Intel machine code

CSC 236

Machine code

- Microprocessor
 - Executes machine instructions
 - O Binary (hex) format
- Assembler source
 - Human-readable
 - Mnemonic representation of machine instructions
 - (And some more: data declaration, labels, directives)
- Assembler & linker
 - Create machine code

Two microprocessor architectures

CISC

- Complex instruction set computer
- 1960-1990
- Microcode
- Lots of function
- Irregular instructions
 - Size
 - Time
- Direct memory access

RISC

- Reduced instruction set computer
- 1990+
- Hardwired
- Few functions
- Fixed size, time
- Pipelining
- Superscalar
- Load-store architecture
 - No direct memory access

X86 is CISC





Basic rules

Details

Instruction length

- 80386 -- pentium
 - 32-bit addressing
 - O Machine instructions can be up to 15 bytes long
- Core
 - 64-bit addressing
 - Machine instructions can be up to 23 bytes long
- 8086
 - 16-bit addressing
 - Machine instructions can be up to 6 bytes long

code		[addr]	[disp]	[data]	
	opcode d w 6 11	mod reg r/m 2 3 3	disp disp low high	data data low high	

- Up to 6 bytes (minimum of 1)
- Four parts
 - Code
 - Addressing
 - Displacement for memory references
 - O Data for immediate values

code	[addr]	[disp]	[data]	
opcode d w	mod reg r/m	disp disp low high	data data	
6 1 1	2 3 3		low high	

- Code
 - Opcode defines instruction (6 bits = 64 unique opcodes)
 - d data direction (eg, reg from/to mem)
 - \circ w size: 1 \Rightarrow word; 0 \Rightarrow byte
- This one generic form of the code byte

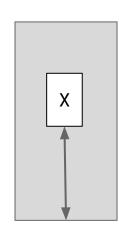
code	[addr]	[disp]	[data]	
opcode d w	mod reg r/m	disp disp low high	data data	
6 1 1	2 3 3		low high	

- addr byte
 - Two modes
 - Specifies operands
 - Extends opcode information
 - Optional
 - Some operands are implicit

- 3 fields
 - o mod Mode
 - o reg Register
 - o r/m Register or memory
- Tells
 - Operands in reg or memory
 - Extends opcode
 - Identifies a register

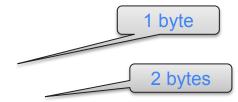
code	[addr]	[disp]	[data]	
opcode d w	mod reg r/m	disp disp low high	data data	
6 11	2 3 3		low high	

- disp displacement
 - Offset into memory
 - Always two bytes if present
 - Reverse byte order
 - O Eg, inc [X]



code		[addr]	[disp]	[data]	
	opcode d w 6 11	mod reg r/m 2 3 3	disp disp low high	data data low high	

- data immediate data
 - Immediate value
 - O Byte or word (1 or 2 bytes)
 - Eg, add al,5
 - Eg, add ax,3A7Bh



Summary of disp and data fields

[disp] if instruction has memory reference

o mov ax,[var]



[data] if instruction has immediate data

o mov ax, 1000



o mov al, 100



Both if instruction has memory reference and immediate data

o mov [var],1000



Largest instruction -- 6 bytes

Conversion

- Convert
 - ASM to machine
 - Machine to ASM
- Three tables
 - Table 1 general information
 - Table 2 ASM to machine
 - Table 3 machine to ASM

- mov ax, [var]
 - Assembler picks location for var
 - We don't know it (when we write)
 - Suppose var is at offset 12₁₀
 - O [disp] = 0C 00

- mov ax, [var]
 - O Look in Table 2
 - Several choices

mov

mov reg/mem to reg/mem
mov immed to mem
mov immed to reg
mov mem to accumulator
mov accumulator to mem
mov reg/mem to seg reg
mov seg reg to reg/mem

1000 10dw	mod reg r/m	disp-lo	disp-hi		
1100 011w	mod 000 r/m	disp-lo	disp-hi	data-lo	data-hi
1011 w reg	data-lo	data-hi			Water
1010 000w	disp-lo	disp-hi			
1010 001w	disp-lo	disp-hi			
1000 1110	mod 0 sr r/m	disp-lo	disp-hi		
1000 1100	mod 0 sr r/m	disp-lo	disp-hi		

