Microprocessor & Interfacing Lecture 20 Logic Instructions

ECS DEPARTMENT
DRONACHARYA COLLEGE OF ENGINEERING

Contents

- Introduction
- Logic Instructions
- PSW
- Examples

Introduction

- Logical instruction are those instruction which perform logical operation such as
- AND
- OR
- XOR
- Not

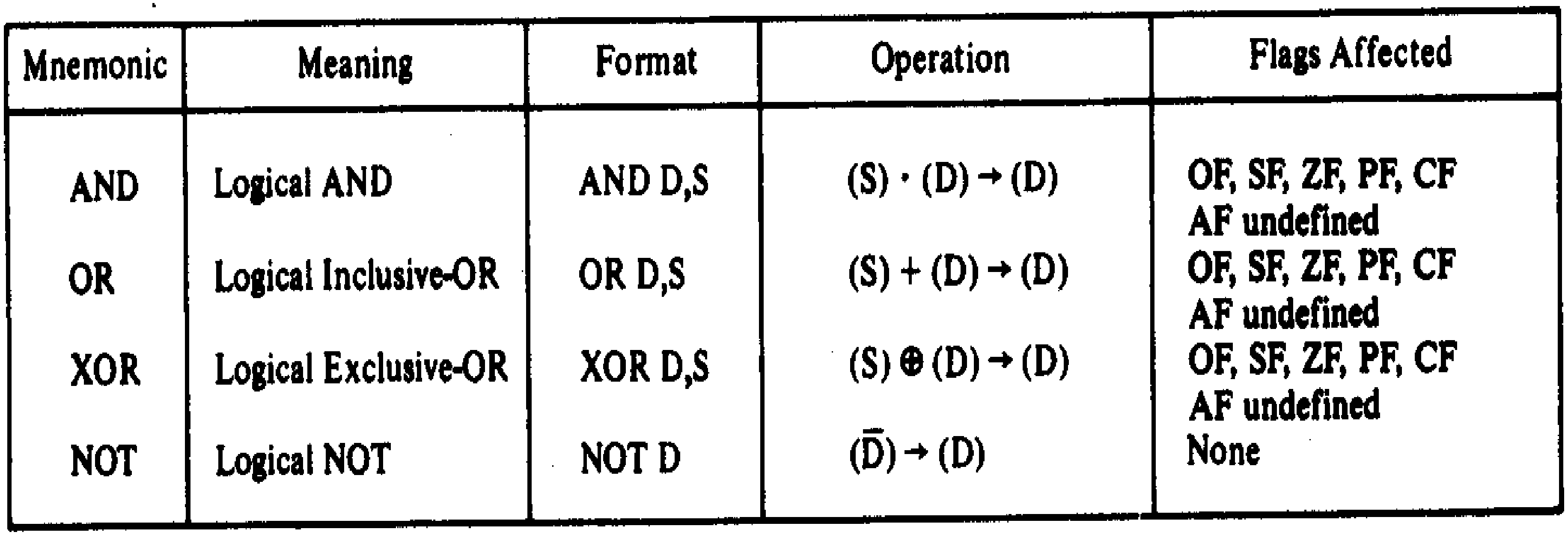
Logic Instructions

- These instructions perform logical operations on data stored in registers, memory and status flags.

- The logical operations are:
  - AND
  - OR
  - XOR
  - Rotate
  - Compare
  - Complement

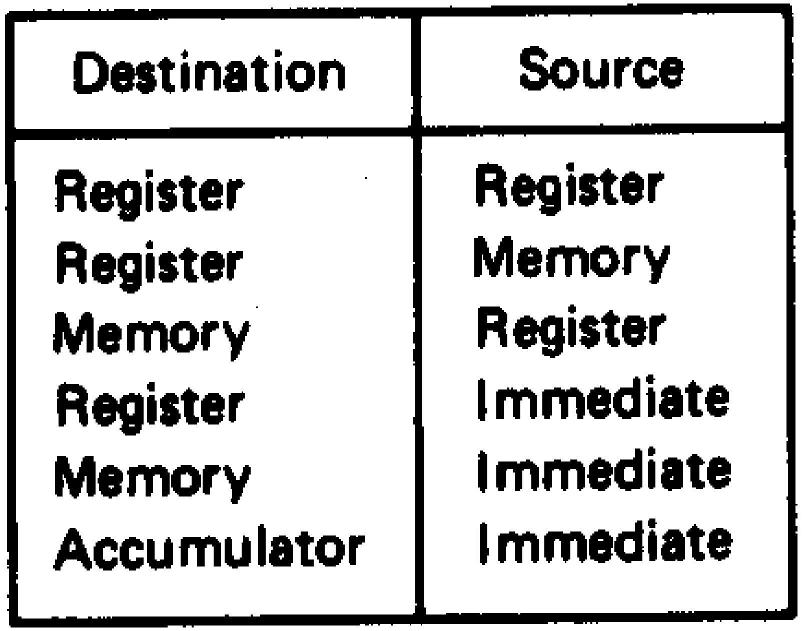
Logic Instructions

- The logic instructions include
  - AND
  - OR
  - XOR (Exclusive-OR)



