 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Department of Computer Engineering</b>	
<b>Subject: Fundamental of Processors (01CE0509)</b>	<b>Aim: To perform Shift and Rotate Instructions in 8086 Microprocessor.</b>	
<b>Experiment No: 06</b>	<b>Date:</b>	<b>Enrolment No:92201703058</b>

**Aim: To perform Shift and Rotate Instructions in 8086 Microprocessor.**

**Apparatus :**Computer System.

**Theory:**Any machine (system) works on machine language, which consists of binary numbers. In the 8086 microprocessor, we have 16-bit registers to handle our data. Sometimes, the need to perform some necessary **shift and rotate operations** on our data may occur according to the given condition and requirement. So, for that purpose, we have various Shift and Rotate instructions present in the 8086 microprocessor. Let us discuss them one by one and understand their working:

1. SHR : Shift Right
2. SAR : Shift Arithmetic Right
3. SHL : Shift Left
4. SAL : Shift Arithmetic Left
5. ROL : Rotate Left
6. ROR : Rotate Right
7. RCL : Rotate Carry Left
8. RCR : Rotate Carry Right


### 1) SHR : Shift Right

The SHR instruction is an abbreviation for 'Shift Right'. This instruction simply shifts the mentioned bits in the register to the right side one by one by inserting the same number (bits that are being shifted) of zeroes from the left end. The rightmost bit that is being shifted is stored in the Carry Flag (CF).

Syntax: SHR Register, Bits to be shifted

Example: SHR AX, 2



 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Department of Computer Engineering</b>	
<b>Subject: Fundamental of Processors (01CE0509)</b>	<b>Aim: To perform Shift and Rotate Instructions in 8086 Microprocessor.</b>	
<b>Experiment No: 06</b>	<b>Date:</b>	<b>Enrolment No:92201703058</b>

## 2) SAR : Shift Arithmetic Right

The SAR instruction stands for 'Shift Arithmetic Right'. This instruction shifts the mentioned bits in the register to the right side one by one, but instead of inserting the zeroes from the left end, the MSB is restored. The rightmost bit that is being shifted is stored in the Carry Flag (CF).

Syntax: SAR Register, Bits to be shifted

Example: SAR BX, 5




## 3) SHL : Shift Left

The SHL instruction is an abbreviation for 'Shift Left'. This instruction simply shifts the mentioned bits in the register to the left side one by one by inserting the same number (bits that are being shifted) of zeroes from the right end. The leftmost bit that is being shifted is stored in the Carry Flag (CF).

Syntax: SHL Register, Bits to be shifted

Example: SHL AX, 2



 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Department of Computer Engineering</b>	
<b>Subject: Fundamental of Processors (01CE0509)</b>	<b>Aim: To perform Shift and Rotate Instructions in 8086 Microprocessor.</b>	
<b>Experiment No: 06</b>	<b>Date:</b>	<b>Enrolment No:92201703058</b>

#### 4) SAL : Shift Arithmetic Left

The SAL instruction is an abbreviation for 'Shift Arithmetic Left'. This instruction is the same as SHL.

Syntax: SAL Register, Bits to be shifted

Example: SAL CL, 2




#### 5) ROL : Rotate Left

The ROL instruction is an abbreviation for 'Rotate Left'. This instruction rotates the mentioned bits in the register to the left side one by one such that leftmost bit that is being rotated is again stored as the rightmost bit in the register, and it is also stored in the Carry Flag (CF).

Syntax: ROL Register, Bits to be shifted

Example: ROL AH, 4



 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Department of Computer Engineering</b>	
<b>Subject: Fundamental of Processors (01CE0509)</b>	<b>Aim: To perform Shift and Rotate Instructions in 8086 Microprocessor.</b>	
<b>Experiment No: 06</b>	<b>Date:</b>	<b>Enrolment No:92201703058</b>

## 6) ROR : Rotate Right

The ROR instruction stands for 'Rotate Right'. This instruction rotates the mentioned bits in the register to the right side one by one such that rightmost bit that is being rotated is again stored as the MSB in the register, and it is also stored in the Carry Flag (CF).

Syntax: ROR Register, Bits to be shifted

Example: ROR AH, 4




## 7) RCL : Rotate Carry Left

This instruction rotates the mentioned bits in the register to the left side one by one such that leftmost bit that is being rotated it is stored in the Carry Flag (CF), and the bit in the CF moved as the LSB in the register.

Syntax: RCL Register, Bits to be shifted

Example: RCL CH, 1



 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Department of Computer Engineering</b>	
<b>Subject: Fundamental of Processors (01CE0509)</b>	<b>Aim: To perform Shift and Rotate Instructions in 8086 Microprocessor.</b>	
<b>Experiment No: 06</b>	<b>Date:</b>	<b>Enrolment No:92201703058</b>

## 8) RCR : Rotate Carry Right


This instruction rotates the mentioned bits in the register to the right side such that rightmost bit that is being rotated it is stored in the Carry Flag (CF), and the bit in the CF moved as the MSB in the register.

