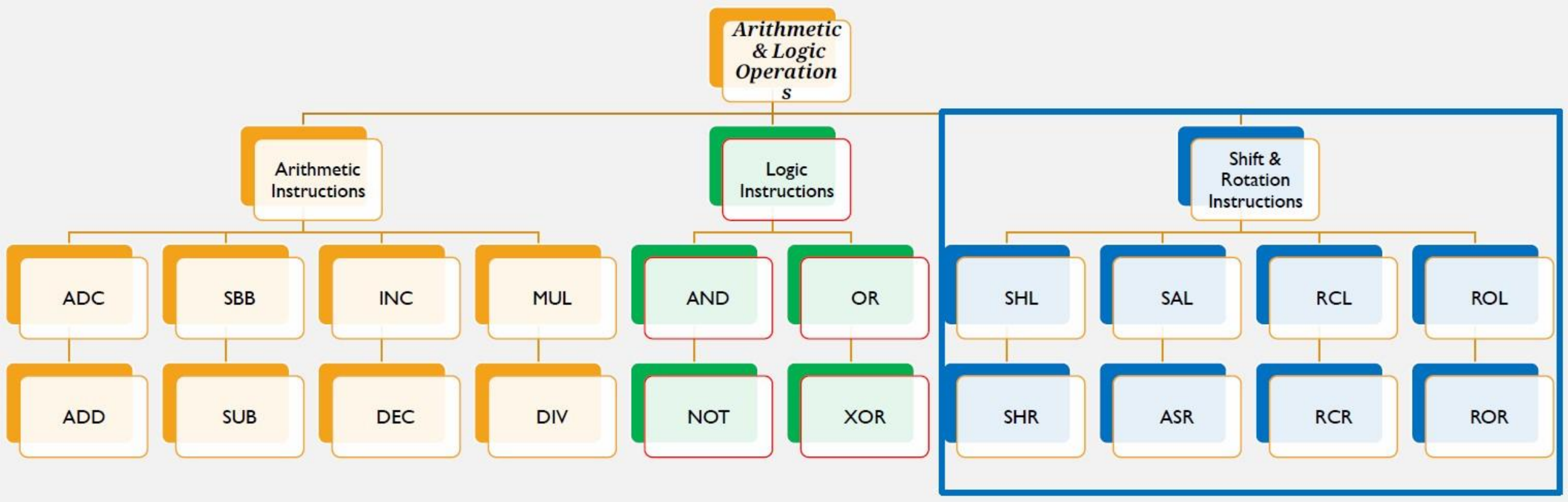
Shift & Rotation Instruction

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

Font

Shift

- In this s
Instruct

12 Rotate & Shift Instructions

13 Introduction

14 Shift Instructions

15 Shift Instructions

16 Shift Instructions

SLIDE 14 OF 25 ENGLISH (UNITED STATES)

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
```

drag a file here to open

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

F400:0154 F400:0154