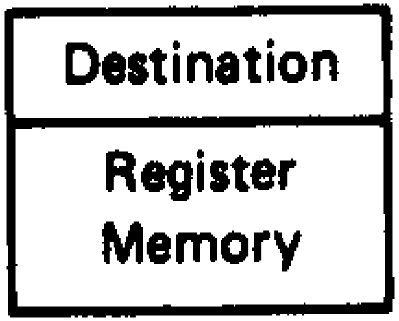
Mnemonic	Meaning	Format	Operation	Flags Affected
AND	Logical AND	AND D,S	$(S) \cdot (D) \to (D)$	OF, SF, ZF, PF, CF AF undefined
OR	Logical Inclusive-OR	OR D,S	$(S) + (D) \to (D)$	OF, SF, ZF, PF, CF AF undefined
XOR	Logical Exclusive-OR	XOR D,S	$(S) \oplus (D) \to (D)$	OF, SF, ZF, PF, CF AF undefined
NOT	Logical NOT	NOT D	$(\bar{D}) \to (D)$	None

Cont..



Destination	Source
Register	Register
Register	Memory
Memory	Register
Register	Immediate
Memory	Immediate
Accumulator	Immediate

Allowed operands for AND, OR, and XOR instructions



Destination
Register
Memory

Allowed operands for NOT instruction

PSW (Program Status Word)

- Flag affected
  - 0 reset
  - 1 set
- S Sign (Bit 7)
- Z Zero (Bit 6)
- AC Auxiliary Carry (Bit 4)
- P Parity (Bit 2)
- CY Carry (Bit 0)

AND, OR, XOR

- Any 8-bit data, or the contents of register, or memory location can logically have
  - AND operation
  - OR operation
  - XOR operation

  with the contents of accumulator.
- The result is stored in accumulator.

Rotate

- Each bit in the accumulator can be shifted either left or right to the next position.

Compare

- Any 8-bit data, or the contents of register, or memory location can be compares for:
  - Equality
  - Greater Than
  - Less Than

  with the contents of accumulator.
- The result is reflected in status flags.

Complement

- The contents of accumulator can be complemented.
- Each 0 is replaced by 1 and each 1 is replaced by 0.

Logical Instructions

Opcode	Operand	Description
CMP	R M	Compare register or memory with accumulator

- The contents of the operand (register or memory) are compared with the contents of the accumulator.
- Both contents are preserved .
- The result of the comparison is shown by setting the flags of the PSW as follows:

Opcode	Operand	Description
CMP	R M	Compare register or memory with accumulator

- if (A) < (reg/mem): carry flag is set
- if (A) = (reg/mem): zero flag is set
- if (A) > (reg/mem): carry and zero flags are reset.
- **Example:** CMP B or CMP M

Logical Instructions

Opcode	Operand	Description
CPI	8-bit data	Compare immediate with accumulator

- The 8-bit data is compared with the contents of accumulator.
- The values being compared remain unchanged.
- The result of the comparison is shown by setting the flags of the PSW as follows:

Opcode	Operand	Description
CPI	8-bit data	Compare immediate with accumulator

- if (A) < data: carry flag is set
- if (A) = data: zero flag is set
- if (A) > data: carry and zero flags are reset
- **Example:** CPI 89H

Logical Instructions

Opcode	Operand	Description
ANA	R M	Logical AND register or memory with accumulator

- The contents of the accumulator are logically ANDed with the contents of register or memory.
- The result is placed in the accumulator.
- If the operand is a memory location, its address is specified by the contents of H-L pair.
- S, Z, P are modified to reflect the result of the operation.
- CY is reset and AC is set.
- **Example:** ANA B or ANA M.

Opcode	Operand	Description
ANI	8-bit data	Logical AND immediate with accumulator

- The contents of the accumulator are logically ANDed with the 8-bit data.
- The result is placed in the accumulator.
- S, Z, P are modified to reflect the result.
- CY is reset, AC is set.
- **Example:** ANI 86H.

Logical Instructions

Opcode	Operand	Description
ORA	R M	Logical OR register or memory with accumulator

- The contents of the accumulator are logically ORed with the contents of the register or memory.
- The result is placed in the accumulator.
- If the operand is a memory location, its address is specified by the contents of H-L pair.
- S, Z, P are modified to reflect the result.
- CY and AC are reset.
- **Example:** ORA B or ORA M.

Opcode	Operand	Description
ORI	8-bit data	Logical OR immediate with accumulator

- The contents of the accumulator are logically ORed with the 8-bit data.
- The result is placed in the accumulator.
- S, Z, P are modified to reflect the result.
- CY and AC are reset.
- **Example:** ORI 86H.

Logical Instructions

Opcode	Operand	Description
XRA	R M	Logical XOR register or memory with accumulator

- The contents of the accumulator are XORed with the contents of the register or memory.
- The result is placed in the accumulator.
- If the operand is a memory location, its address is specified by the contents of H-L pair.
- S, Z, P are modified to reflect the result of the operation.
- CY and AC are reset.
- **Example:** XRA B or XRA M.

Opcode	Operand	Description
XRI	8-bit data	XOR immediate with accumulator

- The contents of the accumulator are XORed with the 8-bit data.
- The result is placed in the accumulator.
- S, Z, P are modified to reflect the result.
- CY and AC are reset.
- **Example:** XRI 86H.

