

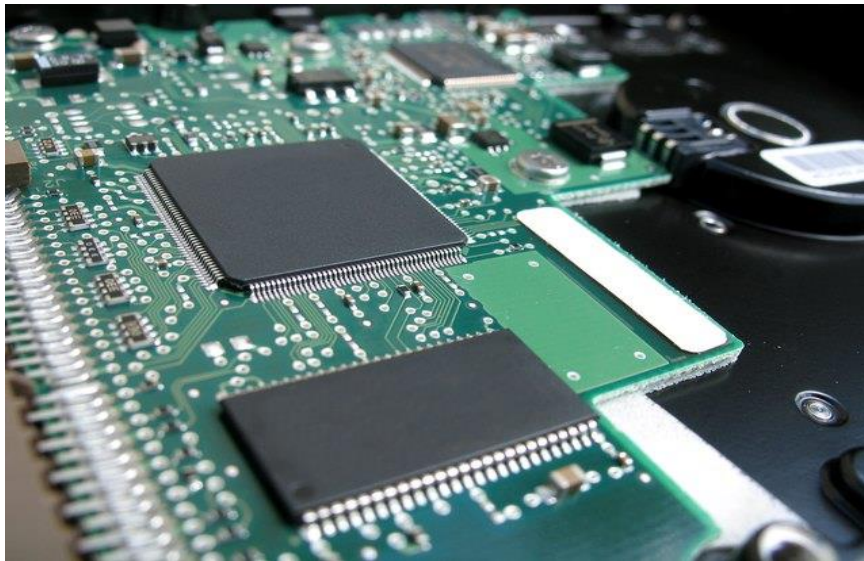


جامعة القادسية
كلية التربية



Lecture 10(part2)

Microprocessors



Prepared By:

Firas Abdulrahman Yosif

Logic Instructions (part 2)

من الايعازات البرمجية المهمة والتي تستخدم بكثرة داخل البرامج هي الايعازات المنطقية

Logic instruction والبوابات المنطقية الرئيسية المستخدمة هي AND, OR, XOR, NOT . في الجزء الثاني من هذا الموضوع سنتكلم عن بوابة XOR وبوابة NOT مع ذكر خصائصهما وامثلة لكل منهما. والجدول ادناه يوضح الصيغ العامة للايعازات المنطقية وطريقة عملها والاعلام flags التي تتأثر بهذه الايعازات.

■ The logic instructions include

- ❖ AND
- ❖ OR
- ❖ XOR (Exclusive-OR)
- ❖ NOT

Mnemonic	Meaning	Format	Operation	Flags affected
AND	Logical AND	AND D, S	$(S) \cdot (D) \rightarrow (D)$	OF, SF, ZF, PF, CF AF undefined
OR	Logical Inclusive-OR	OR D, S	$(S) \vee (D) \rightarrow (D)$	OF, SF, ZF, PF, CF AF undefined
XOR	Logical exclusive-OR	XOR D, S	$(S) \oplus (D) \rightarrow (D)$	OF, SF, ZF, PF, CF AF undefined
NOT	Logical NOT	NOT D	$(\text{NOT } D) \rightarrow (D)$	None

1. XOR Instruction

The general form: XOR destination, source

بوابة XOR تستخدم الصيغ التالية:

a) XOR register, register

في هذه الصيغة يتم استخدام سجل في الهدف وسجل في المصدر اي ممكن وضع في الهدف او في المصدر اي سجل من سجلات الـ general register وكذلك الـ pointers register ولكن ماعدا سجل الـ IP لا يمكن التعامل معه داخل ايعاز الـ XOR. بوابة الـ XOR تسمى (exclusive – OR) وتكون بين محتوى السجل الذي في المصدر وبين محتوى السجل الذي في الهدف ثم وضع ناتج العملية المنطقية في سجل الهدف (destination register).