- mov ax, [var]
 - O Look in Table 2
 - Several choices

mov

mov reg/mem to reg/mem 🗸

mov immed to mem

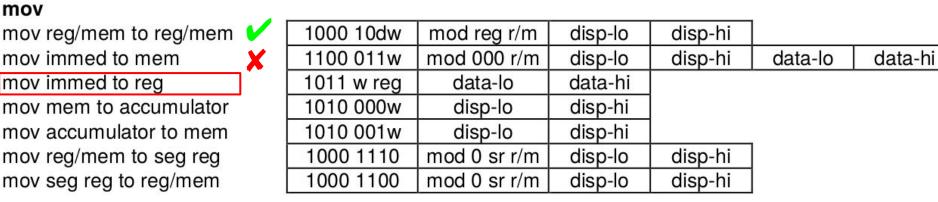
mov immed to reg mov mem to accumulator mov accumulator to mem mov reg/mem to seg reg mov seg reg to reg/mem

	1000 10dw	mod reg r/m	disp-lo	disp-hi		
	1100 011w	mod 000 r/m	disp-lo	disp-hi	data-lo	data-hi
	1011 w reg	data-lo	data-hi			
	1010 000w	disp-lo	disp-hi			
	1010 001w	disp-lo	disp-hi			
3	1000 1110	mod 0 sr r/m	disp-lo	disp-hi		
	1000 1100	mod 0 sr r/m	disp-lo	disp-hi		

- mov ax, [var]
 - Look in Table 2
 - Several choices

mov

mov immed to mem mov immed to reg mov mem to accumulator mov accumulator to mem mov reg/mem to seg reg mov seg reg to reg/mem



- mov ax, [var]
 - O Look in Table 2
 - Several choices

mov

mov reg/mem to reg/mem
mov immed to mem
mov immed to reg
mov mem to accumulator
mov accumulator to mem

mov reg/mem to seg reg mov seg reg to reg/mem

1000 10dw	mod reg r/m	disp-lo	disp-hi		
1100 011w	mod 000 r/m	disp-lo	disp-hi	data-lo	
1011 w reg	data-lo	data-hi			
1010 000w	disp-lo	disp-hi			
1010 001w	disp-lo	disp-hi			
1000 1110	mod 0 sr r/m	disp-lo	disp-hi		
1000 1100	mod 0 sr r/m	disp-lo	disp-hi		

data-hi

- mov ax, [var]
 - O Look in Table 2
 - Several choices

mov

mov reg/mem to reg/mem
mov immed to mem
mov immed to reg
mov mem to accumulator
mov accumulator to mem

mov	reg/mem to seg reg
mov	seg reg to reg/mem

				-	
1000 10dw	mod reg r/m	disp-lo	disp-hi		
1100 011w	mod 000 r/m	disp-lo	disp-hi	data-lo	data-hi
1011 w reg	data-lo	data-hi			a)
1010 000w	disp-lo	disp-hi			
1010 001w	disp-lo	disp-hi			
1000 1110	mod 0 sr r/m	disp-lo	disp-hi		
1000 1100	mod 0 sr r/m	disp-lo	disp-hi		

- mov ax, [var]
 - O Look in Table 2
 - Several choices

mov

mov reg/mem to reg/mem
mov immed to mem
mov immed to reg
mov mem to accumulator
mov accumulator to mem
mov reg/mem to seg reg

mov seg reg to reg/mem

	1000 10dw	mod reg r/m	disp-lo	disp-hi		
	1100 011w	mod 000 r/m	disp-lo	disp-hi	data-lo	data-hi
	1011 w reg	data-lo	data-hi	1,22		#\
	1010 000w	disp-lo	disp-hi			
	1010 001w	disp-lo	disp-hi			
9	1000 1110	mod 0 sr r/m	disp-lo	disp-hi		
	1000 1100	mod 0 sr r/m	disp-lo	disp-hi		

- mov ax, [var]
 - O Look in Table 2
 - Several choices

mov

mov reg/mem to reg/mem
mov immed to mem
mov immed to reg
mov mem to accumulator
mov accumulator to mem
mov reg/mem to seg reg
mov seg reg to reg/mem

 1000 10dw	mod reg r/m	disp-lo	disp-hi	N.	
1100 011w	mod 000 r/m	disp-lo	disp-hi	data-lo	data-hi
1011 w reg	data-lo	data-hi			21
1010 000w	disp-lo	disp-hi			
1010 001w	disp-lo	disp-hi			
1000 1110	mod 0 sr r/m	disp-lo	disp-hi		
1000 1100	mod 0 sr r/m	disp-lo	disp-hi		