Logical Instructions

Opcode	Operand	Description
RAL	None	Rotate accumulator left through carry

- Each binary bit of the accumulator is rotated left by one position through the Carry flag.
- Bit D7 is placed in the Carry flag, and the Carry flag is placed in the least significant position Do.
- CY is modified according to bit D7.
- S, Z, P, AC are not affected.
- **Example:** RAL.

Opcode	Operand	Description
RAR	None	Rotate accumulator right through carry

- Each binary bit of the accumulator is rotated right by one position through the Carry flag.
- Bit Do is placed in the Carry flag, and the Carry flag is placed in the most significant position D7.
- CY is modified according to bit Do.
- S, Z, P, AC are not affected.
- **Example:** RAR.

Circular Left shift

Opcode	Operand	Description
RLC	None	Rotate accumulator left

- Each binary bit of the accumulator is rotated left by one position.
- Bit D7 is placed in the position of Do as well as in the Carry flag.
- CY is modified according to bit D7.
- S, Z, P, AC are not affected.
- **Example:** RLC.

Circular right shift

Opcode	Operand	Description
RRC	None	Rotate accumulator right

- Each binary bit of the accumulator is rotated right by one position.
- Bit Do is placed in the position of D7 as well as in the Carry flag.
- CY is modified according to bit Do.
- S, Z, P, AC are not affected.
- **Example:** RRC.

Logical Instructions

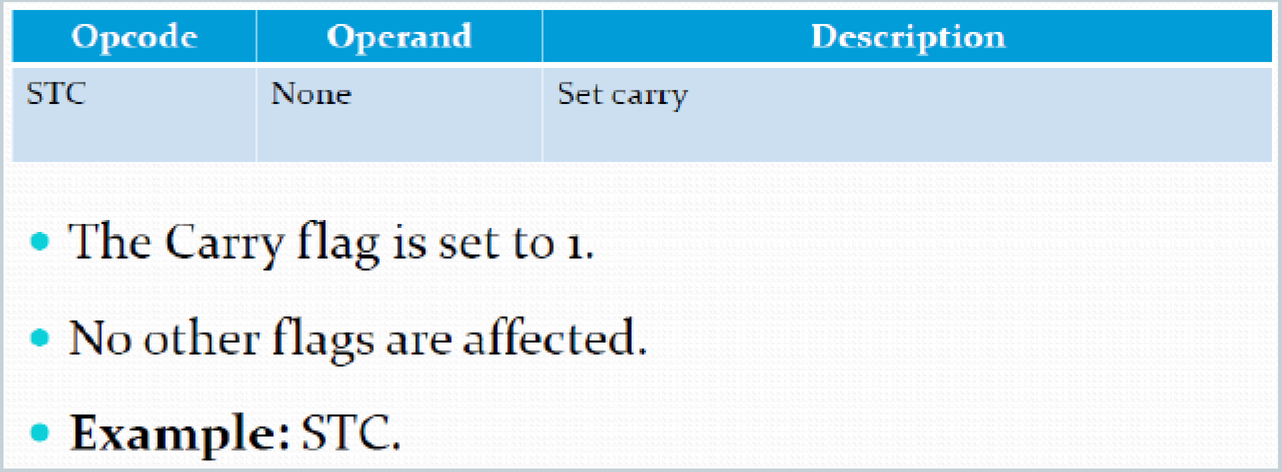
Opcode	Operand	Description
CMA	None	Complement accumulator

- The contents of the accumulator are complemented.
- No flags are affected.
- **Example:** CMA.

Opcode	Operand	Description
CMC	None	Complement carry

- The Carry flag is complemented.
- No other flags are affected.
- **Example:** CMC.

Logical Instructions



Opcode	Operand	Description
STC	None	Set carry

- The Carry flag is set to 1.
- No other flags are affected.
- **Example:** STC.

Example

- Describe the results of executing the following instructions?

```
MOV AL, 01010101B
AND AL, 00011111B
OR AL, 11000000B
XOR AL, 00001111B
NOT AL
```

Solution:

- (AL)=01010101_2 . 00011111_2= 00010101_2=15_{16}

Executing the OR instruction, we get

- (AL)= 00010101_2 +11000000_2= 11010101_2=$D5_{16}$

Executing the XOR instruction, we get

- (AL)= 11010101_2 XOR 00001111_2= 11011010_2=DA_{16}

Executing the NOT instruction, we get

- (AL)= (NOT)11011010_2 = 00100101_2=25_{16}

Example

- Masking and setting bits in a register
- Solution: Mask off the upper 12 bits of the word of data in AX
- AND AX, $000F_{16}$