```
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

message

PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM

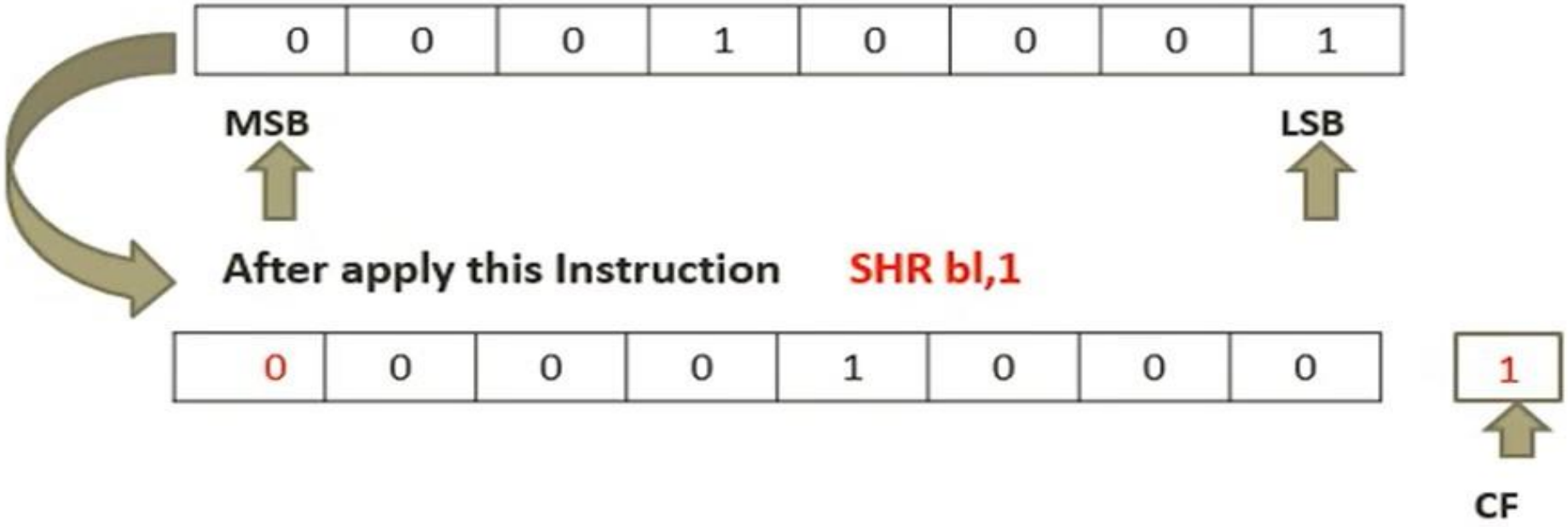
OK

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
Font

Shift Instru

2. SHR :

file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about

```
; You may customize this and other star
; The location of this template is c:\e

org 100h

mov bl,00001111b
shr bl,1
;0000 0111
;shr bl,1
;0011 1100
ret
```

original source co...

```
; You may customize this
; The location of this t

org 100h

mov bl,00001111b
shr bl,1
;0000 0111
;shr bl,1
;0011 1100
ret
```

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 help
Load reload step back single step delay ms: 0

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 F400:0154

message
PROGRAM HAS RETURNED CO
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

12 13 14 15 16

SLIDE 15 OF 25 ENGLISH (UNITED STATES)

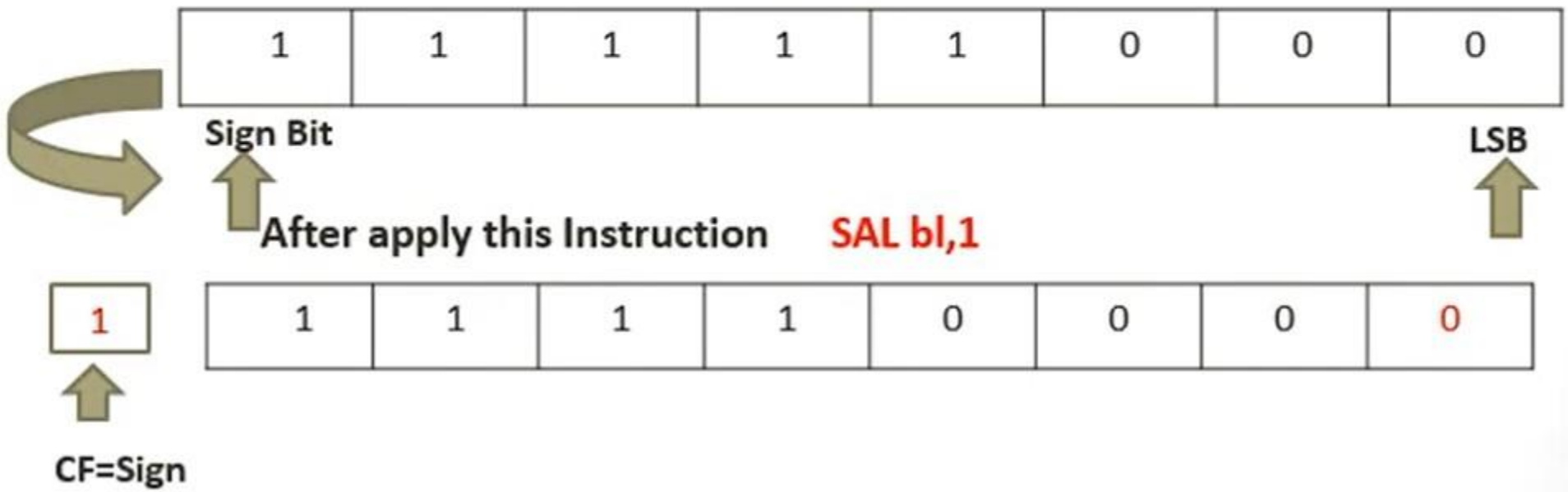
Shift Instructions

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: F4

	signed	unsigned
dec:	-12	244

ascii char: ô

oct: 364

bin: 11110100

TRANSITIONS ANIMATIONS SLIDE SHOW

Font

Shift Instru

3. SAL : (

Signifi

14 15 16 17 18

SLIDE 16 OF 25 ENGLISH (UNITED STATES)