ملاحظة: يجب ان يكون السجل نفس الحجم في المصدر وفي الهدف اي 8bit register مع 8bit وكذلك 16bit register مع 16bit register. وهذه بعض الامثلة على استخدام سجل مع سجل:

Examples:

XOR BX, SI

XOR AL, AH

XOR CL, AH

XOR SP, DI

XOR CH, DL

XOR SP, BP

$C = (A \oplus B)$		
A	B	C = A \oplus B
0	0	0
0	1	1
1	0	1
1	1	0

Truth table of XOR gate

لاحظ في الايعاز الاخير يتم عمل XOR بين محتوى السجل BP وبين محتوى السجل SP ثم وضع الناتج في السجل SP اي في الهدف (destination).

Ex) What's result for this program after executing it then explain status flags?

MOV CL, 0E4H

MOV DL, 08DH

XOR CL, DL

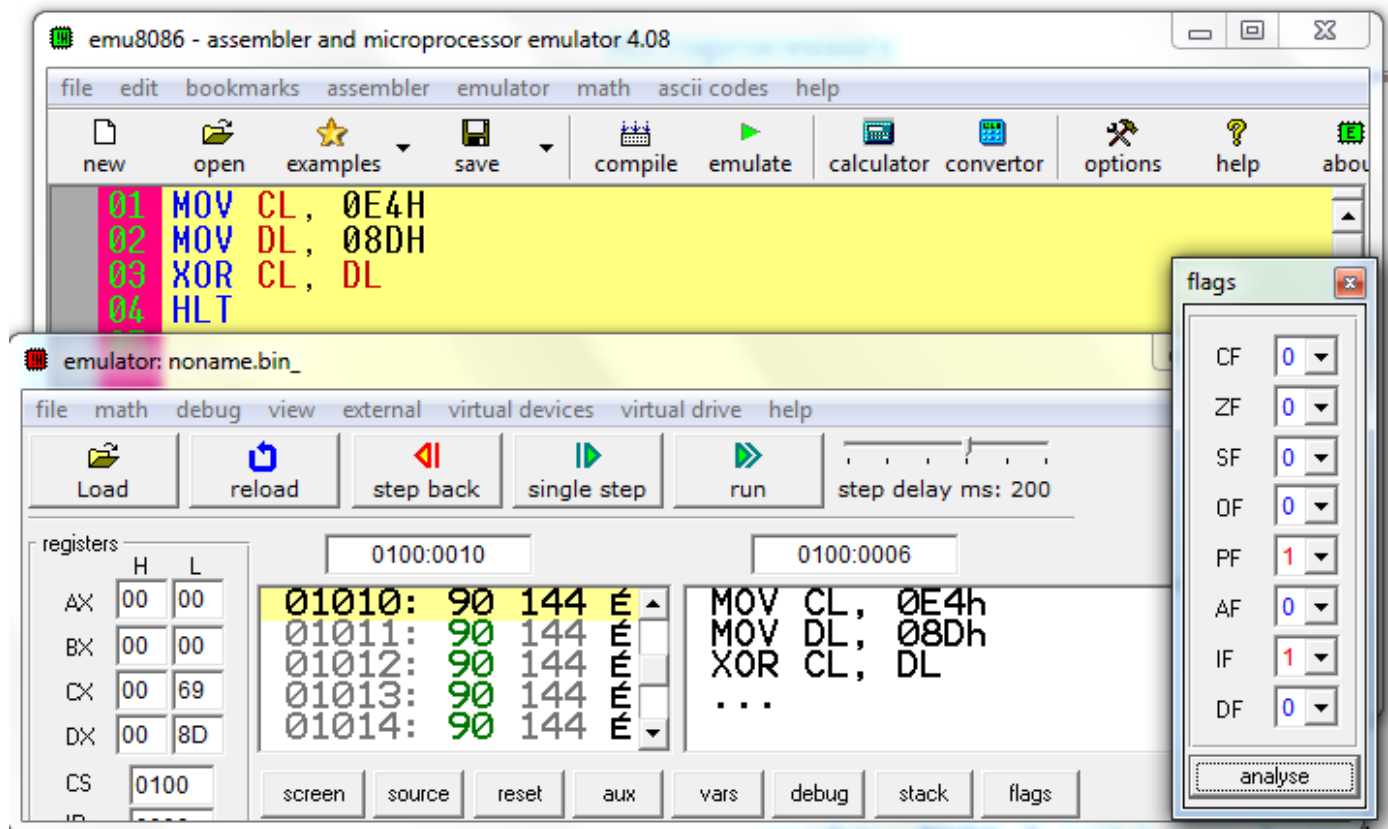
HLT

Solution)

CL= (69)₁₆ → (0000 0000 0110 1001)_b

CF=0, SF=0, OF=0, PF=1, AF=0, ZF=0

لاحظ تنفيذ البرنامج باستخدام الـ 8086 emulator :



هذه بعض الامثلة على الابعازات الخاطئة:

- XOR DS, BH (false)
التصحيح: XOR CL, BH
- XOR DL, DX (false)
التصحيح: XOR DL, DH
او XOR DX, DX

b) XOR register, data