Syntax: RCR Register, Bits to be shifted

Example: RCR BH, 6



**Program:**

 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Department of Computer Engineering</b>	
<b>Subject: Fundamental of Processors (01CE0509)</b>	<b>Aim: To perform Shift and Rotate Instructions in 8086 Microprocessor.</b>	
<b>Experiment No: 06</b>	<b>Date:</b>	<b>Enrolment No:92201703058</b>

### Emulate the code:

edit: D:\emu8086\MySource\Practical\_6.asm

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options

emulator: Practical\_6.com

file math debug view external virtual devices virtual drive help

Load reload step back single step run step delay ms: 0

registers	H	L
AX	00	00
BX	00	00
CX	00	30
DX	00	00
CS	0700	
IP	0100	
SS	0700	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

0700:0100	0700:0100
07100: EB 235 6	JMP 0103h
07101: 01 001 0	RETF 0C88Ch
07102: CA 202 11	MOV DS, AX
07103: 8C 140 1	MOV BL, [00102h]
07104: C8 200 11	SHL BL, 1
07105: 8E 142 11	SAR BL, 1
07106: D8 216 11	SAR BL, 1
07107: 8A 138 11	SAR BL, 1
07108: 1E 030 11	SHR BL, 1
07109: 02 002 11	SHR BL, 1
0710A: 01 001 11	SHR BL, 1
0710B: D0 208 11	SHL BL, 1
0710C: E3 227 11	SHL BL, 1
0710D: D0 208 11	ROR BL, 1
0710E: FB 251 11	ROR BL, 1
0710F: D0 208 11	ROR BL, 1
07110: FB 251 11	RCR BL, 1
07111: D0 208 11	RCR BL, 1
07112: FB 251 11	ROL BL, 1
07113: D0 208 11	ROL BL, 1
07114: EB 235 6	RCL BL, 1
07115: D0 208 11	...


original source c...

```

01 ; NAME: ASIF ALAM
02 ; ENROLLMENT: 92201703058
03
04
05 ORG 100H
06 LD DATA
07 A DB 11001010B
08
09 .CODE
10
11 MOV AX, EDATA
12 MOV DS, AX
13
14 MOV BL, A
15
16 SHL BL, 1
17 SAR BL, 3
18
19 CUD DI ?

```

screen source reset aux vars debug stack flags

 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Department of Computer Engineering</b>	
<b>Subject: Fundamental of Processors (01CE0509)</b>	<b>Aim: To perform Shift and Rotate Instructions in 8086 Microprocessor.</b>	
<b>Experiment No: 06</b>	<b>Date:</b>	<b>Enrolment No: 92201703171</b>

**Check the value of Bx register and verify the Bx register data step by step:**

### Step01: MOV BL,A

extended value viewer

watch: **BX**

☒ word
☐ byte

	H	L
hex:	00	CA
bin:	00000000	11001010
oct:	000	312

decimal 8 bit

unsigned:	0	202
signed:	0	-54
ascii:		ü

decimal 16 bit

unsigned:	202
signed:	202

flags

CF	0
ZF	0
SF	0
OF	0
PF	0
AF	0
IF	1
DF	0

analyse

### Step02: SHL BL,1

extended value viewer

watch: **BX**

☒ word
☐ byte

	H	L
hex:	00	94
bin:	00000000	10010100
oct:	000	224

decimal 8 bit

unsigned:	0	148
signed:	0	-108
ascii:		ö


decimal 16 bit

unsigned:	148
signed:	148

flags

CF	1
ZF	0
SF	1
OF	0
PF	0
AF	0
IF	1
DF	0

analyse

 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Department of Computer Engineering</b>	
<b>Subject: Fundamental of Processors (01CE0509)</b>	<b>Aim: To perform Shift and Rotate Instructions in 8086 Microprocessor.</b>	
<b>Experiment No: 06</b>	<b>Date:</b>	<b>Enrolment No:92201703058</b>

### Step03: SAR BL,3

extended value viewer

watch: **BX**

☒ word
☐ byte

	H	L
hex:	00	94
bin:	00000000	10010100
oct:	000	224

decimal 8 bit

unsigned:	0	148
signed:	0	-108
ascii:		ö

decimal 16 bit

unsigned:	148
signed:	148

flags

CF	1
ZF	0
SF	1
OF	0
PF	0
AF	0
IF	1
DF	0

analyse

### Step04: SHR BL,3

extended value viewer

watch: **BX**

☒ word
☐ byte

	H	L
hex:	00	1E
bin:	00000000	00011110
oct:	000	036

decimal 8 bit

unsigned:	0	30
signed:	0	30
ascii:		▲

decimal 16 bit


unsigned:	30
signed:	30

flags

CF	0
ZF	0
SF	0
OF	0
PF	1
AF	0
IF	1
DF	0

analyse



 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Department of Computer Engineering</b>	
<b>Subject: Fundamental of Processors (01CE0509)</b>	<b>Aim: To perform Shift and Rotate Instructions in 8086 Microprocessor.</b>	
<b>Experiment No: 06</b>	<b>Date:</b>	<b>Enrolment No:92201703058</b>

