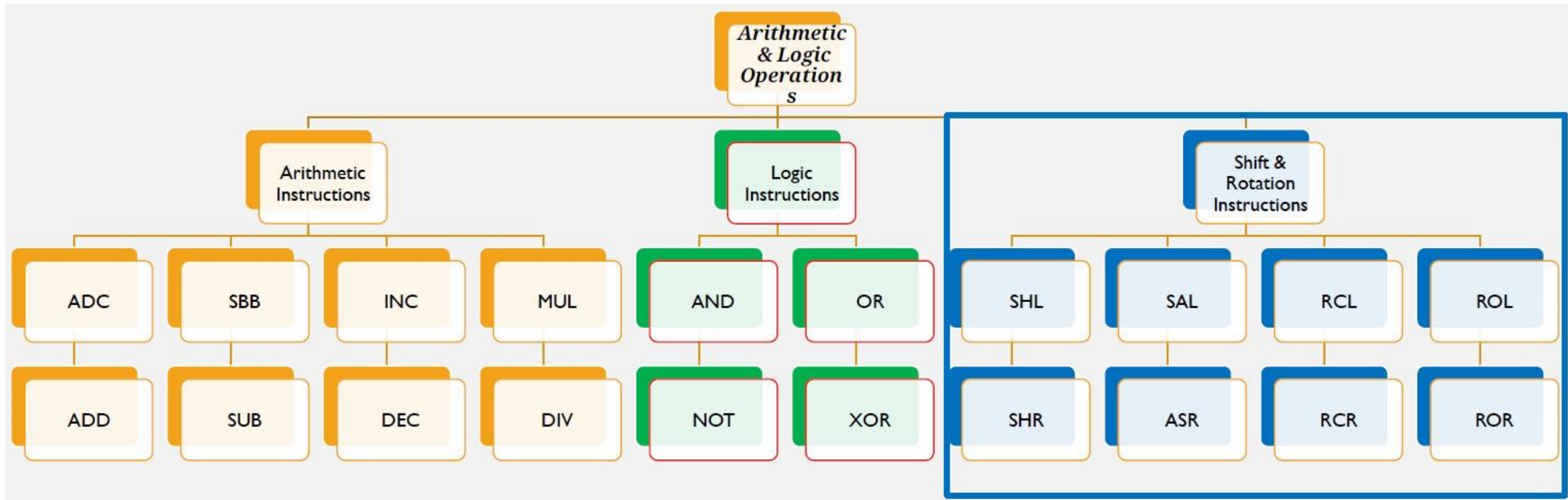


Shift & Rotation Instruction

REVIEW OF ASSEMBLY LANGUAGE

ARITHMETIC & LOGIC OPERATIONS



Introduction

- Shift and rotate instructions manipulate binary numbers at the binary bit level, as did the AND, OR, Exclusive-OR, and NOT instructions.
- The microprocessor contains a complete complement of shift and rotate instructions that are used to shift or rotate any memory data or register.
- Shift Instructions in assembly language is used to shift the most significant bit (MSB) or the least significant bit (LSB) in binary number (in a register or a variable).