في هذه الصيغة يتم استخدام سجل في الهدف وبيانات في المصدر ولايجوز وضع البيانات في الهدف. ملاحظة: لايجوز استخدام ال Segment register في ايعاز ال XOR .

هذه بعض الامثلة:

Examples)

XOR CL, (10011110)_b

XOR AX, (80)₁₆

XOR DI, 03367H

وهذه بعض الابعازات الخاطئة :

- XOR CX, SS (false)
التصحيح: XOR CX, AX
- XOR 4599H, SP (false)
التصحيح : XOR SP, 4599H

c) XOR Register, Memory

XOR Memory, Register

في هذه الصيغة يكون التعامل مع الذاكرة سواء كانت في المصدر او في الهدف ويمكن استخدام العنونة المباشرة (direct addressing mode) او العنونة الغير مباشرة (indirect addressing mode) داخل ايعاز الـ AND . لاحظ هذه اليعازات:

Examples)

XOR [DI], SP

XOR [388], AX

XOR CL, [BX]

Example) find contain of [2500] &[2501] after execute this program?

MOV SP, 04F96H

MOV [2500], 0D379H

XOR [2500], SP

HLT

Solution)

4F96	—————>	0100 1111 1001 0110	
D379	—————>	1101 0011 0111 1001	XOR
		<u>1001 1100 1110 1111</u>	