Setting B4 of the byte at the offset address CONTROL_FLAGS

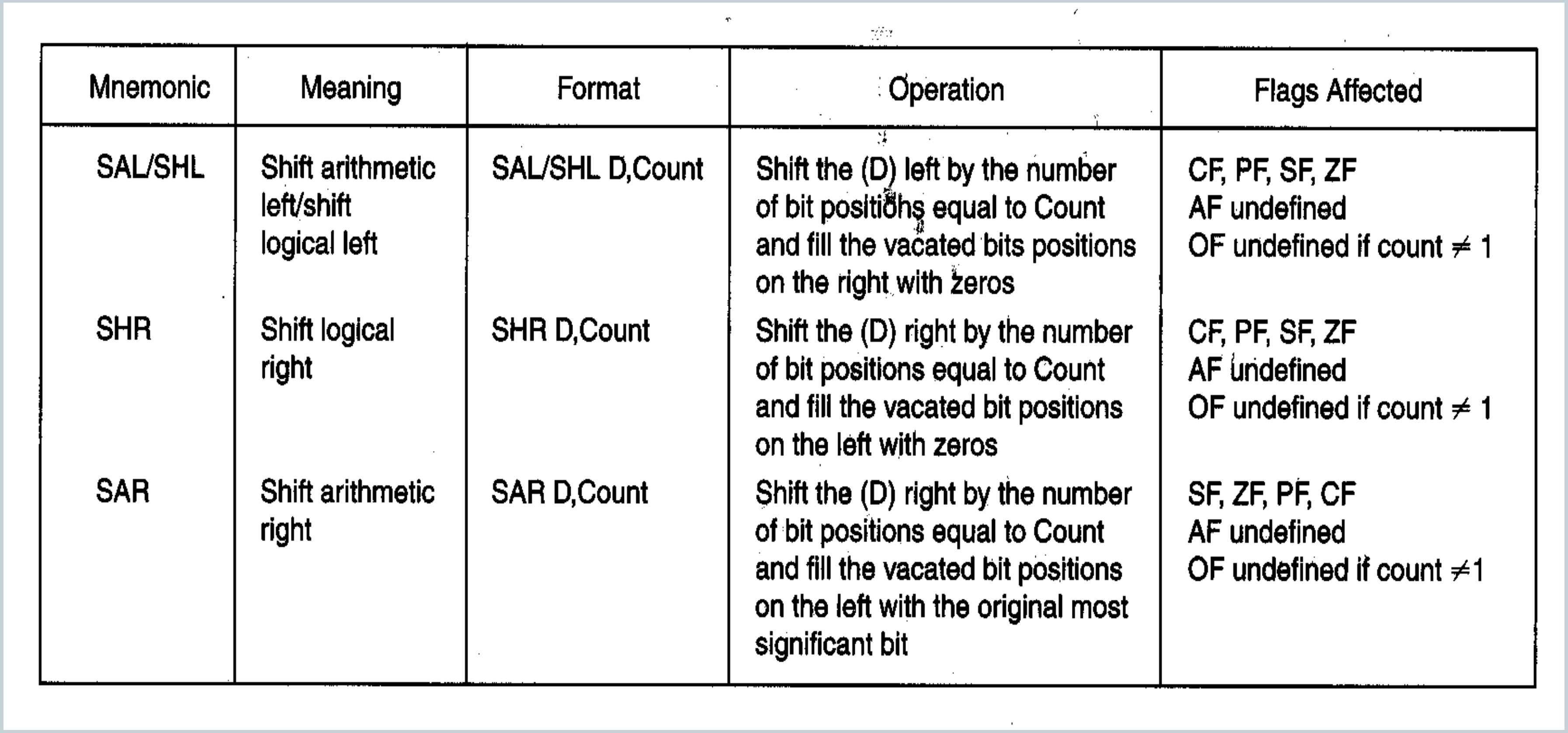
MOV AL, [CONTROL_FLAGS]

OR AL, 10H

MOV [CONTROL_FLAGS], AL

- Executing the above instructions, we get (AL)=$XXXXXXXX_2$ $+00010000_2$= $XXX1XXXX_2$