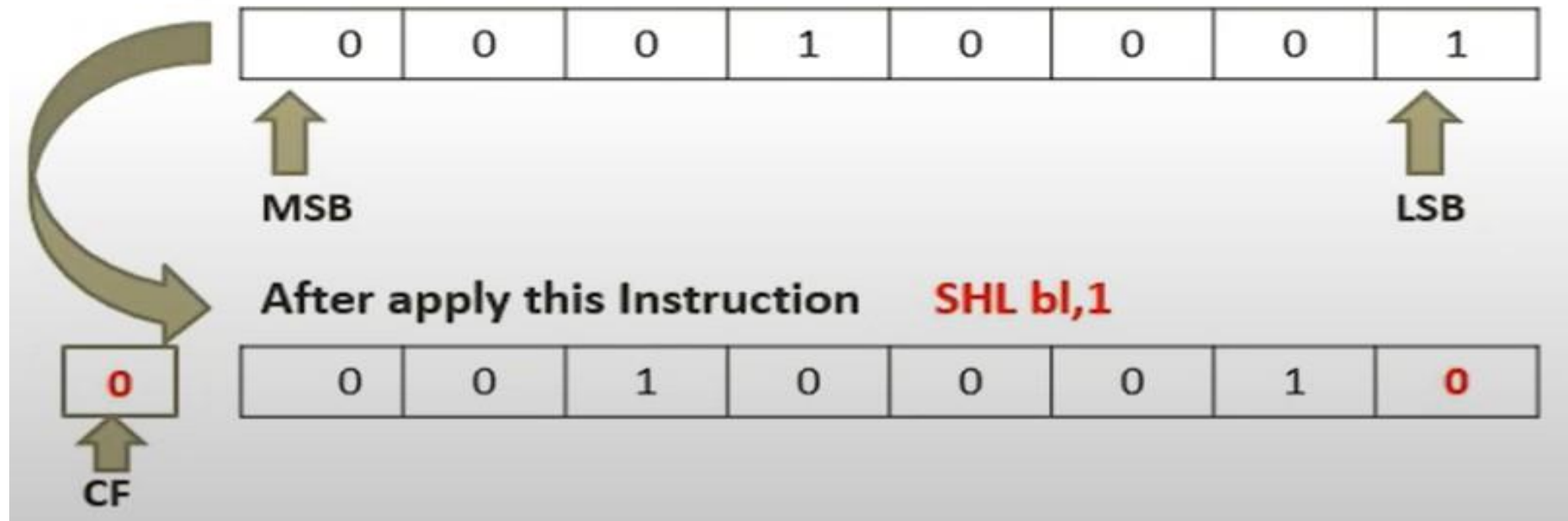
Shift Instructions

- In this section of course, we will deal with four type of Shift Instruction ,don't forget Shift Instructions deal with number by converting to binary.

1. SHL : (Shift Left Instruction) it works as add zero to LSB(Last Significant bit) and MSB (Most significant bit) will move to CF.

you can understand as followed MOV left bit in CF and add 0 to right

For Example let Deal with byte (MOV BL , 17d)



8 bit 16 bit

hex: 20

signed unsigned

dec: 32 32

ascii char: ☐

oct: 40

bin: 00100000

TRANSITIONS ANIMATIONS SLIDE SHOW

Calibri (Body) 18 A A A

B I U S abc AV Aa A

Font

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options help about

```

01
02 ; You may customize this and other star
03 ; The location of this template is c:\e
04
05 org 100h
06
07 mov bl,00001111b
08 shl bl,1
09 ;0001 1110
10
11 ret
12
13
14
15
16

```

original source co...

```

01
02 ; You may customize this
03 ; The location of this t
04
05 org 100h
06
07 mov bl,00001111b
08 shl bl,1
09 ;0001 1110
10
11 ret
12
13
14
15
16
17
18

```

emulator: noname.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	1E
CX	00	05
DX	00	00
CS	F400	
IP	0154	
SS	0700	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

message

PROGRAM HAS RETURNED CONTROL TO THE OPERATING SYSTEM

OK

F415F: 00 000 NULL ADD [BX + SI], AL
 F4160: FF 255 RES ADD BH, BH
 F4161: FF 255 RES DEC BP
 F4162: CD 205 = ADD BH, CL
 F4163: 1A 026 → ADD [BX + SI], AL
 F4164: CF 207 ± ADD [BX + SI], AL
 F4165: 00 000 NULL ...

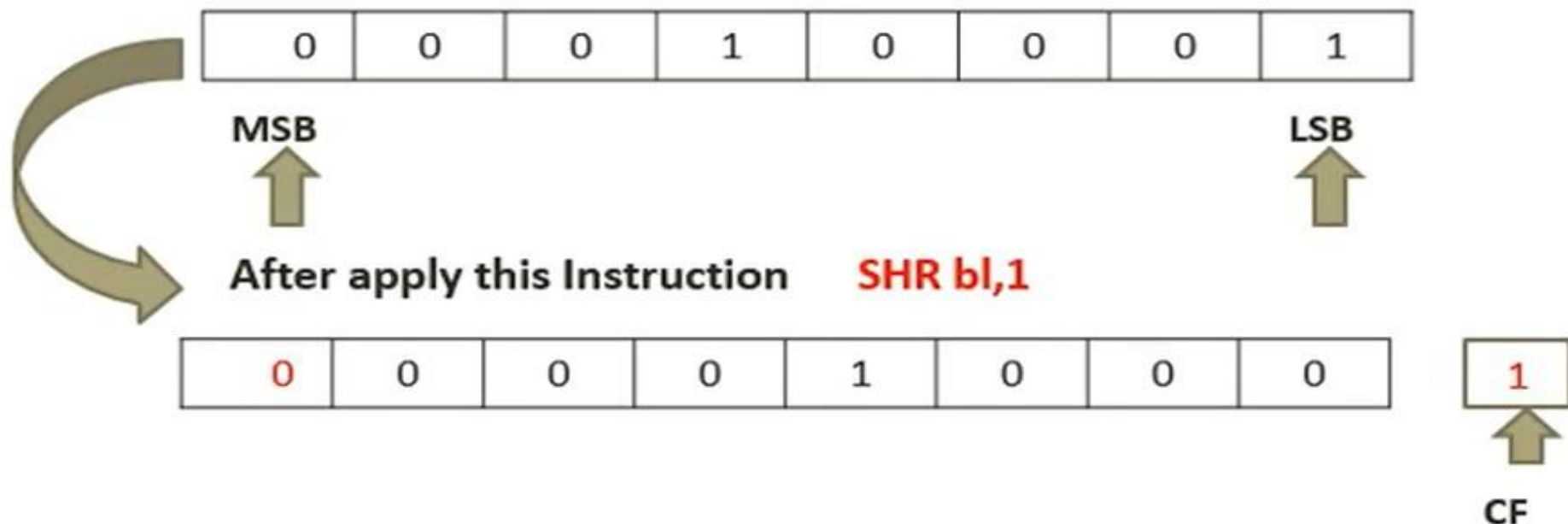
screen source reset aux vars debug stack flags