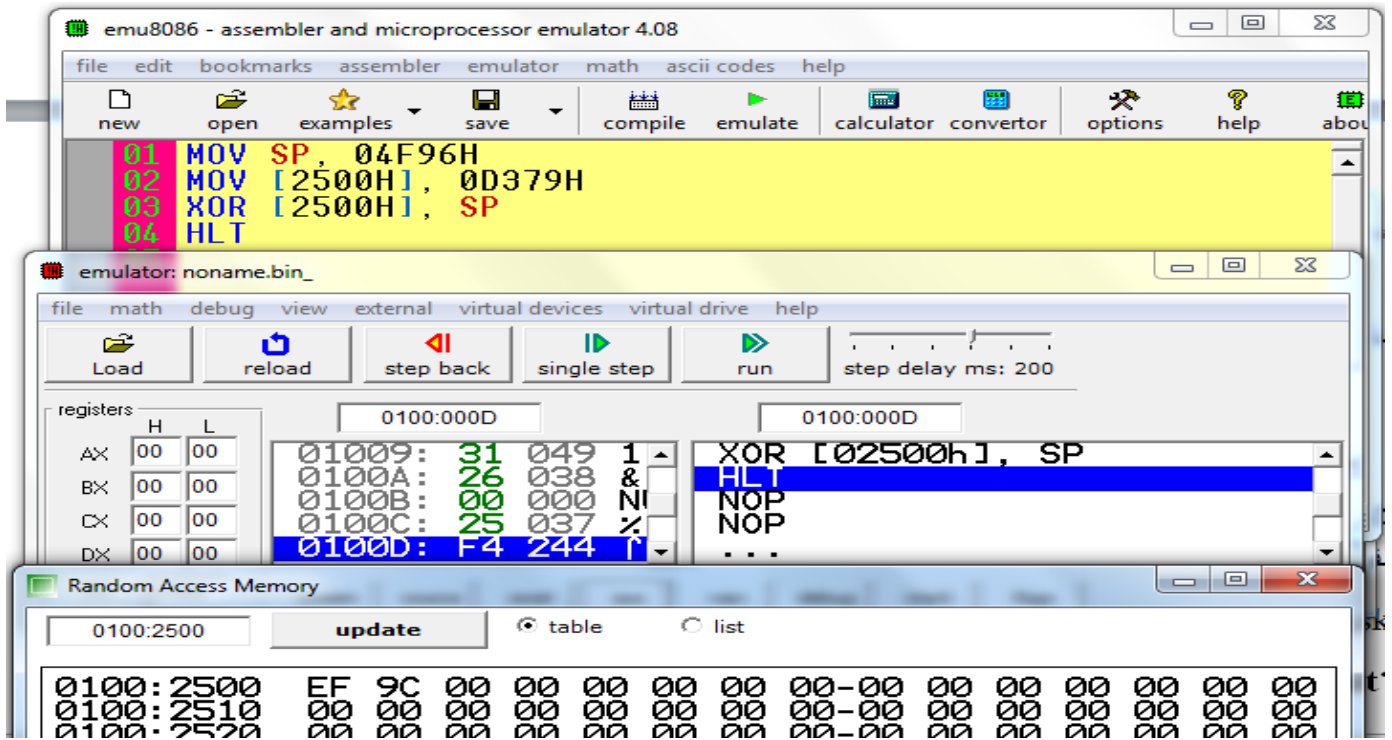
[2500]= 0EFH

[2501]= 09CH

Microprocessors

الشكل ادناه يوضح تنفيذ البرنامج باستخدام الـ 8086 emulator :

+



ملاحظة: تستخدم بوابة XOR لعكس (reverse) قيمة bit معين داخل البايٲ او في الـ word اي 0) الى 1 او 1 الى 0) مع بقاء بقية البتات كما هي وذلك عن طريق وضع واحد تحت البت المراد تغيير قيمته مع وضع صفر تحت بقية البتات . وكما موضح في المثال التالي:

Ex) Reverse b7 and b14 in SI=0DA48H using logic instructions, then write the result?

Sol.) SI= 1101 1010 0100 1000

$$\begin{array}{r}
 0100\ 0000\ 1000\ 0000 \\
 \hline
 1001\ 1010\ 1100\ 1000 \\
 \hline
 \text{SI} = 09\text{AC}8\text{H}
 \end{array}
 \quad \text{XOR}$$

2. NOT Instruction

The general form: **NOT** destination

عند استخدام بوابة **NOT** ممكن استخدام الصيغ التالية:

a) NOT register

في هذه الصيغة يتم استخدام سجل في الهدف ,اي ممكن وضع في الهدف اي سجل من سجلات الـ general register وكذلك الـ pointers register ولكن ماعدا سجل الـ IP لايمكن التعامل معه داخل ايعاز الـ **NOT** , وبوابة **NOT** تكافئ المتمم الاول الـ 1's complement .
ملاحظة: ممكن ان يكون السجل في الهدف من نوع register 8bit او register 16bit. وهذه بعض الامثلة على استخدام سجل مع سجل:

Examples:

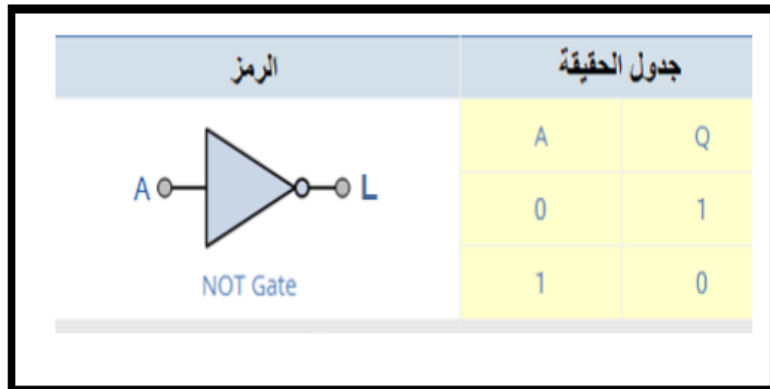
NOT BX

NOT CH

NOT AH

NOT DI

NOT SP



The diagram shows a NOT Gate symbol with input A and output L. To its right is a truth table with two columns: 'الرمز' (Symbol) and 'جدول الحقيقة' (Truth Table). The truth table has three rows: the first row shows input A and output Q; the second row shows input 0 and output 1; the third row shows input 1 and output 0.

الرمز	جدول الحقيقة
A	Q
0	1
1	0

Truth table of NOT gate

NOT BP

لاحظ في الايعاز الاخير تم عمل **NOT** على محتوى السجل BP ثم وضع الناتج في نفس السجل BP اي في الهدف.

Microprocessors

Ex) What's result for this program after execution it then explain status flags?

MOV BX, 02167H

MOV AX, 0C312H

NOT BX

NOT AX

HLT

Solution)