- mov ax, [var]
 - O Look in Table 2
 - Several choices

mov

mov reg/mem to reg/mem
mov immed to mem
mov immed to reg
mov mem to accumulator
mov accumulator to mem
mov reg/mem to seg reg
mov seg reg to reg/mem

1000 10dw	mod reg r/m	disp-lo	disp-hi		
1100 011w	mod 000 r/m	disp-lo	disp-hi	data-lo	
1011 w reg	data-lo	data-hi			225
1010 000w	disp-lo	disp-hi			
1010 001w	disp-lo	disp-hi		_	
1000 1110	mod 0 sr r/m	disp-lo	disp-hi		
1000 1100	mod 0 sr r/m	disp-lo	disp-hi		

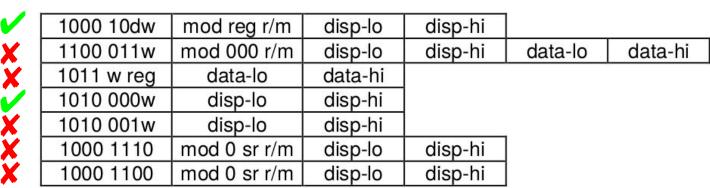
data-hi

- mov ax, [var]
 - Look in Table 2
 - Several choices

The accumulator is used more often than other registers. Eg, CBW, CWD, MUL, DIV ...

mov

mov reg/mem to reg/mem mov immed to mem mov immed to reg mov mem to accumulator mov accumulator to mem mov reg/mem to seg reg mov seg reg to reg/mem

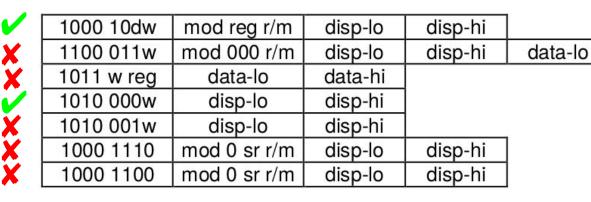


- mov ax, [var]
 - Look in Table 2
 - Several choices
 - 0 1010 000w

disp lo	disp hi
---------	---------

mov

mov reg/mem to reg/mem
mov immed to mem
mov immed to reg
mov mem to accumulator
mov accumulator to mem
mov reg/mem to seg reg
mov seg reg to reg/mem



data-hi

- mov ax, [var]
 - Look in Table 2
 - Several choices
 - O 1010 000w disp lo disp hi

- What is w?
 - \circ Size: 1 = word; 0 = byte
 - ax is word

- mov ax, [var]
 - Look in Table 2
 - Several choices

0 1010 000w	disp lo	disp hi
-------------	---------	---------

- O 1010 0001 0C 00
- O A1 0C 00

Example 1 a & b

mov al,[var]

mov ah,[var]

Example 1 a & b

mov al,[var]

1010 000w	disp lo	disp hi
1010 0000	0C	00

A0 0C 00

- Accumulator
 - \bigcirc Word = ax
 - O Byte = al (not ah)

mov

mov reg/mem to reg/mem
mov immed to mem
mov immed to reg
mov mem to accumulator
mov accumulator to mem
mov reg/mem to seg reg
mov seg reg to reg/mem

•					_	
	1000 10dw	mod reg r/m	disp-lo	disp-hi		
	1100 011w	mod 000 r/m	disp-lo	disp-hi	data-lo	data-hi
	1011 w reg	data-lo	data-hi			
	1010 000w	disp-lo	disp-hi			
	1010 001w	disp-lo	disp-hi			
	1000 1110	mod 0 sr r/m	disp-lo	disp-hi		
	1000 1100	mod 0 sr r/m	disp-lo	disp-hi		

mov ah,[var]

Example 1 a

mov al,[var]

1010 000w	disp lo	disp hi
1010 0000	9C	99

A0 0C 00

- Accumulator
 - O Word = ax
 - O Byte = al (not ah)

mov ah,[var]

Later

- Main memory
- Are these bytes
 - O Data or
 - O Instructions?

... 14 79 E3 2F 1A EE 68 ...

Main memory

... 14 79 E3 2F 1A EE 68 ...

- Are these bytes
 - O Data or
 - O Instructions?
 - Cannot determine
- Code segment memory

Main memory

... 14 79 E3 2F 1A EE 68 ...

- Are these bytes
 - Data or
 - O Instructions?
 - Cannot determine
- Code segment memory
 - Still cannot tell
 - Overrides allow code or data in any segment

Main memory

- ... 14 79 E3 2F 1A EE 68 ...
- Need to know code beginning
- Can you decode backwards?
 - Not in general
- Can you decode forwards?
 - O Better be able to!