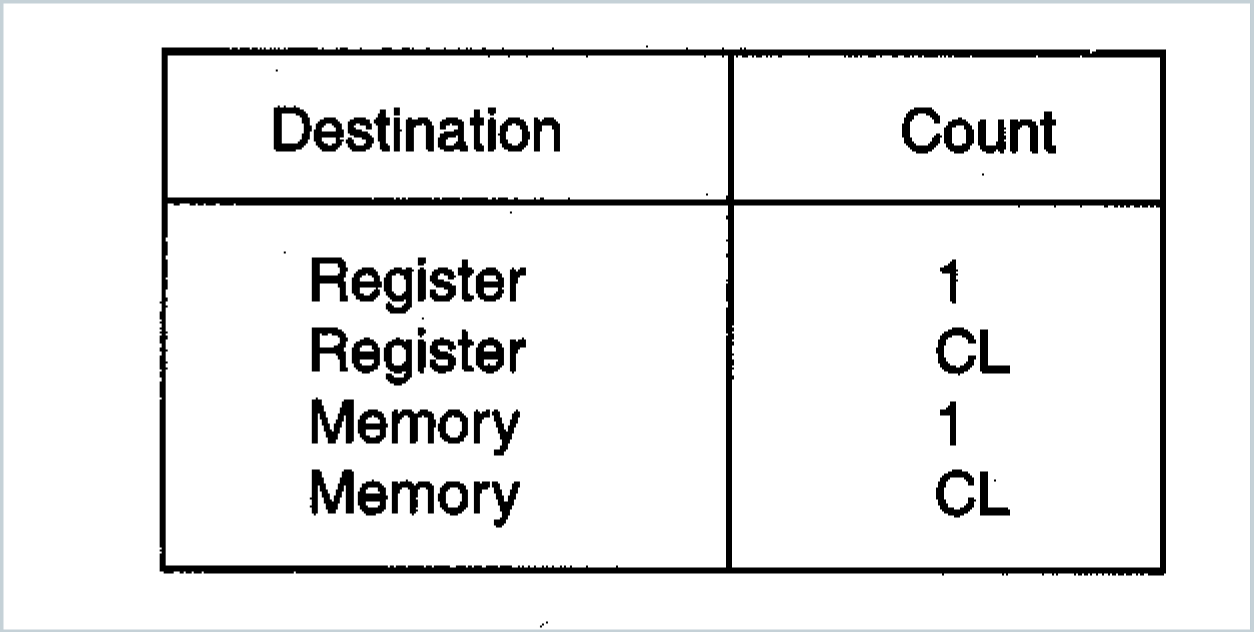
Shift Instructions



Mnemonic	Meaning	Format	Operation	Flags Affected
SAL/SHL	Shift arithmetic left/shift logical left	SAL/SHL D,Count	Shift the (D) left by the number of bit positions equal to Count and fill the vacated bits positions on the right with zeros	CF, PF, SF, ZF AF undefined OF undefined if count ≠ 1
SHR	Shift logical right	SHR D,Count	Shift the (D) right by the number of bit positions equal to Count and fill the vacated bit positions on the left with zeros	CF, PF, SF, ZF AF undefined OF undefined if count ≠ 1
SAR	Shift arithmetic right	SAR D,Count	Shift the (D) right by the number of bit positions equal to Count and fill the vacated bit positions on the left with the original most significant bit	SF, ZF, PF, CF AF undefined OF undefined if count ≠1

Cont..

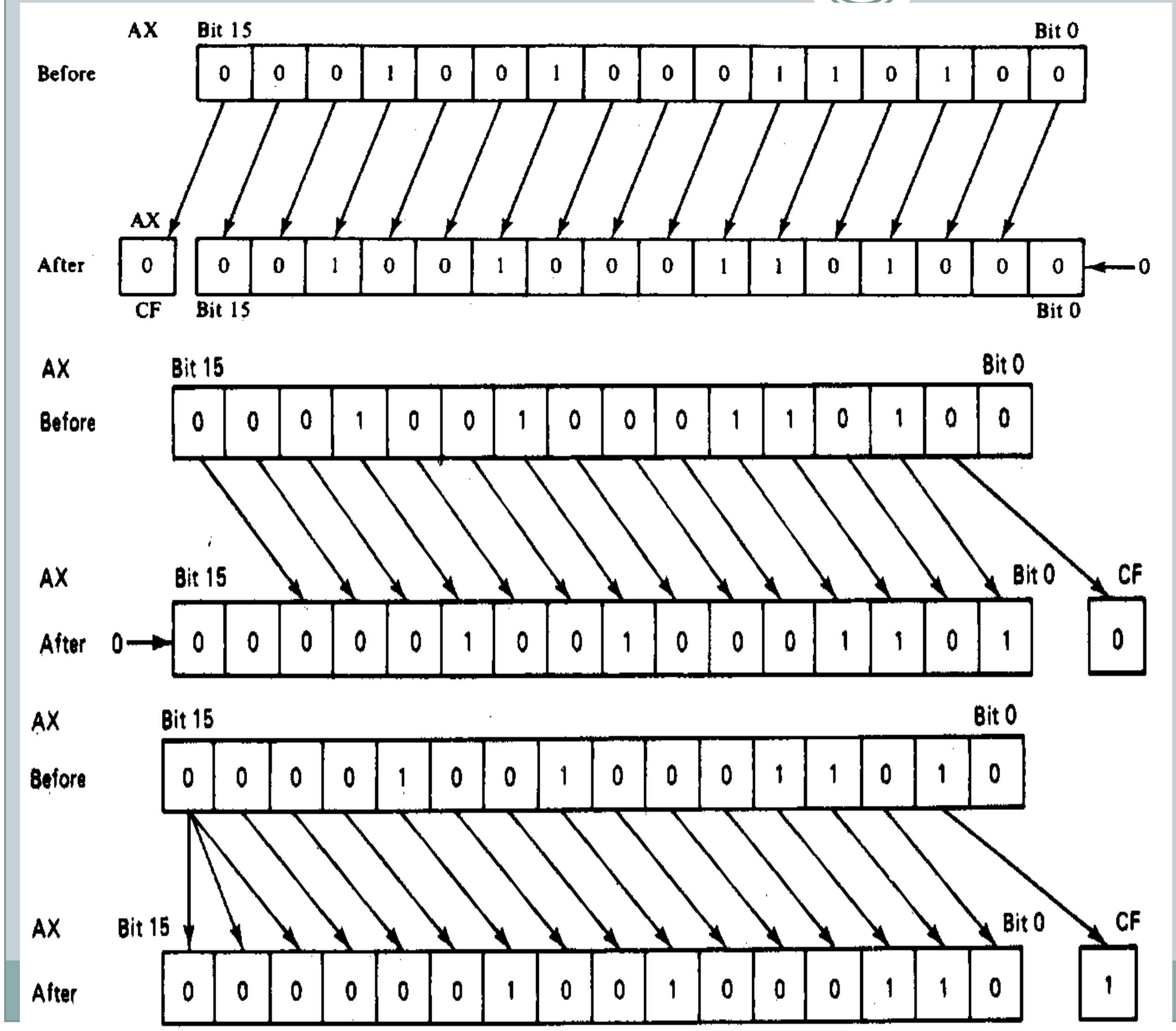
- Shift instructions: SHL, SHR, SAL, SAR



Destination	Count
Register	1
Register	CL
Memory	1
Memory	CL

Allowed operands for shift instructions

Cont..