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options help about

```
; You may customize this and other star
; The location of this template is c:\e

org 100h

mov bl,-6d
sal bl,1
;11111010
;11110100
ret
```

original source co...

```
; You may customize this
; The location of this t

org 100h

mov bl,-6d
sal bl,1
;11111010
;11110100
ret
```

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 help

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

F400:0154 F400:0154

message

PROGRAM HAS RETURNED CO
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

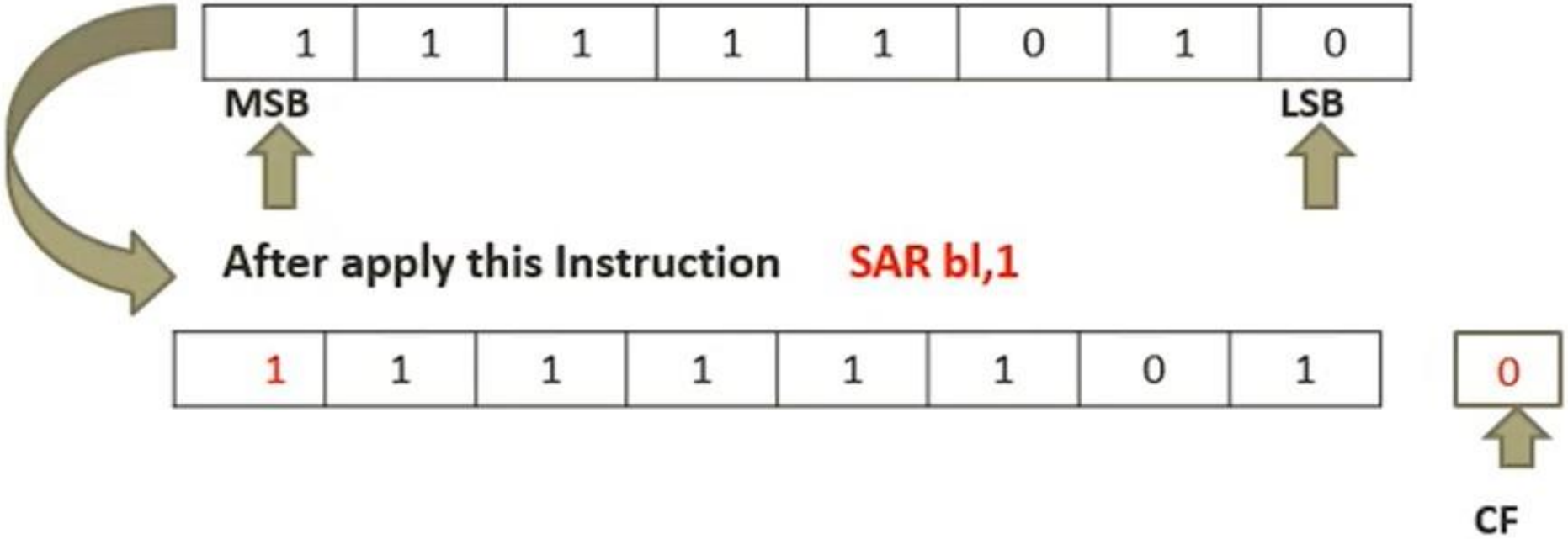
drag a file here to open

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: FB

	signed	unsigned
dec:	-5	251

ascii char: √

oct: 373

bin: 11111011

TRANSITIONS ANIMATIONS SLIDE SHOW

Calibri (Body) 18

Font

Shift Instru

4. SAR : (S

```
; You may customize this and other star
; The location of this template is c:\e

org 100h

mov bl,-9d
sar bl,1
;11110111
;11111011
ret
```

```
; You may customize this
; The location of this t

org 100h

mov bl,-9d
sar bl,1
;11110111
;11111011
ret
```

flags

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

analyse

emulator: noname.com_

Load reload step back single step delay ms: 0

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 CO
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

