOV BX,AX
38     JMP INPUT_LOOP
39 END_INPUT_LOOP:
line: 93 col: 1

```

```

edit: D:\CSE 3-1\Sessional\Microprocessor and Interfacing Sessional\lab:
file edit bookmarks assembler emulator math ascii codes h
new open examples save compile emulate

39 END_INPUT_LOOP:
40     MOV A,BX
41     ;NEW LINE
42     MOV AH,2
43     MOV DL,0AH
44     INT 21H
45     MOV DL,0DH
46     INT 21H
47     ;B MSG
48     MOV AH,9
49     LEA DX,BM
50     INT 21H
51     ;B INPUT
52     ;FAST BX=0
53     XOR BX,BX
54 INPUT_LOOP2:
55     ;CHAR INPUT
56     MOV AH,1
57     INT 21H
58     ;IF \N\R, STOP TAKING INPUT
59     CMP AL,10
60     JE END_INPUT_LOOP2
61     CMP AL,13
62     JE END_INPUT_LOOP2
63     ;FAST CHAR TO DIGIT
64     ;ALSO CLEARS AH
65     AND AX,000FH
66     ;SAVE AX
67     MOV CX,AX
68     ;BX=BX*10+AX
69     MOV AX,10
70     MUL BX
71     ADD AX,CX
72     MOV BX,AX
73     JMP INPUT_LOOP2
74 END_INPUT_LOOP2:
75     MOV B,BX
76
77     MOV AX,5
line: 93 col: 1

```

```

edit: D:\CSE 3-1\Sessional\Microprocessor and Interfacing Sessional\lab 5\PROG
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calc

76
77     MOV AX,5
78     MOV BX,A
79     IMUL BX
80     SUB AX,7
81     MOV A,AX
82
83     MOV AX,A
84     SUB AX,B
85     MOV BX,B
86     SUB BX,10
87     IMUL BX
88     MOV B,AX
89
90     MAIN ENDP
91 END MAIN
92 RET
93
variables
size: word elements: 1
edit show as: signed
A 18
B 15
C 0
AM 4156h
BM 4156h

emulator: PROBL
file math debug
Load
registers
AX 00 0F
BX 00 03
CX 00 03
DX 00 00
CS 0700
IP 01BA
SS 0700
SP FFFE
BP 0000
SI 0000
DI 0000
DS 0700
ES 0700

emulator screen (80x25 chars)
VALUE OF A: 5
VALUE OF B: 13
line: 93

```

Discussion:

Finally it can be said that, we learned about so many new instruction called Logic, Shift and Rotate Instructions & Multiplication and Division Instructions. By using those instruction, we are able solve new logical problem like Palindrome Check or Byte Display etc.