Convert this machine code

O 80 06 0B 00 14 ...

<u>Table-3 8086 Instruction Decoding (Machine to Assembler</u>

push seg reg
pop seg reg
add reg/mem to reg/mem
add immed to accumulator
or reg/mem to reg/mem
or immed to accumulator
adc reg/mem to reg/mem
adc immed to accumulator

	- 2		
000 sr 110			
000 sr 111			
0000 00dw	mod reg r/m	disp-lo	disp-hi
0000 010w	data-lo	data-hi	
0000 10dw	mod reg r/m	disp-lo	disp-hi
0000 110w	data-lo	data-hi	
0001 00dw	mod reg r/m	disp-lo	disp-hi
0001 010w	data-lo	data-hi	

Look at code byte = 1000 0000

- Convert this machine code
 - O 80 65 0B 00 14 ...

<u>Table-3 8086 Instruction Decoding (Machine to Assembler</u>

push seg reg
pop seg reg
add reg/mem to reg/mem
add immed to accumulator
or reg/mem to reg/mem
or immed to accumulator
adc reg/mem to reg/mem
adc immed to accumulator

000 sr 110			
000 sr 111			
0000 00dw	mod reg r/m	disp-lo	disp-hi
0000 010w	data-lo	data-hi	
0000 10dw	mod reg r/m	disp-lo	disp-hi
0000 110w	data-lo	data-hi	
0001 00dw	mod reg r/m	disp-lo	disp-hi
0001 010w	data-lo	data-hi	

Look at code byte = 1000 0000

- Convert this machine code
 - O 80 06 0B 00 14 ...
- Table 3

add immed to reg/mem or immediate to reg/mem adc immed to reg/mem sbb immed from reg/mem and immed to reg/mem sub immed from reg/mem xor immed to reg/mem cmp reg/mem to immed

22	1000 000w	mod 000 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 001 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 010 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 011 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 100 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 101 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 110 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 111 r/m	disp-lo	disp-hi	data-lo	data-hi

- **Look at code byte =** 1000 0000
- $w = 0 \Rightarrow size is byte$

- Convert this machine code
 - 80 06 0B 00 14 ...
- Table 3

add immed to reg/mem
or immediate to reg/mem
adc immed to reg/mem
sbb immed from reg/mem
and immed to reg/mem
sub immed from reg/mem
xor immed to reg/mem
cmp reg/mem to immed

1000 000w	mod 000 r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod 001 r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod 010 r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod 011 r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod 100 r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod 101 r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod 110 r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod 111 r/m	disp-lo	disp-hi	data-lo	data-hi

Look at code byte = 1000 0000

- Convert this machine code
- Table 3

Now look at addr byte

add immed to reg/mem
or immediate to reg/mem
adc immed to reg/mem
sbb immed from reg/mem
and immed to reg/mem
sub immed from reg/mem
xor immed to reg/mem
cmp reg/mem to immed

25	1000 000w	mod 000 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 001 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 010 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 011 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 100 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 101 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 110 r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w	mod 111 r/m	disp-lo	disp-hi	data-lo	data-hi

Look at code byte = 1000 0000

Convert this machine code

○ 80 06 0<mark>8 00 14 ...</mark>

• Table 3

Now look at addr byte

add immed to reg/mem	
or immediate to reg/mem	
adc immed to reg/mem	
sbb immed from reg/mem	
and immed to reg/mem	
sub immed from reg/mem	
xor immed to reg/mem	
cmp rea/mem to immed	