SHL AX, 1

SHR AX, CL
(CL)=2

SAR AX, CL
(CL)=2

Example

- Assume that CL contains 02_{16} and AX contains $091A_{16}$. Determine the new contents of AX and the carry flag after the instruction SAR AX, CL is executed
- Solution:

$(AX)=0000001001000110_2=0246_{16}$ and the carry flag is $(CF)=1_2$

Example

- Isolate the bit B3 of the byte at the offset address CONTROL_FLAGS.
- Solution:

```
MOV AL, [CONTROL_FLAGS]
MOV CL, 04H
SHR AL, CL
```

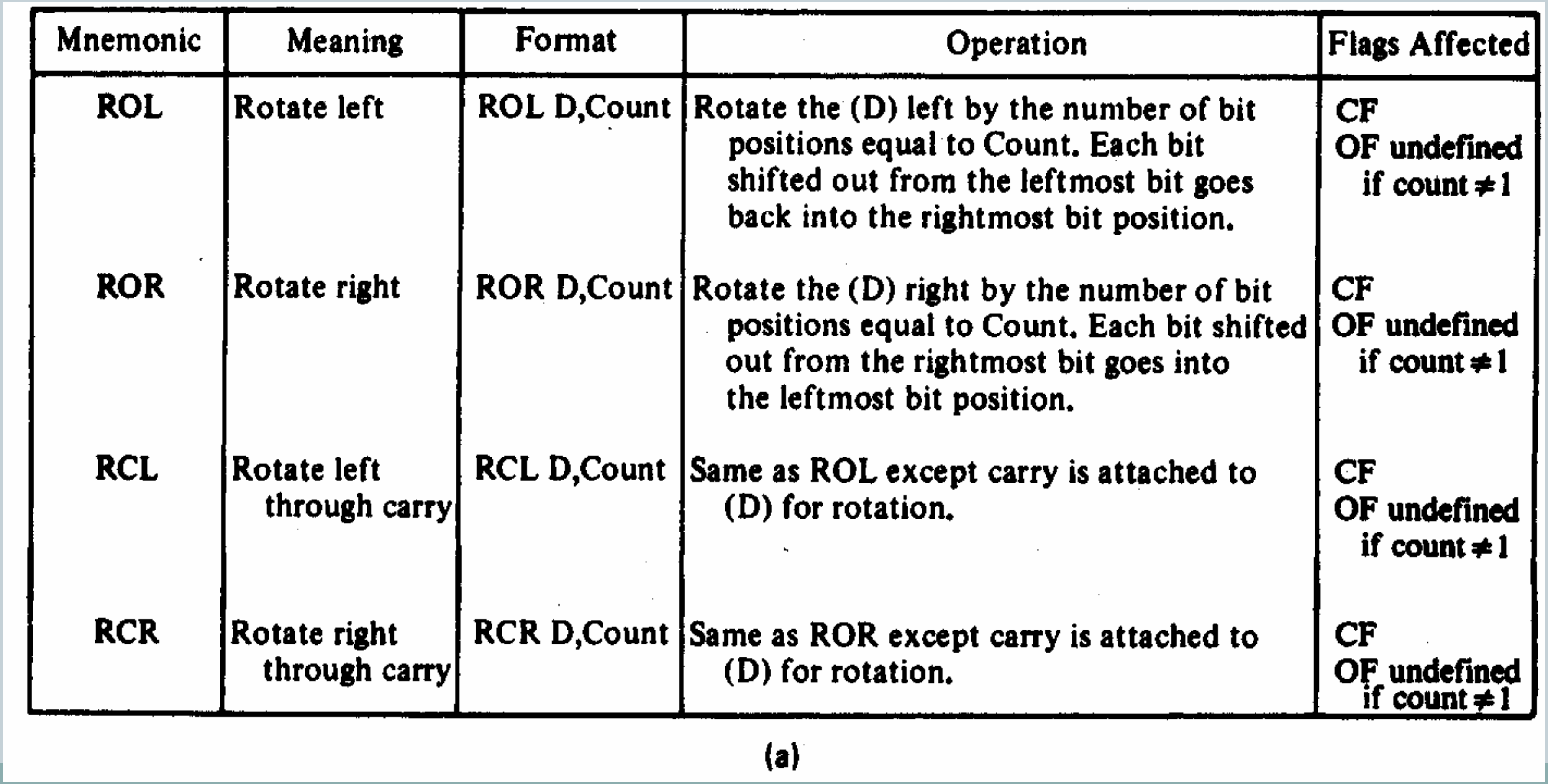
- Executing the instructions, we get

  $(AL)=0000B_7B_6B_5B_4$

  $(CF)=B_3$

Rotate Instructions

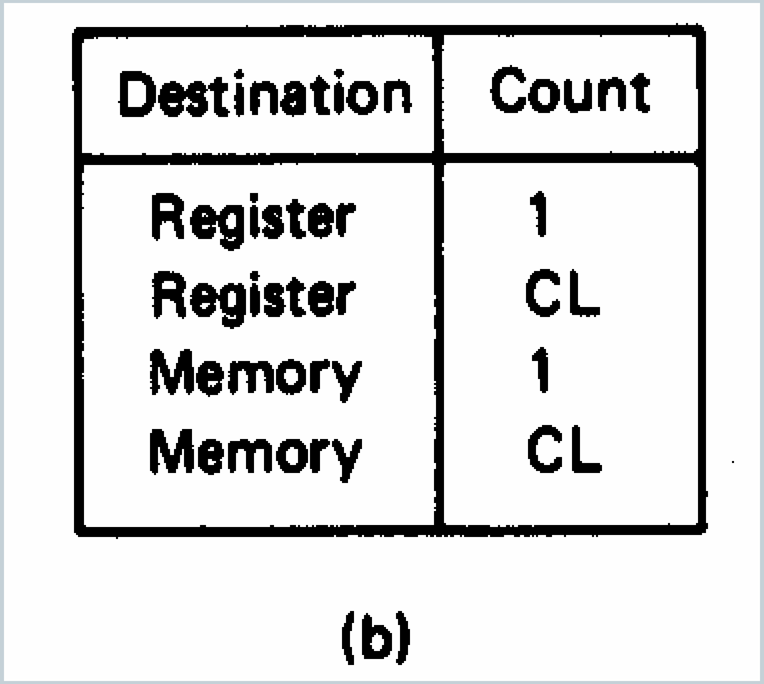
- Rotate instructions: ROL, ROR, RCL, RCR



Mnemonic	Meaning	Format	Operation	Flags Affected