Shift Instructions

2. SHR : (Shift Right Instruction) it works as add zero to MSB (Most Significant bit) and LSB (Least significant bit) will move to CF.

you can understand as followed MOV left bit in CF and add 0 to right

For Example let Deal with byte (Mov BL, 17d)



8 bit 16 bit

hex: 3C

signed unsigned

dec: 60 60

ascii char: <

oct: 74

bin: 00111100

TRANSITIONS ANIMATIONS SLIDE SHOW

Calibri (Body) 18 A A

B I U S abc AV Aa A

Font

Shift Instru

2. SHR : Signifi

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options help about

```
01 ; You may customize this and other star
02 ; The location of this template is c:\e
03
04
05 org 100h
06
07 mov bl,00001111b
08 shr bl,1
09 ;0000 0111
10 ;shr bl,1
11 ;0011 1100
12 ret
13
14
15
```

original source co...

```
01
02 ; You may customize this
03 ; The location of this t
04
05 org 100h
06
07 mov bl,00001111b
08 shr bl,1
09 ;0000 0111
10 ;shr bl,1
11 ;0011 1100
12 ret
13
14
15
16
17
18
19
```

flags

CF 1

ZF 0

SF 0

OF 0

PF 0

AF 0

IF 0

DF 0

analyse

emulator: noname.com_

file math debug view external virtual devices

Load reload step back single

registers

	H	L
AX	00	00
BX	00	07
CX	00	05
DX	00	00
CS	F400	
IP	0154	
SS	0700	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

F400:0154

message

PROGRAM HAS RETURNED CC
TO THE OPERATING SYSTEM

OK

F415F: 00 000 NULL
F4160: FF 255 RES
F4161: FF 255 RES
F4162: CD 205 =
F4163: 1A 026 +
F4164: CF 207 ±
F4165: 00 000 NULL

ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADD BH, CL
ADD [BX + SI], AL
ADD [BX + SI], AL

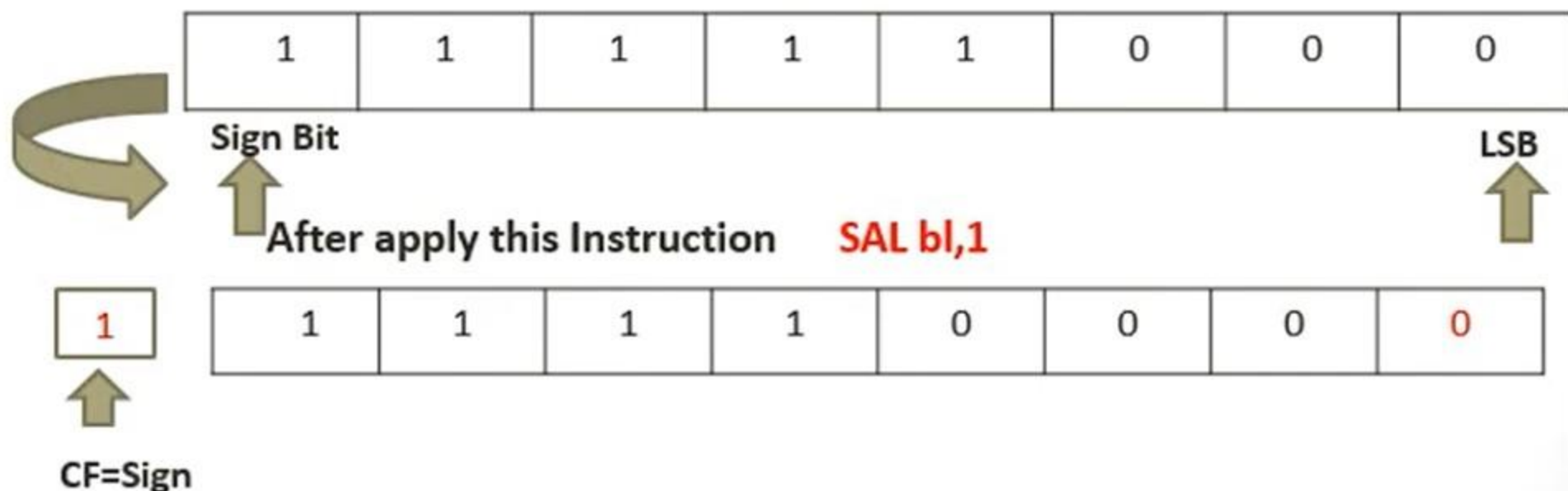
- 12 Rotate & Shift Instructions
- 13 Introduction
- 14 Shift Instructions
- 15 Shift Instructions
- 16 Shift Instructions