1000 000w	mod	000	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	001	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	010	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	011	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	100	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	101	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	110	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	111	r/m	disp-lo	disp-hi	data-lo	data-hi
	1000 000w 1000 000w 1000 000w 1000 000w 1000 000w 1000 000w	1000 000w mod 1000 000w mod 1000 000w mod 1000 000w mod 1000 000w mod 1000 000w mod	1000 000w mod 001 1000 000w mod 010 1000 000w mod 011 1000 000w mod 100 1000 000w mod 101 1000 000w mod 110	1000 000w mod 001 r/m 1000 000w mod 010 r/m 1000 000w mod 011 r/m 1000 000w mod 100 r/m 1000 000w mod 101 r/m 1000 000w mod 110 r/m	1000 000w mod 001 r/m disp-lo 1000 000w mod 010 r/m disp-lo 1000 000w mod 011 r/m disp-lo 1000 000w mod 100 r/m disp-lo 1000 000w mod 101 r/m disp-lo 1000 000w mod 110 r/m disp-lo	1000 000w mod 001 r/m disp-lo disp-hi 1000 000w mod 010 r/m disp-lo disp-hi 1000 000w mod 011 r/m disp-lo disp-hi 1000 000w mod 100 r/m disp-lo disp-hi 1000 000w mod 101 r/m disp-lo disp-hi 1000 000w mod 110 r/m disp-lo disp-hi	1000 000w mod 001 r/m disp-lo disp-hi data-lo 1000 000w mod 010 r/m disp-lo disp-hi data-lo 1000 000w mod 011 r/m disp-lo disp-hi data-lo 1000 000w mod 100 r/m disp-lo disp-hi data-lo 1000 000w mod 101 r/m disp-lo disp-hi data-lo 1000 000w mod 110 r/m disp-lo disp-hi data-lo

Look at code byte = 1000 0000

- Convert this machine code
 - O 80 06 0<mark>8 00 14 ...</mark>
- Table 3
 - add immed to reg/mem

Now look at addr byte

add immed to reg/mem	
or immediate to reg/mem	
adc immed to reg/mem	
sbb immed from reg/mem	
and immed to reg/mem	
sub immed from reg/mem	
xor immed to reg/mem	
cmp reg/mem to immed	

1000 000w	mod	000	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	001	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	010	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	011	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	100	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	101	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	110	r/m	disp-lo	disp-hi	data-lo	data-hi
1000 000w	mod	111	r/m	disp-lo	disp-hi	data-lo	data-hi

- Convert this machine code
 - O 80 06 0B 00 14 ...
- Table 3
 - add immed to reg/mem

Now look at addr byte

0000 0110 = 00 000 110

OP	ERAND (S)	MOD	REG	R/M	DISP	IMM
1.	single register	11	special code	register		
2.	single memory	00	special code	110	mem addr	
3.	immediate to register	11	special code	register		data
4.	immediate to memory	00	special code	110	mem addr	data
	register and memory	00	register	110	mem addr	
6.	register and register	11	register_1	register_2		

- Convert this machine code
 - O 80 06 0B 00 14 ...
- Table 3
 - o add immed to reg/mem

Now look at addr byte

OPERAND(S)	MOD	REG	R/M	DISP	IMM
X1. single register	11	special code	register		10
single memory	00	special code	110	mem addr	
★3. immediate to register	11	special code	register		data
4. immediate to memory	00	special code	110	mem addr	data
5. register and memory	00	register	110	mem addr	
imes 6. register and register	11	register_1	register_2		

- Convert this machine code
 - O 80 06 0B 00 14 ...
- Table 3
 - o add immed to reg/mem

Now look at addr byte

OPERAND(S)	MOD	REG	R/M	DISP	IMM
X 1. single register	11	special code	register		
X2. single memory	00	special code	110	mem addr	
X3. immediate to register	11	special code	register		data
4. immediate to memory	00	special code	110	mem addr	data
5. register and memory	00	register	110	mem addr	
imes6. register and register	11	register_1	register_2		

- Convert this machine code
 - O 80 06 0B 00 14 ...
- Table 3
 - o add immed to reg/mem

Now look at addr byte

OPERAND(S)	MOD	REG	R/M	DISP	IMM
X 1. single register	11	special code	register		
X2. single memory	00	special code	110	mem addr	
X3. immediate to register	11	special code	register		data
\checkmark 4. immediate to memory	00	special code	110	mem addr	data
5. register and memory	00	register	110	mem addr	
imes6. register and register	11	register_1	register_2		

- Convert this machine code
 - O 80 06 0B 00 14 ...
- Table 3
 - o add immed to reg/mem

Now look at addr byte

OPERAND(S)	MOD	REG	R/M	DISP	IMM
X1. single register	11	special code	register		
X2. single memory	00	special code	110	mem addr	
X3. immediate to register	11	special code	register		data
\checkmark 4. immediate to memory	00	special code	110	mem addr	data
\times 5. register and memory	00	register	110	mem addr	
imes6. register and register	11	register_1	register_2		

- Convert this machine code
 - O 80 06 0B 00 14 ...
- Table 3
 - o add immed to reg/mem