ROL	Rotate left	ROL D,Count	Rotate the (D) left by the number of bit positions equal to Count. Each bit shifted out from the leftmost bit goes back into the rightmost bit position.	CF OF undefined if count ≠ 1
ROR	Rotate right	ROR D,Count	Rotate the (D) right by the number of bit positions equal to Count. Each bit shifted out from the rightmost bit goes into the leftmost bit position.	CF OF undefined if count ≠ 1
RCL	Rotate left through carry	RCL D,Count	Same as ROL except carry is attached to (D) for rotation.	CF OF undefined if count ≠ 1
RCR	Rotate right through carry	RCR D,Count	Same as ROR except carry is attached to (D) for rotation.	CF OF undefined if count ≠ 1

(a)

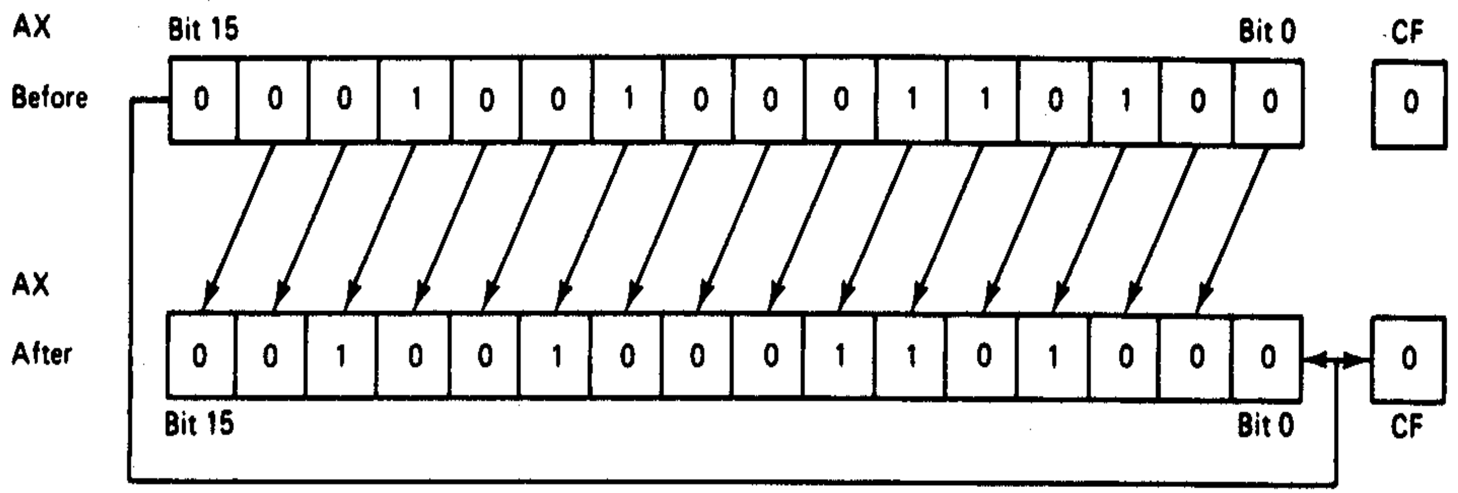
Cont..



