Computer Structure and Language

The 8086/8088 Assembly Language

Hamid Sarbazi-Azad
Department of Computer Engineering
Sharif University of Technology (SUT)
Tehran, Iran

1

8086/88 has 7 types of instructions:

1. Data Transfer Instructions
2. Arithmetic Instructions
3. Bit Manipulation Instructions
4. String Instructions
5. Program Execution Transfer Instructions
6. Processor Control Instructions
7. Interrupt Instructions

2

The 8086/88's Logic Instructions:

And Instructions: AND

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **Register/Memory with Register to Either**

  Possible combinations:

  and reg1,reg2 ≡ reg1 ← (reg1)∧(reg2); update Flags;
  and reg,memory ≡ reg ← (reg) ∧(EA); update Flags;
  and memory,reg ≡ EA ← (EA) ∧(reg); update Flags;

Example:

and cx,word ptr [si -5] ≡ cx ← (cx)∧($M_{(si)-5}$); update Flags;

dw Md Reg R/M Disp. Low

Machine code: 00100011 01 001 100 11111011 ≡ 234CFBh

3

The 8086/88's Logic Instructions:

And Instructions: AND

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **Immediate to Register/Memory**

Example 1:

and bx,25h ≡ bx ← (bx)∧ 0025h; update Flags;

Machine code:

w Md R/M Data Low Data High

10000001 11 100 011 00100101 00000000 ≡ 81E32500h

Example 2:

and word ptr [bx+20][di],1234h ≡ $M_{(bx)+(di)+20}$ ←($M_{(bx)+(di)+20}$)∧1234h; update Flags;

Machine code:

w Md R/M Disp. Low Data Low Data High

10000001 01 100 001 00010100 00110100 00010010 ≡ 8161143412h

4

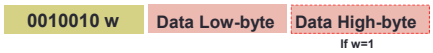
The 8086/88's Logic Instructions:

And Instructions: AND

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **Immediate to Accumulator**

and al,data8 ≡ al ← (al) ∧ data8; update Flags; if w=0
and ax,data16 ≡ ax ← (ax) ∧ data16; update Flags; if w=1



Example 1: and ax,100 ≡ ax ← (ax) ∧ 100; update Flags;

w Data Low Data High
Machine code: 00100101 01100100 00000000 ≡ 256400h

Example 2: and al,-1 ≡ al ← (al) ∧ FFh; update Flags;

w Data Low
Machine code: 00100100 11111111 ≡ 24FFh

5

The 8086/88's Logic Instructions:

Or Instructions: OR

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **Register/Memory with Register to Either**

Possible combinations:

or reg1,reg2 ≡ reg1 ← (reg1) ∨ (reg2); update Flags;
or reg,memory ≡ reg ← (reg) ∨ (EA); update Flags;
or memory,reg ≡ EA ← (EA) ∨ (reg); update Flags;

Example:

or dx,word ptr [bx + 5] ≡ dx ← (dx) ∨ ($M_{(bx)+5}$); update Flags;

dw Md Reg R/M Disp. Low
Machine code: 00001011 01 010 111 00000101 ≡ 0B5705h

6

The 8086/88's Logic Instructions:

Or Instructions: OR

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **Immediate to Register/Memory**

Example 1:

or dl,2 ≡ dl ← (dl)∨ 02h; update Flags;

Machine code:

w	Md	R/M	Data Low	
10000000	11 001	010	00000010	≡ 80CA02h

Example 2:

or word ptr [bx+1][si+1],8000h ≡ $M_{(bx)+(si)+2} \leftarrow (M_{(bx)+(si)+2}) \vee$ 8000h; update Flags;

Machine code:

w	Md		R/M	Disp. Low	Data Low	Data High	
10000001	01	001	000	00000010	00000000	10000000	≡ 8148020080h

7

The 8086/88's Logic Instructions:

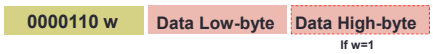
Or Instructions: OR

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **Immediate to Accumulator**

or al,data8 ≡ al ← (al)∨data8; update Flags; if w=0

or ax,data16 ≡ ax ← (ax)∨data16; update Flags; if w=1



Example 1: or ax,1000h ≡ ax ← (ax) ∨ 1000; update Flags;

Machine code:

w	Data Low	Data High	
00001101	00000000	00010000	≡ 0D0010h

Example 2: or al,-128 ≡ al ← (al) ∨ 80h; update Flags;