Now look at addr byte

0000 0110 = 00 000 110

add immed to reg/mem

				t-	
1000 000w	mod 000 r/m	disp-lo	disp-hi	data-lo	data hi

OPERAND (S)	MOD	REG	R/M	DISP	IMM
X1. single register	11	special code	register		40
X2. single memory	00	special code	110	mem addr	
X3. immediate to registe	er 11	special code	register		data
4. immediate to memory	00	special code	110	mem addr	data
★ 5. register and memory	00	register	110	mem addr	
\times 6. register and registe	er 11	register 1	register 2		

Convert this machine code

○ 80 06 0B 00 14

- Table 3
 - add immed to reg/mem

Now look at addr byte

		ERAND (S)	MOD	REG	R/M	DISP IM	IM
X	1.	single register	11	special code	register		
X	2.	single memory	00	special code	110	mem addr	
		immediate to register	11	special code	register	da	ıta
V	4.	immediate to memory	00	special code	110	mem addr da	ıta
X	5.	register and memory	00	register	110	mem addr	
X	6.	register and register	11	register 1	register 2		

- Convert this machine code
 - O 80 06 0B 00 14 ...
- Table 3

 - add immed to reg/mem
 - add [var], 14h; var is at offset $000B = 11_{10}$

	OPI	ERAND (S)	MOD	REG	R/M	DISP	IMM
		single register	11	special code	register		
X	2.	single memory	00	special code	110	mem addr	
×	3.	immediate to register	11	special code	register		data
	A	respectively. The state of the	00	special code	110	mem addr	data
X	5.	register and memory	00	register	110	mem addr	
X	6.	register and register	11	register 1	register 2		

Now look at addr byte

- Convert
 - o mov dx,[var]
 - o var at offset 12₁₀ into data segment
- This is similar to example 1
 - mov ax,[var]
 - ax is *accumulator* has many shortcuts

- mov dx, [var]
 - O Look in Table 2
 - Several choices

mov

mov reg/mem to reg/mem mov immed to mem mov immed to reg mov mem to accumulator mov accumulator to mem mov reg/mem to seg reg mov seg reg to reg/mem

				-	
1000 10dw	mod reg r/m	disp-lo	disp-hi		
1100 011w	mod 000 r/m	disp-lo	disp-hi	data-lo	data-hi
1011 w reg	data-lo	data-hi			
1010 000w	disp-lo	disp-hi			
1010 001w	disp-lo	disp-hi			
1000 1110	mod 0 sr r/m	disp-lo	disp-hi		
1000 1100	mod 0 sr r/m	disp-lo	disp-hi		

- mov dx, [var]
 - O Look in Table 2
 - Several choices

mov

mov reg/mem to reg/mem
mov immed to mem
mov immed to reg
mov mem to accumulator
mov accumulator to mem
mov reg/mem to seg reg
mov seg reg to reg/mem

				-	
1000 10dw	mod reg r/m	disp-lo	disp-hi		
1100 011w	mod 000 r/m	disp-lo	disp-hi	data-lo	data-
1011 w reg	data-lo	data-hi			V
1010 000w	disp-lo	disp-hi			
1010 001w	disp-lo	disp-hi		_	
1000 1110	mod 0 sr r/m	disp-lo	disp-hi		
1000 1100	mod 0 sr r/m	disp-lo	disp-hi		

- mov dx, [var]
 - O Look in Table 2
 - Several choices only one will work

1000 10dw	mod reg r/m	disp-lo	disp-hi

mov

mov reg/mem to reg/mem
mov immed to mem
mov immed to reg
mov mem to accumulator
mov accumulator to mem
mov reg/mem to seg reg
mov seg reg to reg/mem

				-	
1000 10dw	mod reg r/m	disp-lo	disp-hi		
1100 011w	mod 000 r/m	disp-lo	disp-hi	data-lo	
1011 w reg	data-lo	data-hi		,	
1010 000w	disp-lo	disp-hi			
1010 001w	disp-lo	disp-hi		_	
1000 1110	mod 0 sr r/m	disp-lo	disp-hi		
1000 1100	mod 0 sr r/m	disp-lo	disp-hi		

data-hi

- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work
 - 1000 10dw | mod reg r/m | disp-lo | disp-hi

data size is word (dx)

⇒ w = 1

- mov dx, [var]
 - O Look in Table 2
 - O Several choices only one will work
 - o 1000 10d1 | mod reg r/m | disp-lo | disp-hi

- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work

0	1000 10d1	mod reg r/m	disp-lo	disp-hi
---	-----------	-------------	---------	---------