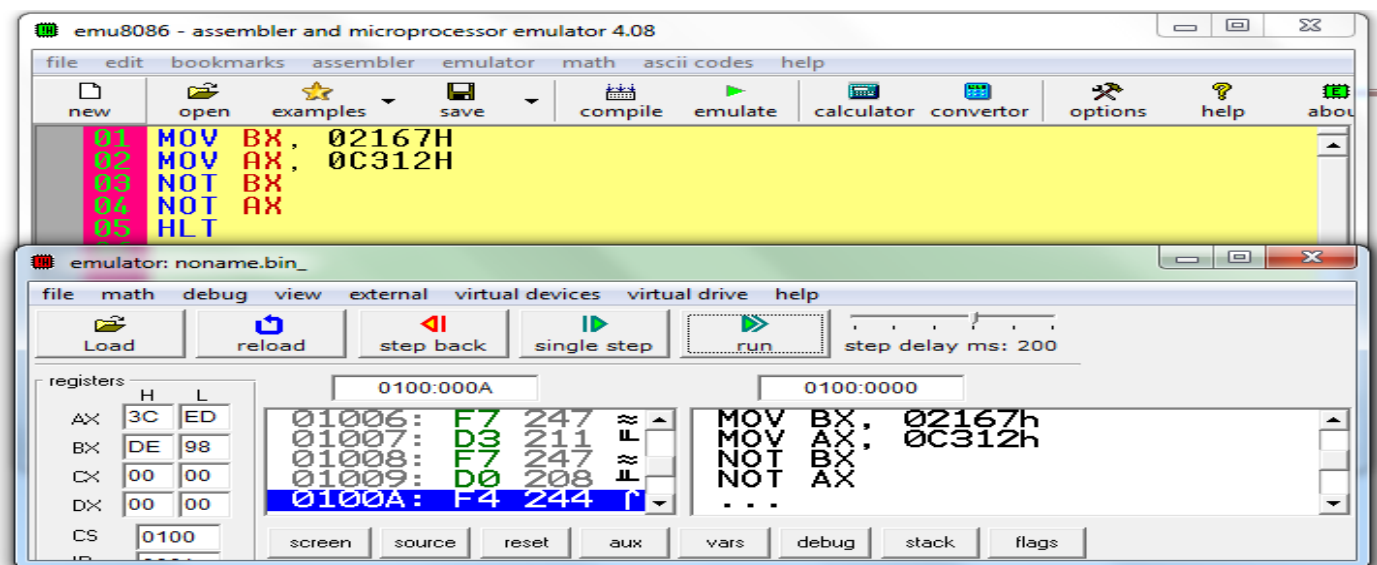
BX= 1101 1110 1001 1000

AX= 0011 1100 1110 1101

BX : CF=0, ZF=0, SF=1, OF=0, AF=0, PF=0

AX: CF=0, ZF=0, SF=0, OF=0, AF=0, PF=1

لاحظ تنفيذ البرنامج باستخدام الـ 8086 emulator :



ملاحظة: لايجوز استخدام البيانات بصورة مباشرة في صيغة NOT ولكن يتم استخدام البيانات عن طريق استخدام سجل وسطي. وهذه بعض البرامج على استخدام البيانات مع بوابة NOT :

EX1)

MOV AX, 2397H

NOT AX

HLT

EX2)

MOV CH, 24H

NOT CH

HLT

c) NOT Memory

في هذه الصيغة يكون التعامل مع الذاكرة في الهدف ويمكن استخدام العنوان المباشرة (direct addressing mode) او العنوان الغير مباشرة (indirect addressing mode) داخل ايعاز الـ NOT . لاحظ هذه اليعازات:

Examples)

NOT [DI]

NOT [88H]

NOT [BP]

Microprocessors

Example) find contain of [BX] after execute this program?

```
MOV BX, 0700H
```

```
MOV [BX], 089FCH
```

```
NOT W.[BX]
```