Shift Instructions

3. SAL : (Shift Arithmetic Left) it works as add zero to LSB (Last Significant bit) and MSB (Most significant bit) will move to CF. No change in sign of number.

you can understand as followed MOV left bit in CF and add 0 to right

For Example let Deal with byte (Mov BL, -8d)



8 bit 16 bit

hex:

signed unsigned

dec:

ascii char:

oct:

bin:

TRANSITIONS ANIMATIONS SLIDE SHOW

Font

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options help about

```

01
02 ; You may customize this and other star
03 ; The location of this template is c:\e
04
05 org 100h
06
07 mov bl,-6d
08 sal bl,1
09 ;11111010
10 ;11110100
11 ret
12
13
14
15

```

original source co...

```

01
02 ; You may customize this
03 ; The location of this t
04
05 org 100h
06
07 mov bl,-6d
08 sal bl,1
09 ;11111010
10 ;11110100
11 ret
12
13
14
15
16
17
18

```

Shift Instru

3. SAL : (Signifi

flags

CF

ZF

SF

OF

PF

AF

IF

DF

analyse

emulator: noname.com_

file math debug view external virtual devices

Load reload step back single

step delay ms: 0

registers

	H	L
AX	00	00
BX	00	F4
CX	00	05
DX	00	00
CS	F400	
IP	0154	
SS	0700	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

message

PROGRAM HAS RETURNED CC TO THE OPERATING SYSTEM

OK

F415F: 00 000 NULL ADD [BX + SI], AL

F4160: FF 255 RES ADD BH, BH

F4161: FF 255 RES DEC BP

F4162: CD 205 = ADD BH, CL

F4163: 1A 026 → ADD [BX + SI], AL

F4164: CF 207 ± ADD [BX + SI], AL

F4165: 00 000 NULL ...

screen source reset aux vars debug stack flags

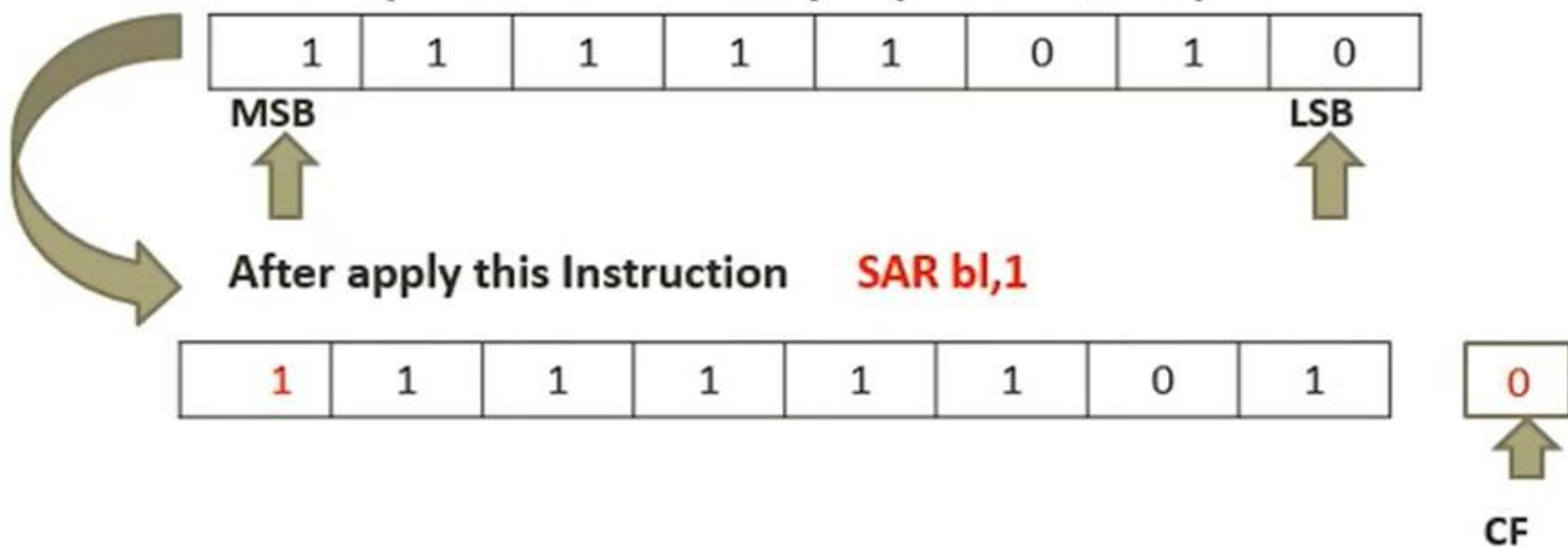
- Shift Instructions
- Shift Instructions
- Shift Instructions
- Shift Instructions
- Shift Instructions
- Applying Shift Instructions in Emulator