OPERAND (S)	MOD	REG	R/M	DISP	IMM
1. single register	11	special code	register		\$\frac{1}{2}
2. single memory	00	special code	110	mem addr	
3. immediate to register	11	special code	register		data
4. immediate to memory	00	special code	110	mem addr	data
5. register and memory	00	register	110	mem addr	
6. register and register	11	register 1	register 2		

- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work

0	1000 10d1	00 reg 110	disp-lo	disp-hi
---	-----------	------------	---------	---------

OPERAND (S)	MOD	REG	R/M	DISP	IMM
1. single register	11	special code	register		\$\frac{1}{2}
2. single memory	00	special code	110	mem addr	
3. immediate to register	11	special code	register		data
4. immediate to memory	00	special code	110	mem addr	data
5. register and memory	00	register	110	mem addr	
6. register and register	11	register 1	register 2		

- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work

0	1000 10d1	00 reg 110	disp-lo	disp-hi
---	-----------	------------	---------	---------

OPERAND(S)	MOD	REG	R/M	DISP	IMM
1. single register	11	special code	register		## ## ## ## ## ## ## ## ## ## ## ## ##
2. single memory	00	special code	110	mem addr	
3. immediate to register	11	special code	register		data
4. immediate to memory	00	special code	110	mem addr	data
5. register and memory	00	register	110	mem addr	
6. register and register	11	register 1	register 2		

- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work

0	1000 10d1	00 reg 110	disp-lo	disp-hi
---	-----------	------------	---------	---------

Table 1	Word	d	Byte
Table I	000	AX	000 AL
	001	CX	001 CL
	010	DX	010 DL
	011	ВХ	011 BL
	100	SP	100 AH
	101	BP	101 CH
	110	SI	110 DH
	111	DT	111 BH

- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work
 - o 1000 10d1 00 **010** 110 disp-lo disp-hi

Table 1	Word	Ĺ	Byte
Table I	000	AX	000 AI
	001	CX	001 CI
	010	DX	010 DI
	011	ВХ	011 BI
	100	SP	100 AF
	101	BP	101 CF
	110	SI	110 DF
	111	DI	111 BH

- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work

0	1000 10d1	00 010 110	disp-lo	disp-hi
---	-----------	------------	---------	---------

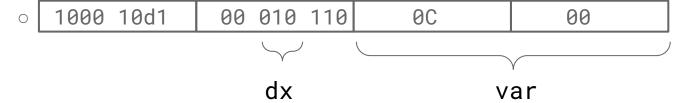
Displacement was given as 12₁₀. Hex = **00 OC**

- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work

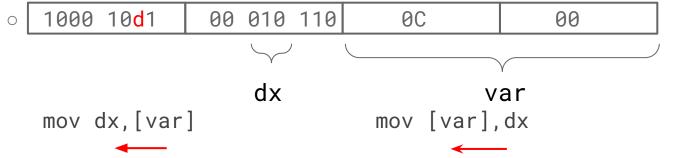
0	1000 10d1	00 010 110	0C	00
---	-----------	------------	----	----

Displacement was given as 12₁₀. Hex = 00 OC

- mov dx, [var]
 - Look in Table 2
 - O Several choices only one will work

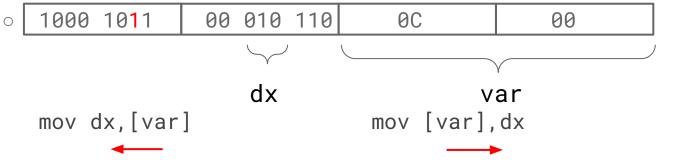


- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work



D Direction of data movement. Only needed for 2-operand instructions (register & memory) or (register & register). See OPERANDS 5 and 6. d = 0 REG field in ADDR is the source operand d = 1 REG field in ADDR is the destination operand

- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work

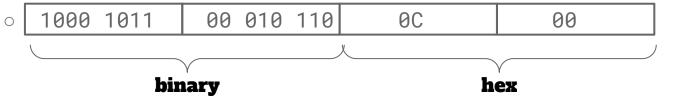


D Direction of data movement. Only needed for 2-operand instructions (register & memory) or (register & register). See OPERANDS 5 and 6. d = 0 REG field in ADDR is the source operand d = 1 REG field in ADDR is the destination operand

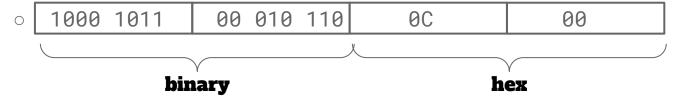
- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work