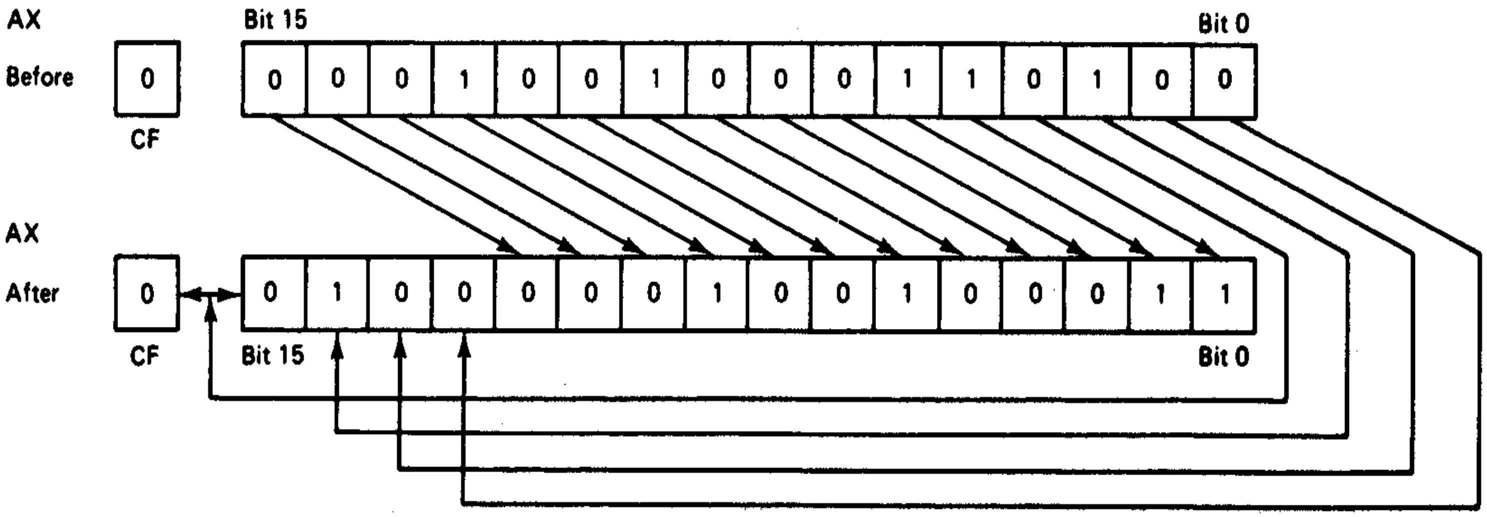
Destination	Count
Register	1
Register	CL
Memory	1
Memory	CL

(b)

Cont..



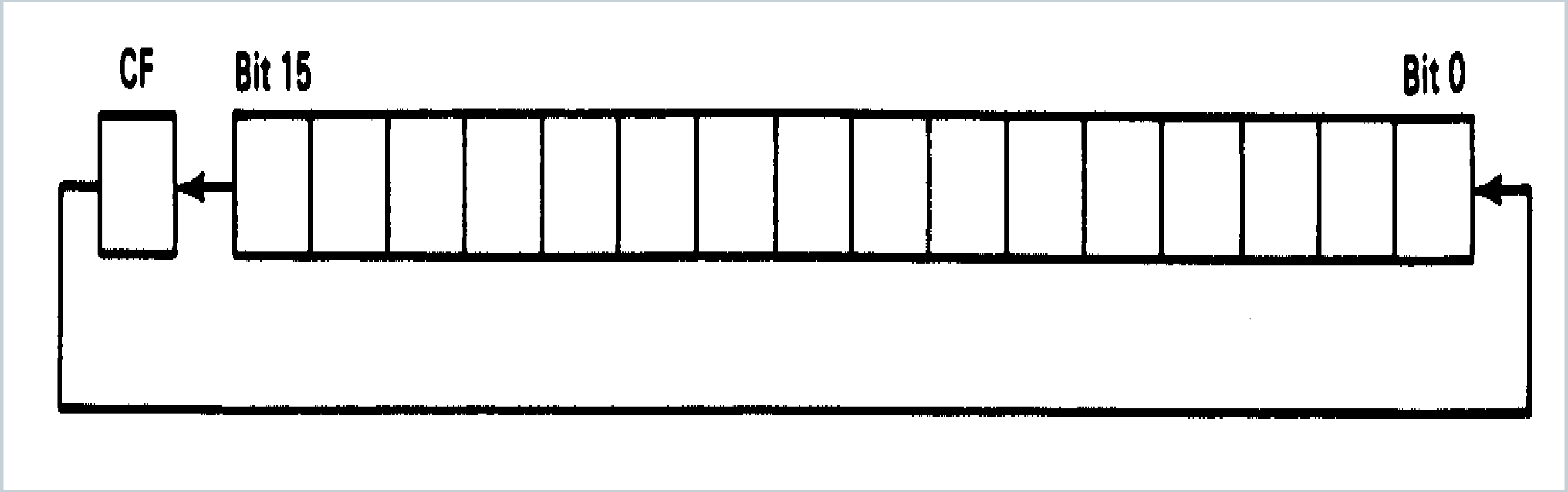
ROL AX, 1



ROR AX, CL
(CL)=4

Cont..

- For RCL, RCR, the bits are rotate through the carry flag



Example

- What is the result in BX and CF after execution of the following instructions?

RCR BX, CL

- Assume that, prior to execution of the instruction, $(CL)=04_{16}$, $(BX)=1234_{16}$, and $(CF)=0$

Solution:

- The original contents of BX are $(BX) = 0001001000110100_2 = 1234_{16}$
- Execution of the RCR command causes a 4-bit rotate right through carry to take place on the data in BX, the results are
  - $(BX) = 1000000100100011_2 = 8123_{16}$
  - $(CF) = 0_2$

Example

- Disassembly and addition of 2 hexadecimal digits stored as a byte in memory.

Solution:

```
MOV AL, [HEX_DIGITS]
MOV BL, AL
MOV CL, 04H
ROR BL, CL
AND AL, 0FH
AND BL, 0FH
ADD AL, BL
```