Machine code:

w	Data Low	
00001100	10000000	≡ 0C80h

8

The 8086/88's Logic Instructions:

Exclusive-Or Instructions: XOR

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **Register/Memory with Register to Either**

Possible combinations:

xor reg1,reg2 ≡ reg1 ← (reg1) ⊕ (reg2); update Flags;
xor reg,memory ≡ reg ← (reg) ⊕ (EA); update Flags;
xor memory,reg ≡ EA ← (EA) ⊕ (reg); update Flags;

Example:

xor bp,word ptr [bx] ≡ bp ← (bp) ⊕ ($M_{(bx)}$); update Flags;

	dw	Md	Reg	R/M	
Machine code:	00110011	00	101	111	≡ 332Fh

9

The 8086/88's Logic Instructions:

Exclusive-Or Instructions: XOR

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **Immediate to Register/Memory**

Example 1:

xor cl,60h ≡ cl ← (cl) ⊕ 60h; update Flags;

Machine code:

	w	Md	R/M	Data Low	
	10000000	11	110 001	01100000	≡ 80F160h

Example 2:

xor word ptr [bx+32],4000h ≡ $M_{(bx)+32}$ ← ($M_{(bx)+32}$) ⊕ 4000h; update Flags;

Machine code:

	w	Md	R/M	Disp. Low	Data Low	Data High	
	10000001	01	110 111	00100000	00000000	01000000	≡ 8177200040h

10

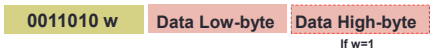
The 8086/88's Logic Instructions:

Exclusive-Or Instructions: XOR

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **Immediate to Accumulator**

xor al,data8 ≡ al ← (al)⊕ data8; update Flags; if w=0
xor ax,data16 ≡ ax ← (ax)⊕ data16; update Flags; if w=1



Example 1: xor ax,7432h ≡ ax ← (ax)⊕ 7432; update Flags;

w Data Low Data High
Machine code: 00110101 00110010 01110100 ≡ 353274h

Example 2: xor al,127 ≡ al ← (al)⊕ 7Fh; update Flags;

w Data Low
Machine code: 00110100 01111111 ≡ 347Fh

11

The 8086/88's Logic Instructions:

Test Instructions: TEST

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **Test Register/Memory and Register**

Possible combinations:
test reg1,reg2 ≡ Realize (reg1) ∧(reg2) and update Flags;
test memory,reg ≡ Realize (EA) ∧(reg) and update Flags;
test reg,memory ≡ Realize (Reg) ∧(EA) and update Flags;

Example:

test cx,word ptr [si -5] ≡ Realize (cx) ∧ ($M_{(si)-5}$) and update Flags;

w Md Reg R/M Disp. Low
Machine code: 10000101 01 001 100 11111011 ≡ 854CFBh

12

The 8086/88's Logic Instructions:

Test Instructions: TEST

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **Test Immediate and Register/Memory**

Example 1:

test cx,256h ≡ Realize (cx)∧ 256h and update Flags;

Machine code:

w	Md	R/M	Data Low	Data High	
11110111	11	000 001	01010110	00000010	≡ F7C15602h

Example 2:

test word ptr [bx+21][si],1004h ≡ Realize ($M_{(bx)+(si)+21}$)∧1004h and update Flags;

Machine code:

w	Md	R/M	Disp. Low	Data Low	Data High	
11110111	01	000 000	00010101	00000100	00010000	≡ F740150410h

13

The 8086/88's Logic Instructions:

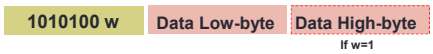
Test Instructions: TEST

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **Test Immediate and Accumulator**

test al,data8 ≡ Realize (al)∧data8 and update Flags; if w=0

test ax,data16 ≡ Realize (ax)∧data16 and update Flags; if w=1



Example 1: test ax,-1 ≡ Realize (ax)∧ FFFFh and update Flags;

Machine code:

w	Data Low	Data High	
10101001	11111111	11111111	≡ A9FFFFh

Example 2: test al,10101010B ≡ Realize (al)∧ AAh and update Flags;

Machine code:

w	Data Low	
10101000	10101010	≡ A8AAh

14

The 8086/88's Logic Instructions:

Bit-wise Complement: NOT

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	-	-	-	x	x	u	x	0

- **NOT (1's complement) Register/Memory**

Example 1:

not bx ≡ bx ← $\overline{(bx)}$; update Flags;