Shift Instructions

4. SAR : (Shift Arithmetic Right) it works same as SHR, except that MSB (Most Significant bit) is filled with a copy of the original MSB and LSB (Least significant bit) will move to CF.

you can understand as followed **MOV left bit in CF and add 0 to right**

For Example let Deal with byte (Mov BL, -6d)



8 bit 16 bit

hex:

signed unsigned

dec:

ascii char: ☐

oct:

bin:

TRANSITIONS ANIMATIONS SLIDE SHOW

Calibri (Body) 18 A A

B I U S abc AV Aa A

Font

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options help about

```

01
02 ; You may customize this and other star
03 ; The location of this template is c:\e
04
05 org 100h
06
07 mov bl, -9d
08 sar bl, 1
09 ;11110111
10 ;11110111
11 ret
12
13
14
15

```

original source co...

```

01
02 ; You may customize this
03 ; The location of this t
04
05 org 100h
06
07 mov bl, -9d
08 sar bl, 1
09 ;11110111
10 ;11110111
11 ret
12
13
14
15
16
17
18

```

Shift Instru

4. SAR : (S

flags

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

analyse

emulator: noname.com_

file math debug view external virtual devices

Load reload step back single

registers

	H	L
AX	00	00
BX	00	FB
CX	00	05
DX	00	00
CS	F400	
IP	0154	
SS	0700	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

F400:0154

message

PROGRAM HAS RETURNED CC TO THE OPERATING SYSTEM

OK

F415F: 00 000 NULL ADD [BX + SI], AL

F4160: FF 255 RES ADD BH, BH

F4161: FF 255 RES DEC BP

F4162: CD 205 = ADD BH, CL

F4163: 1A 026 → ADD [BX + SI], AL

F4164: CF 207 ± ADD [BX + SI], AL

F4165: 00 000 NULL ...

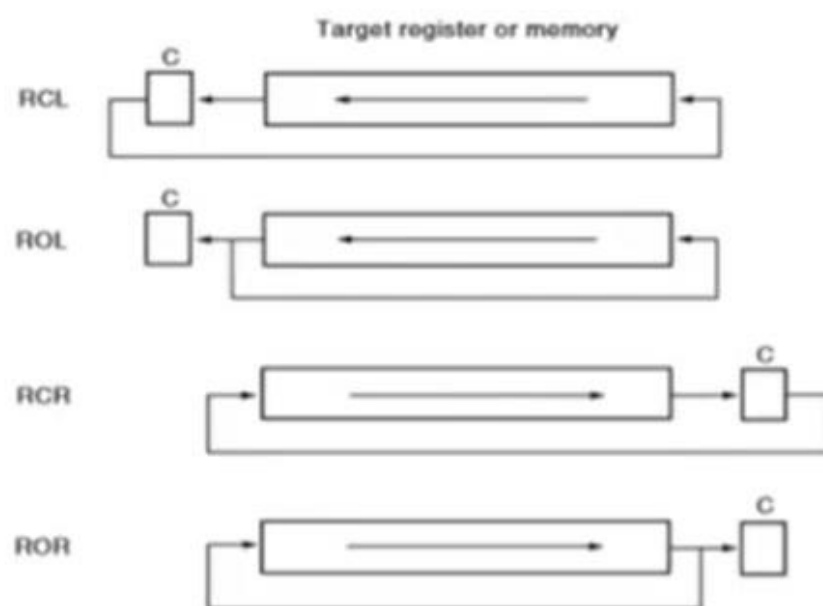
screen source reset aux vars debug stack flags

- Shift Instructions
- Shift Instructions
- Shift Instructions
- Shift Instructions
- Applying Shift Instructions in Emulator

Rotate Instructions

- There are 4 Rotate Instructions
 1. ROL(Rotate Left)
 2. RCL (Rotate cycle left)
 3. ROR (Rotate right)
 4. RCR (Rotate Cycle Right)

FIGURE 5-10 The rotate instructions showing the direction and operation of each rotate.