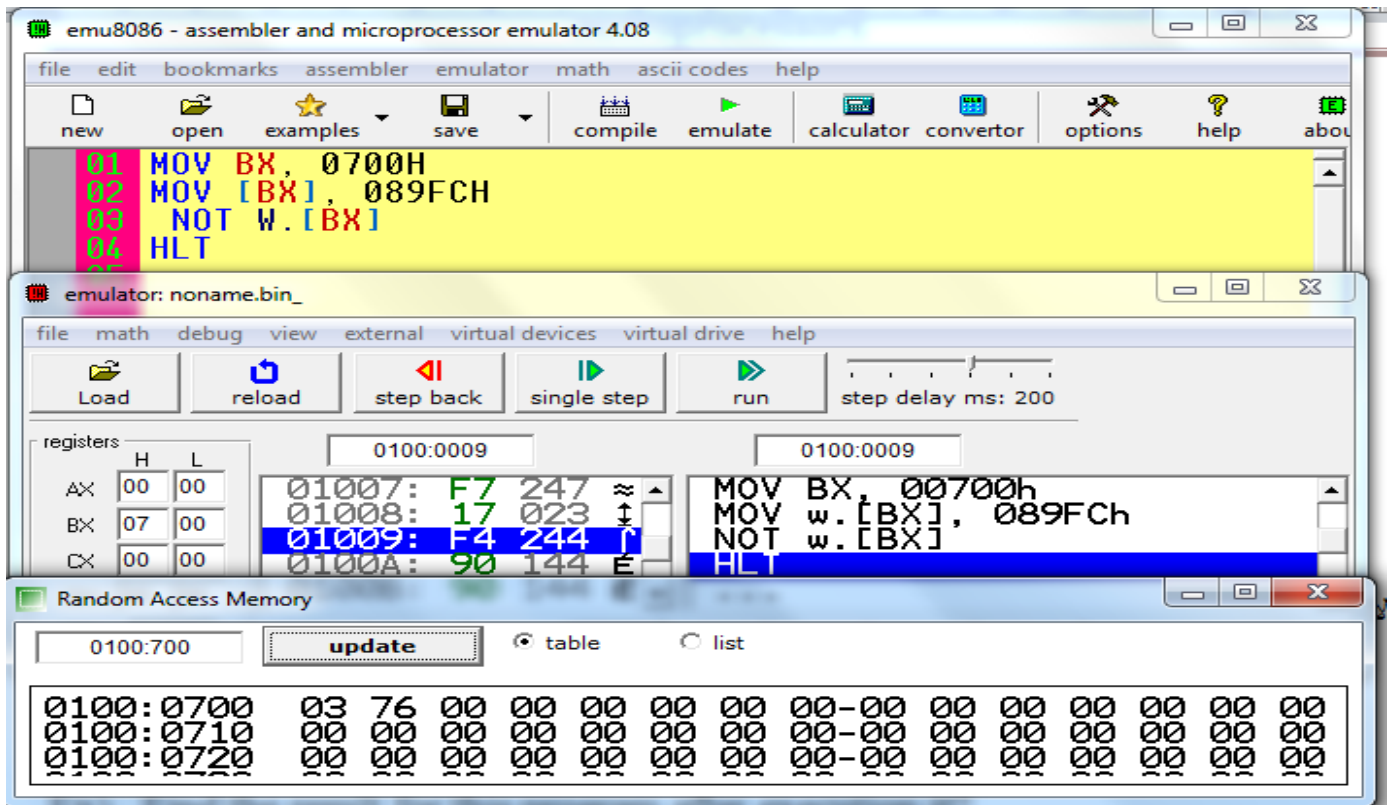
```
HLT
```

Solution)

[BX] → 0000 0011

[BX+1] → 0111 0110

- ملاحظة: في ايعاز الـ NOT تم استخدام الـ W للإشارة بأنه تم استخدام word اي 2byte من مواقع الذاكرة. في الشكل ادناه تم تنفيذ البرنامج باستخدام الـ 8086 emulator :



Microprocessors

Ex) Find the result for this program after execution it?

```
MOV BP, 3325H
```

```
MOV CX, 04E39H
```

```
MOV [BP], 056DAH
```

```
NOT [BP]
```

```
AND CX, [BP]
```

```
HLT
```

Solution) CX= 04621H

The screenshot displays the emu8086 emulator interface. The assembly code is as follows:

```
01 MOV BP, 3325H
02 MOV CX, 04E39H
03 MOV [BP], 056DAH
04 NOT [BP]
05 AND CX, [BP]
06 HLT
```

The emulator window shows the registers and the current instruction being executed:

Register	H	L
AX	00	00
BX	00	00
CX	46	21
DX	00	00

The current instruction being executed is:

```
01011: F4 244 ↑ NOT b.[BP] + 00h
01012: 90 144 E AND CX, [BP] + 00h
01013: 90 144 E ...
```

The Random Access Memory window shows the memory contents at the specified address:

Address	Value
0100:3325	25 56 00 00 00 00 00 00 00-00 00 00 00 00 00 00
0100:3335	00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00
0100:3345	00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00

Microprocessors

EX) Find contain of AX and [SI] for this program after execution it?

MOV SI, 2385H

MOV [SI], 9387H

MOV AX, 6821H

NOT b. [SI]

XOR AX, [SI]

AND AX, 5933H

HLT

Solution) AX= 5911H , [SI]= 78H

The screenshot shows an x86 emulator window titled "emulator: noname.bin_". The assembly code is displayed in a yellow-highlighted area:

```
01 MOV SI, 2385H
02 MOV [SI], 9387H
03 MOV AX, 6821H
04 NOT b. [SI]
05 XOR AX, [SI]
06 AND AX, 5933H
07 HLT
```

Below the code, the "registers" window shows the state of the CPU registers:

Register	H	L
AX	59	11
BX	00	00
CX	00	00
DX	00	00
CS	0100	

The "Random Access Memory" window shows the memory contents at address 0100:2385:

Address	Value
0100:2385	78 93 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100:2395	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00