Machine code: 11110111 (w, Md=11, 010, R/M=011) ≡ F7D3h

Example 2:

not byte ptr array+30[si] ≡ $M_{array+(si)+30} \leftarrow \overline{(M_{array+(si)+30})}$; update Flags;

@array = 300h

Machine code:

w	Md		R/M	Disp. Low	Disp. High
11110110	10	010	100	00011110	00000011

≡ F6941E03h

15

The 8086/88's Logic Instructions:

Shift Instructions:

OF	DF	IF	TF	SF	ZF	AF	PF	CF
x	-	-	-	x	x	u	x	x

- **Shift Logical/Arithmetic Left: SHL/SAL**

shl/sal operand,1 ≡ operand ← (operand)<<1; update Flags; if v=0

shl/sal operand,cl ≡ operand ← (operand)<<(cl); update Flags; if v=1

Example 1: shl bx,1 ≡ bx ← (bx)<<1; update Flags;

Machine code: 11010001 (vw, Md=11, 100, R/M=011) ≡ D1E3h

Example 2:

sal byte ptr [si+2],cl ≡ $M_{(si)+2} \leftarrow (M_{(si)+2})<<(cl)$; update Flags;

Machine code:

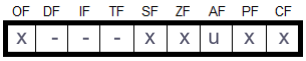
vw	Md		R/M	Disp. Low
11010010	01	100	100	00000010

≡ D26402h

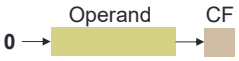
16

The 8086/88's Logic Instructions:

Shift Instructions:



- **Shift Logical Right: SHR**



shr operand,1 ≡ operand ← (operand)>>1; update Flags; if v=0
shr operand,cl ≡ operand ← (operand)>>(cl); update Flags; if v=1

Example 1: shr bl,1 ≡ bl ← (bl)>>1; update Flags;

Machine code: 11010000 (vw Md R/M) = 11 101 011 ≡ D0EBh

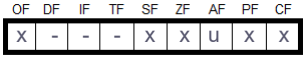
Example 2:
shr word ptr [bx],cl ≡ $M_{(bx)}$ ← ($M_{(bx)}$)>>(cl); update Flags;

Machine code:
11010011 (vw Md R/M) = 00 101 111 ≡ D32Fh

17

The 8086/88's Logic Instructions:

Shift Instructions:



- **Shift Arithmetic Right: SAR**

sar operand,1 ≡ operand ← (operand)>>1; update Flags; if v=0
sar operand,cl ≡ operand ← (operand)>>(cl); update Flags; if v=1

Example 1: sar cx,1 ≡ cx ← (cx)>>1; update Flags;

Machine code: 11010001 (vw Md R/M) = 11 111 001 ≡ D1F9h

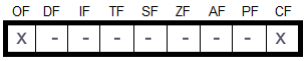
Example 2:
sar byte ptr [si+20],cl ≡ $M_{(si)+20}$ ← ($M_{(si)+20}$)>>(cl); update Flags;

Machine code:
11010010 (vw Md R/M) Disp. Low = 01 111 100 00010100 ≡ D27C14h

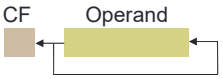
18

The 8086/88's Logic Instructions:

Shift Instructions:



- **Rotate Left: ROL**



rol operand,1 ≡ rotate operand left one bit; update Flags; if v=0
rol operand,cl ≡ rotate operand left (cl) bits; update Flags; if v=1

Example 1: rol bx,1 ≡ rotate (bx) left one bit; update Flags;

Machine code: 11010001 (vw Md R/M) 11 000 011 ≡ D1C3h

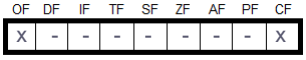
Example 2:
rol byte ptr [di-1],cl ≡ rotate ($M_{(di)-1}$) left (cl) bits; update Flags;

Machine code:
11010010 (vw Md R/M Disp. Low) 01 000 101 11111111 ≡ D245FFh

19

The 8086/88's Logic Instructions:

Shift Instructions:



- **Rotate Right: ROR**

ror operand,1 ≡ rotate operand right one bit; update Flags; if v=0
ror operand,cl ≡ rotate operand right (cl) bits; update Flags; if v=1

Example 1: ror dl,1 ≡ rotate (dl) right one bit; update Flags;