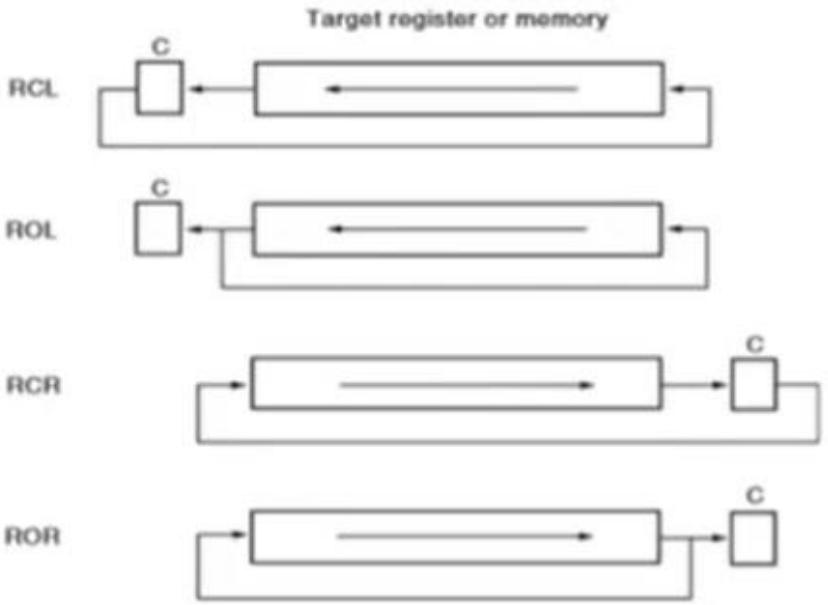
SLIDE 17 OF 25 ENGLISH (UNITED STATES)

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.

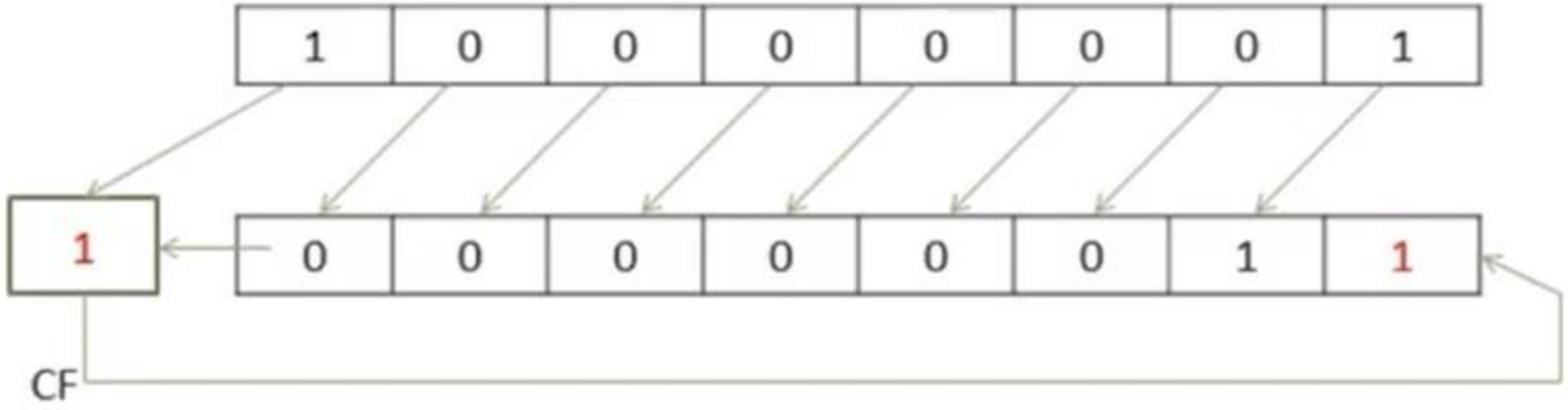


Rotate Instructions

- ROL

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

Font

Rotate Inst

- ROL

```
; You may customize this and other star
; The location of this template is c:\e

org 100h

mov bl,81h
ROL bl,1
mov al,00001111b
rol al,1
;  0001 1110
rol al,1
;  0011 1100
ret
```

```
; You may customize this
; The location of this t

org 100h

mov bl,81h
ROL bl,1
mov al,00001111b
rol al,1
;  0001 1110
rol al,1
;  0011 1100
ret
```

emulator: noname.com

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

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

```
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
...
```

message

PROGRAM HAS RETURNED CO
TO THE OPERATING SYSTEM

OK

flags

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

analyse

SLIDE 22 OF 25 ENGLISH (UNITED STATES)