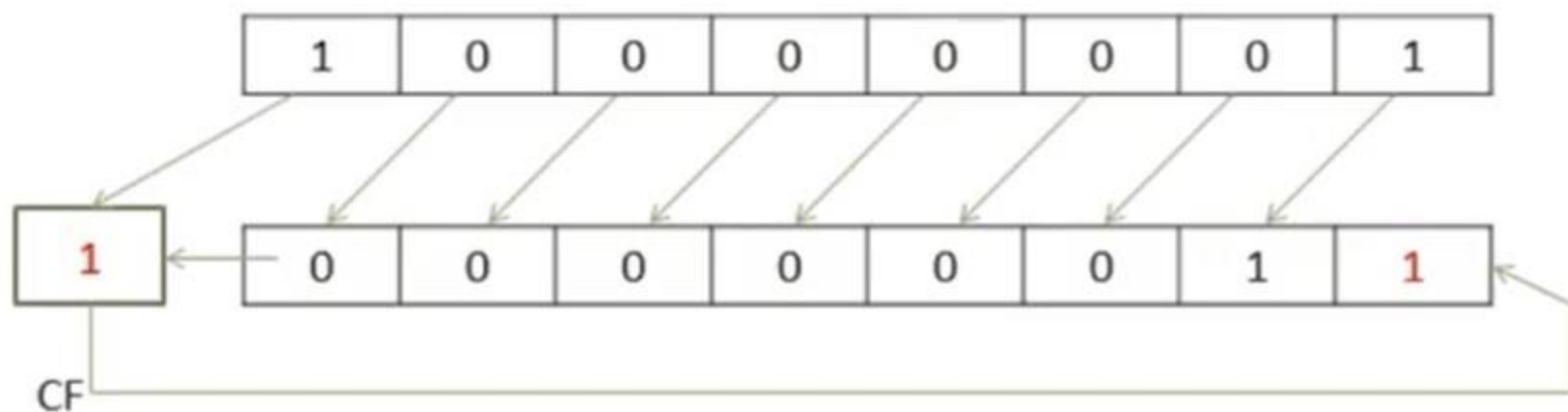


Rotate Instructions

- ROL I

MOV BL,81H

ROL BL,1;After this instruction BL = 03H



8 bit 16 bit

hex: 1E

signed unsigned

dec: 30 30

ascii char: ▲

oct: 36

bin: 00011110

TRANSITIONS ANIMATIONS SLIDE SHOW

Calibri (Body) 18 A* A* A

B I U S abc AV Aa A

Font

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options help about

Rotate Inst

• ROL

```

01
02 ; You may customize this and other star
03 ; The location of this template is c:\e
04
05 org 100h
06
07 mov bl,81h
08 ROL bl,1
09 mov al,00001111b
10 rol al,1
11 ; 0001 1110
12 rol al,1
13 ; 0011 1100
14 ret

```

original source co...

```

01
02 ; You may customize this
03 ; The location of this t
04
05 org 100h
06
07 mov bl,81h
08 ROL bl,1
09 mov al,00001111b
10 rol al,1
11 ; 0001 1110
12 rol al,1
13 ; 0011 1100
14 ret

```

flags

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

analyse

emulator: noname.com_

file math debug view external virtual devices

Load reload step back single

registers

	H	L
AX	00	3C
BX	00	03
CX	00	0B
DX	00	00
CS	F400	
IP	0154	
SS	0700	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

F400:0154

message

PROGRAM HAS RETURNED CC TO THE OPERATING SYSTEM

OK

F415F: 00 000 NULL
F4160: FF 255 RES
F4161: FF 255 RES
F4162: CD 205 =
F4163: 1A 026 →
F4164: CF 207 ±
F4165: 00 000 NULL

ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADD BH, CL
ADD [BX + SI], AL
ADD [BX + SI], AL