Machine code: 11010000 (vw Md R/M) 11 001 010 ≡ D0CAh

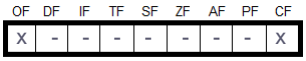
Example 2:
ror byte ptr [bx+21],cl ≡ rotate ($M_{(bx)+21}$) right (cl) bits; update Flags;

Machine code:
11010010 (vw Md R/M Disp. Low) 01 001 111 00010101 ≡ D24F15h

20

The 8086/88's Logic Instructions:

Shift Instructions:



- **Rotate Through Carry Left: RCL**

rcl operand,1 ≡ rotate CF:operand left one bit; update Flags; if v=0
rcl operand,cl ≡ rotate CF:operand left (cl) bits; update Flags; if v=1

Example 1: rcl si,1 ≡ rotate CF:(si) left one bit; update Flags;

Machine code: 11010001 (vw 11 Md 010 R/M 110) ≡ D1D6h

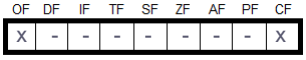
Example 2:
rcl byte ptr [si+2],cl ≡ rotate CF:($M_{(si)+2}$) left (cl) bits; update Flags;

Machine code:
11010010 (vw 01 Md 010 R/M 100) 00000010 (Disp. Low) ≡ D25402h

21

The 8086/88's Logic Instructions:

Shift Instructions:



- **Rotate Through Carry Right: RCR**

rcr operand,1 ≡ rotate CF:operand right one bit; update Flags; if v=0
rcr operand,cl ≡ rotate CF:operand right (cl) bits; update Flags; if v=1

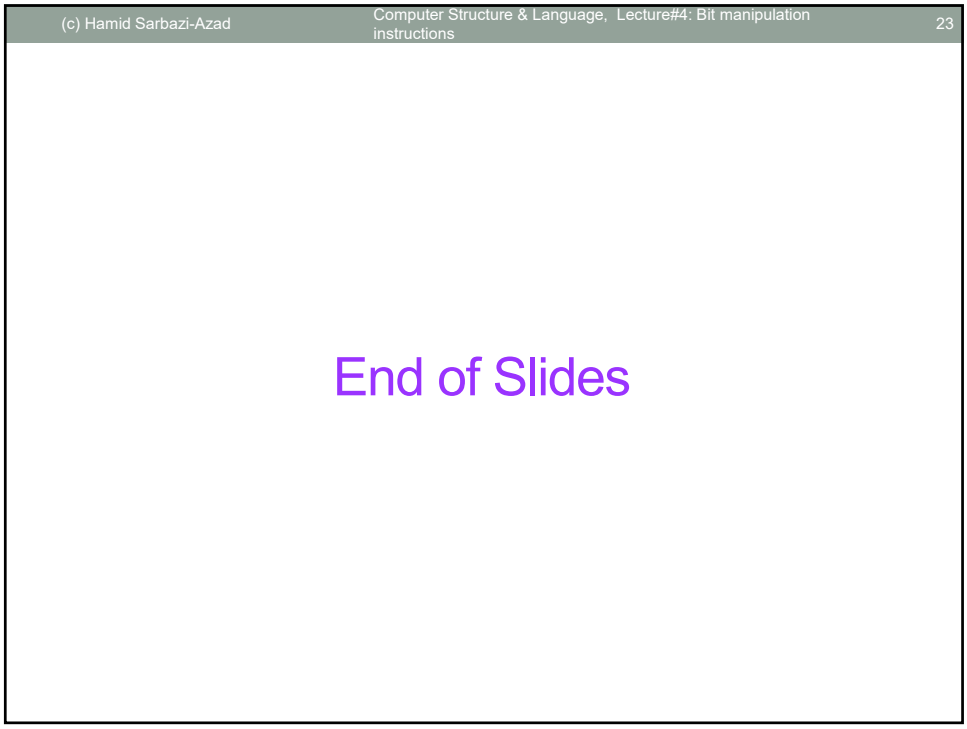
Example 1: rcr ax,1 ≡ rotate CF:(ax) right one bit; update Flags;

Machine code: 11010001 (vw 11 Md 011 R/M 000) ≡ D1D8h

Example 2:
rcr byte ptr [di],cl ≡ rotate CF:($M_{(di)}$) right (cl) bits; update Flags;

Machine code:
11010010 (vw 00 Md 011 R/M 101) == D21Dh

22


(c) Hamid Sarbazi-Azad
Computer Structure & Language, Lecture#4: Bit manipulation instructions
23
End of Slides

23