drag a file here to open

2021/4/27 6:12 ENG

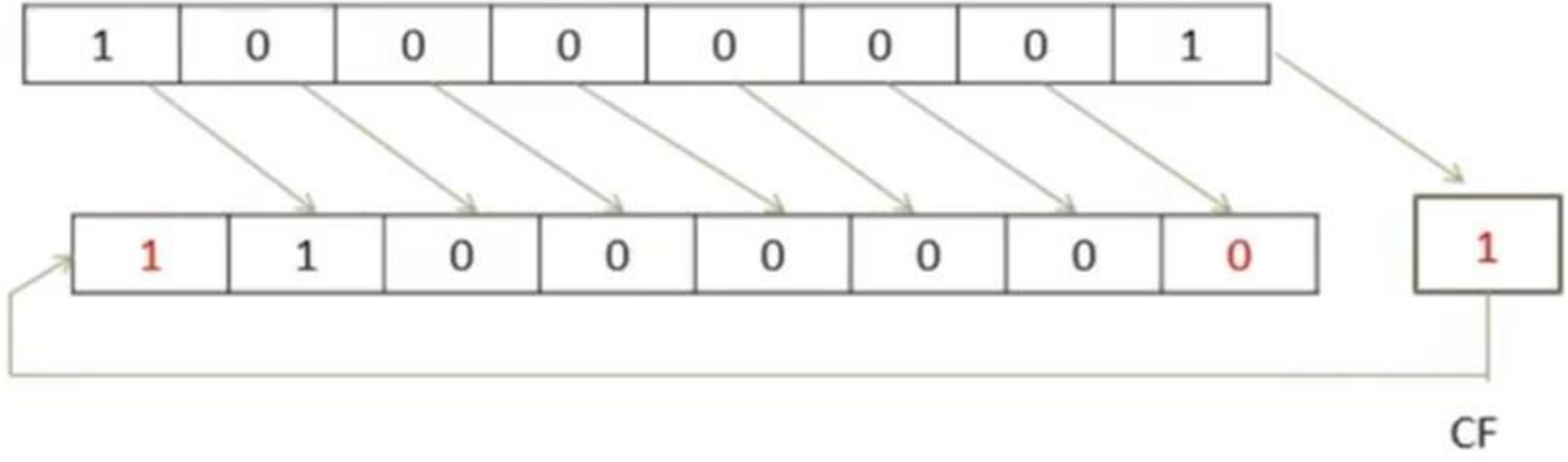
اكتب هنا للبحث

Rotate Instructions

- ROR

MOV BL ,81D

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



8 bit | 16 bit

hex: C3

	signed	unsigned
dec:	-61	195

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

```
; You may customize this and other star
; The location of this template is c:\e

org 100h

mov bl,81h
ROR bl,1
;10000001
;11000000
mov al,00001111b
ror al,1
; 10000111
ror al,1
; 11000011
```

original source co...

```
; You may customize this
; The location of this t

org 100h

mov bl,81h
ROR bl,1
;10000001
;11000000
mov al,00001111b
ror al,1
; 10000111
ror al,1
;  11000011
ret
```

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 help

Load reload step back single step delay ms: 0

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

F400:0154

```
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
...
```

message

PROGRAM HAS RETURNED CO
TO THE OPERATING SYSTEM

OK

screen source reset aux vars debug stack flags

drag a file here to open

SLIDE 23 OF 25 ENGLISH (UNITED STATES)