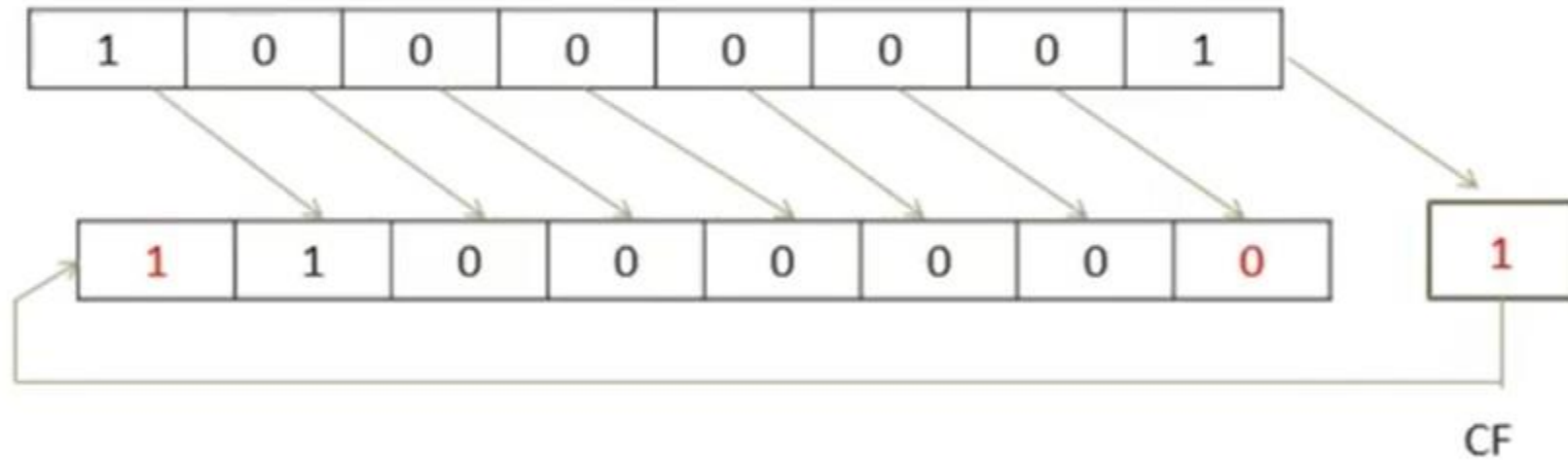
- 20 Applying Shift Instructions to Emulator
- 21 Rotate Instructions
- 22 Rotate Instructions
- 23 Rotate Instructions
- 24 Rotate Instructions

Rotate Instructions

- ROR

MOV BL,81D

ROR BL,1 ;After this instruction BL = C0 H



8 bit16 bit

hex: C3

signedunsigned

dec: -61195

ascii char:

oct: 303

bin: 11000011

TRANSITIONS

ANIMATIONS

SLIDE SHOW

Font

Rotate Inst

ROR

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options help about

01

02 ; You may customize this and other star

03 ; The location of this template is c:\e

04

05 org 100h

06

07 mov bl,81h

08 ROR bl,1

09 ;10000001

10 ;11000000

11 mov al,00001111b

12 ror al,1

13 ; 10000111

14 ror al,1

15 ; 11000011

16 ret

17

18

19

original source co...

01

02 ; You may customize this

03 ; The location of this t

04

05 org 100h

06

07 mov bl,81h

08 ROR bl,1

09 ;10000001

10 ;11000000

11 mov al,00001111b

12 ror al,1

13 ; 10000111

14 ror al,1

15 ; 11000011

16 ret

17

18

19

emulator: noname.com_

file math debug view external virtual devices

Load reload step back single

registers

	H	L
AX	00	C3
BX	00	C0
CX	00	0B
DX	00	00
CS	F400	
IP	0154	
SS	0700	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