### Step05: SAL BL,2

extended value viewer

watch: **BX**

☒ word
☐ byte

	H	L
hex:	00	78
bin:	00000000	01111000
oct:	000	170

decimal 8 bit

unsigned:	0	120
signed:	0	120
ascii:		x

decimal 16 bit

unsigned:	120
signed:	120

flags

CF	0
ZF	0
SF	0
OF	0
PF	1
AF	0
IF	1
DF	0

analyse

### Step06: ROR BL,3

extended value viewer

watch: **BX**

☒ word
☐ byte

	H	L
hex:	00	0F
bin:	00000000	00001111
oct:	000	017

decimal 8 bit

unsigned:	0	15
signed:	0	15
ascii:		*


decimal 16 bit

unsigned:	15
signed:	15

flags

CF	0
ZF	0
SF	0
OF	0
PF	1
AF	0
IF	1
DF	0

analyse

 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Department of Computer Engineering</b>	
<b>Subject: Fundamental of Processors (01CE0509)</b>	<b>Aim: To perform Shift and Rotate Instructions in 8086 Microprocessor.</b>	
<b>Experiment No: 06</b>	<b>Date:</b>	<b>Enrolment No:92201703058</b>

### Step07: RCR BL,2

extended value viewer

watch: **BX**

word
byte

	H	L
hex:	00	83
bin:	00000000	10000011
oct:	000	203

decimal 8 bit

unsigned:	0	131
signed:	0	-125
ascii:		â

decimal 16 bit

unsigned:	131
signed:	131

flags

CF	1
ZF	0
SF	0
OF	1
PF	1
AF	0
IF	1
DF	0

analyse

### Step08: ROL BL,2

extended value viewer

watch: **BX**

word
byte

	H	L
hex:	00	0E
bin:	00000000	00001110
oct:	000	016

decimal 8 bit

unsigned:	0	14
signed:	0	14
ascii:		Ń


decimal 16 bit

unsigned:	14
signed:	14

flags

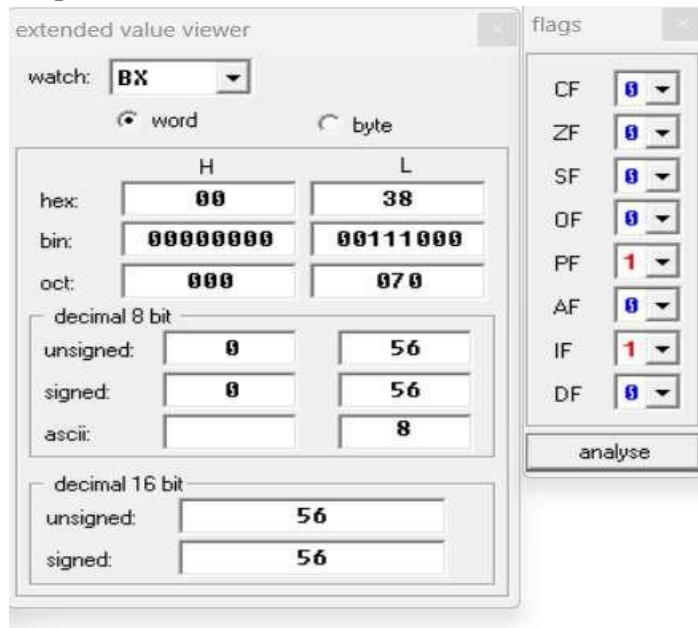
CF	0
ZF	0
SF	0
OF	0
PF	1
AF	0
IF	1
DF	0

analyse

 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Department of Computer Engineering</b>	
<b>Subject: Fundamental of Processors (01CE0509)</b>	<b>Aim: To perform Shift and Rotate Instructions in 8086 Microprocessor.</b>	
<b>Experiment No: 06</b>	<b>Date:</b>	<b>Enrolment No:92201703058</b>

**Check the Bx Register value after all steps and check the output:**

**Step09: RCL BL,2**



The screenshot shows two windows from a debugger. The 'extended value viewer' window displays the contents of the BX register, which is set to 'word' mode. It shows the high byte (H) as 00 and the low byte (L) as 38. Below this, it provides binary, octal, and decimal representations for both bytes. The decimal 16-bit value is shown as 56. The 'flags' window shows the status of various flags: CF (0), ZF (0), SF (0), OF (0), PF (1), AF (0), IF (1), and DF (0). An 'analyse' button is located at the bottom of the flags window.

extended value viewer	
watch: BX	
word	byte
hex: H: 00, L: 38	
bin: 00000000, 00111000	
oct: 000, 070	
decimal 8 bit	
unsigned: 0, 56	
signed: 0, 56	
ascii: , 8	
decimal 16 bit	
unsigned: 56	
signed: 56	

flags	
CF	0
ZF	0
SF	0
OF	0
PF	1
AF	0
IF	1
DF	0
analyse	

**Conclusion:**