6:16 2021/4/27 ENG

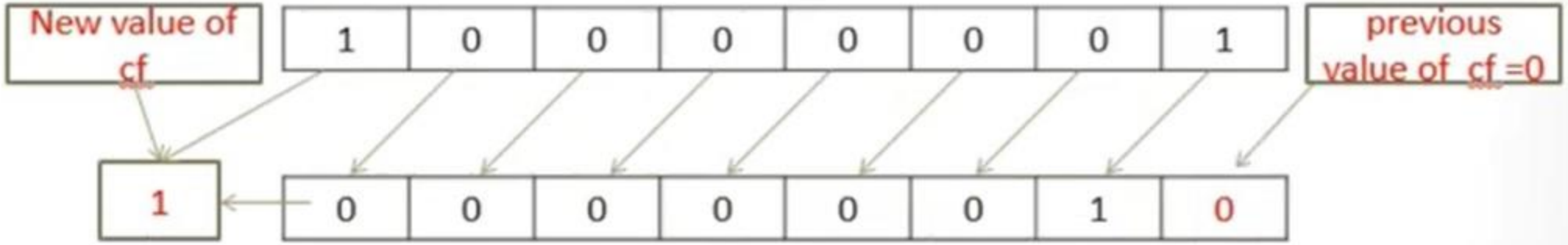
اكتب هنا للبحث

Rotate Instructions

- RCL

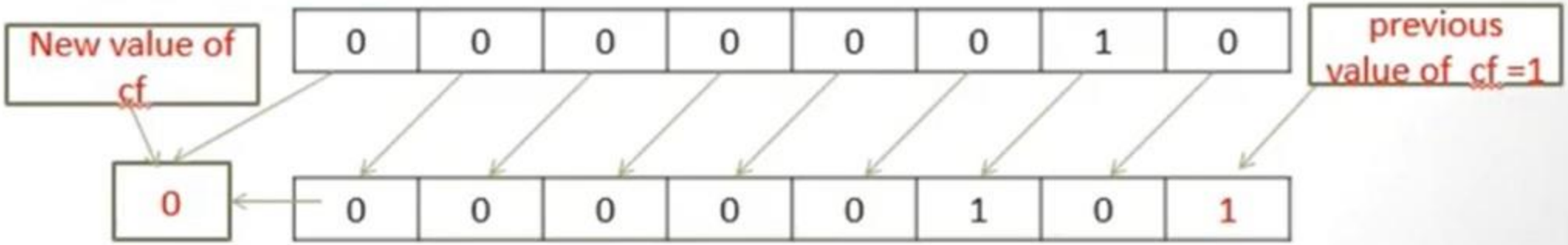
MOV BL ,81H ;

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



CF

RCL BL,1 ; After RCL bl=05h

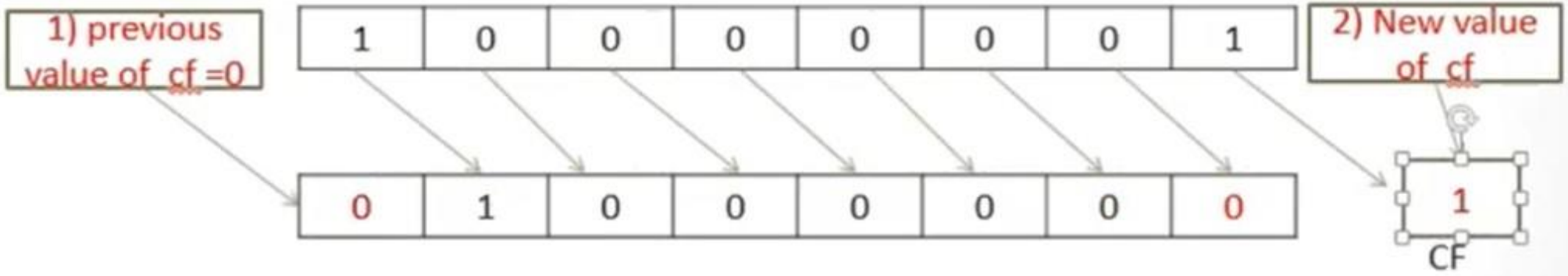


Rotate Instructions

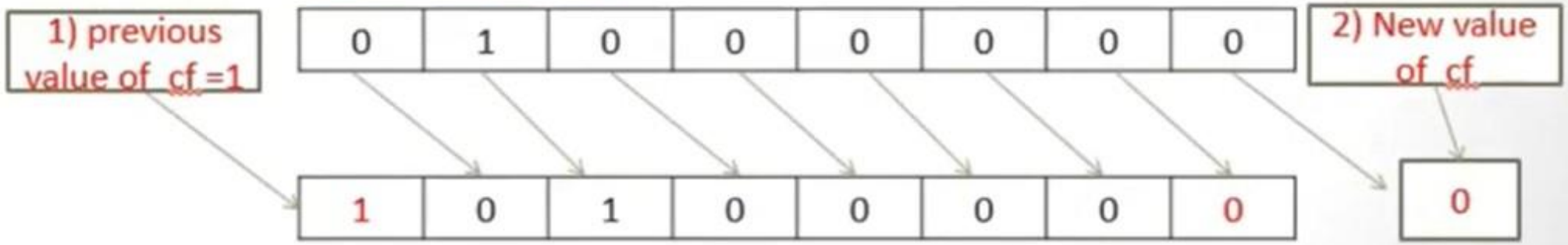
- RCR (Followed instruction are complete one program)

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
```