message

PROGRAM HAS RETURNED CC TO THE OPERATING SYSTEM

OK

F415F: 00 000 NULL

F4160: FF 255 RES

F4161: FF 255 RES

F4162: CD 205 =

F4163: 1A 026 →

F4164: CF 207 ±

F4165: 00 000 NULL

ADD [BX + SI], AL

ADD BH, BH

DEC BP

ADD BH, CL

ADD [BX + SI], AL

ADD [BX + SI], AL

screen source reset aux vars debug stack flags

drag a file here to open

SLIDE 23 OF 25

ENG

2021/4/27

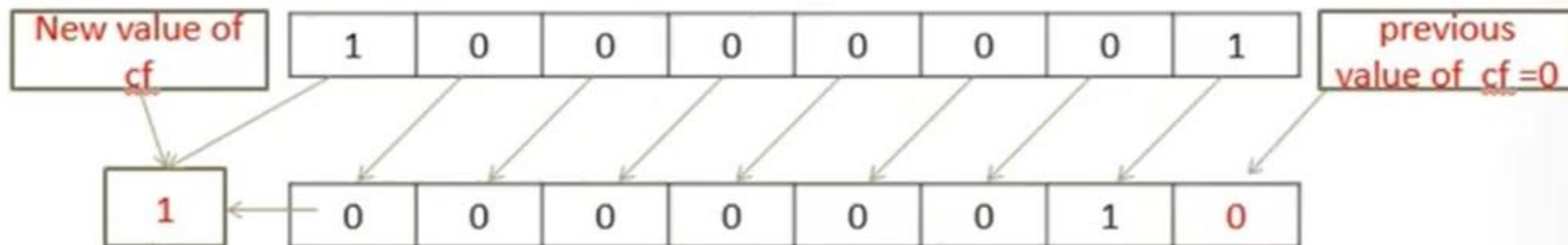
اكتب هنا للبحث

Rotate Instructions

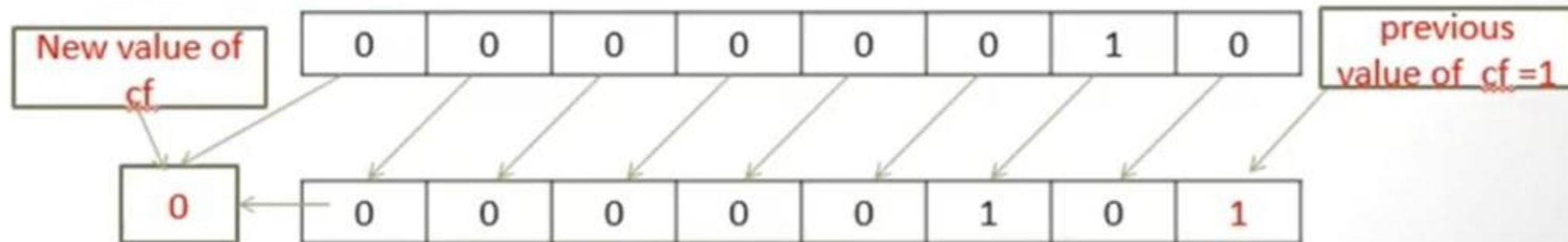
- RCL

MOV BL,81H ;

RCL BL,1 ; After this instruction BL = 02H



RCL BL,1 ; After RCL bl=05h

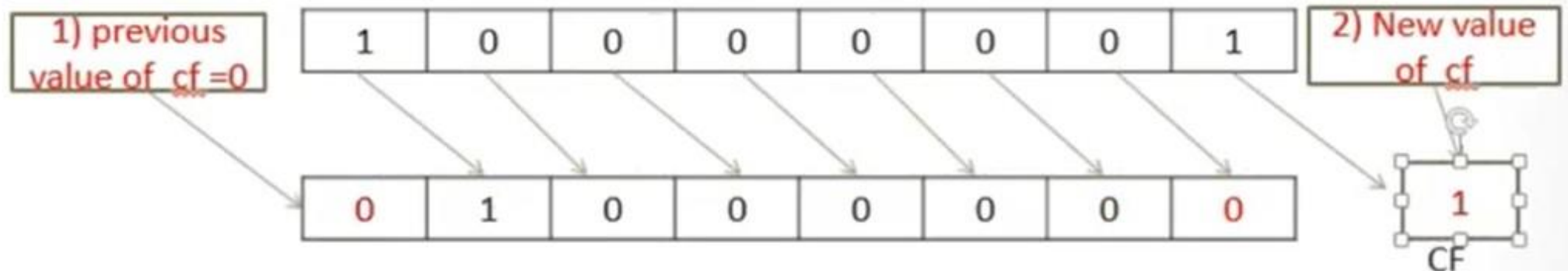


Rotate Instructions

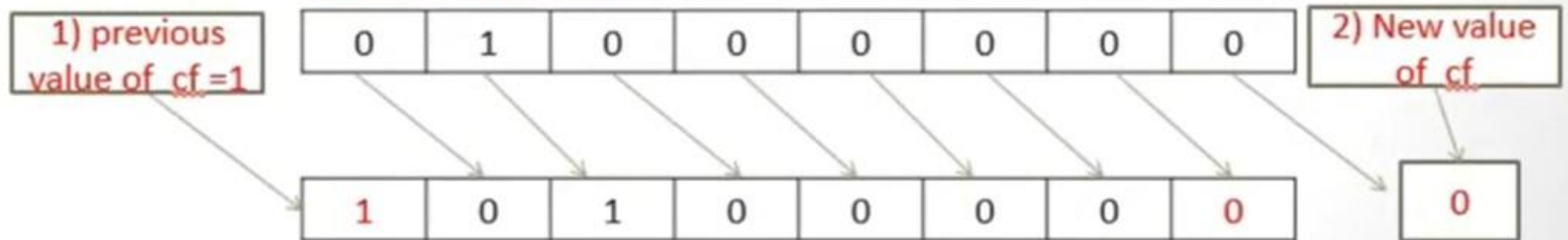
- RCR (Followed instruction are complete one program)

MOV BL,81H ;

RCR BL,1 ;After this instruction BL = 40H



RCR BL,1 ;After RCR bl=A0h



Example

Ex: Answer the following:

1. What does this program do?
2. What is the final result stored in BL register?

```
SUB BL,BL
MOV DL,8
MOV al, DATA1
AGAIN:
ROL al,1
JNC NEXT
INC BL
NEXT: DEC DL
JNZ AGAIN
HLT
DATA1 DB 97
```