0	1000 1011	00 010 110	0C	00
---	-----------	------------	----	----

- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work

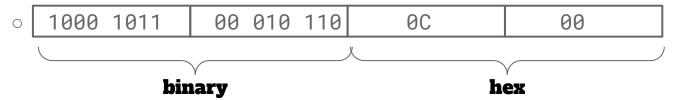


- mov dx, [var]
 - O Look in Table 2
 - O Several choices only one will work



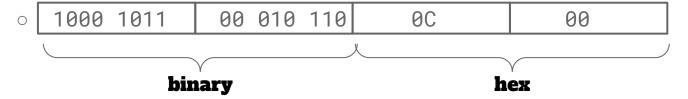
o 1000 1011 0001 0110 — regroup

- mov dx, [var]
 - O Look in Table 2
 - Several choices only one will work



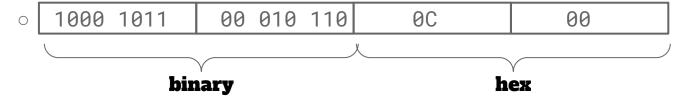
- 1000 1011 0001 0110 regroup
- 0 8

- mov dx, [var]
 - O Look in Table 2
 - Several choices only one will work



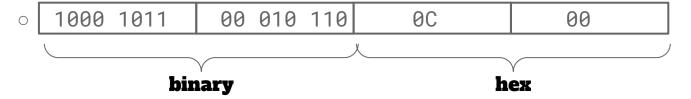
- 1000 1011 0001 0110 regroup
- o 8 B

- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work



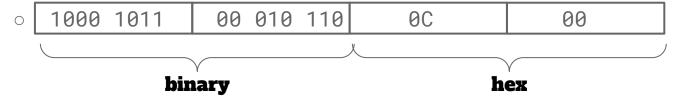
- 1000 1011 0001 0110 regroup
- o 8 B

- mov dx, [var]
 - Look in Table 2
 - Several choices only one will work



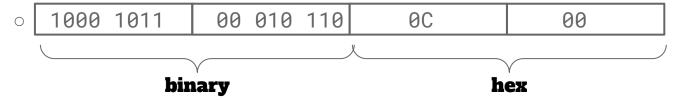
- 1000 1011 0001 0110 regroup
- o 8 B 1 6

- mov dx, [var]
 - Look in Table 2
 - O Several choices only one will work



- 1000 1011 0001 0110 regroup
- o 8B 16

- mov dx, [var]
 - O Look in Table 2
 - Several choices only one will work



- 1000 1011 0001 0110 regroup
- o 8B 16 0C 00

- Convert to machine code
 - o dec [var]
 - O Var is a byte at offset 1000₁₆

Try it.

- Convert to machine code
 - o dec [var]
 - O Var is a byte at offset 1000₁₆
- From table 2
 - o dec 8 bit reg / any size mem

1111 111w	mod 001 r/m	disp-lo	disp-hi
-----------	-------------	---------	---------

OPERAND(S)	MOD	REG	R/M	DISP	IMM
1. single register	11	special code	register		**
2. single memory	00	special code	110	mem addr	
immediate to re	egister 11	special code	register		data
4. immediate to me	emory 00	special code	110	mem addr	data
5. register and me	emory 00	register	110	mem addr	
6. register and re	egister 11	register_1	register_2		

- Convert to machine code
 - dec [var]
 - O Var is a byte at offset 1000₁₆
- From table 2
 - o dec 8 bit reg / any size mem

1111 111w	mod 001 r/m	disp-lo	disp-hi
-----------	-------------	---------	---------

OP	ERAND (S)	MOD	REG	R/M	DISP	IMM
1.	single register	11	special code	register		
2.	single memory	00	special code	110	mem addr	
3.	immediate to register	11	special code	register		data
4.	immediate to memory	00	special code	110	mem addr	data
5.	register and memory	00	register	110	mem addr	
6.	register and register	11	register_1	register_2		

- Convert to machine code
 - o dec [var]
 - O Var is a byte at offset 1000₁₆
- From table 2
 - o dec 8 bit reg / any size mem

0

1111 111w mod 001 r/m disp-lo disp-hi

OPERAND(S)	MOD	REG	R/M	DISP	IMM
1. single register	11	special code	register		**
2. single memory	00	special code	110	mem addr	
immediate to re	egister 11	special code	register		data
4. immediate to me	emory 00	special code	110	mem addr	data
5. register and me	emory 00	register	110	mem addr	
6. register and re	egister 11	register_1	register_2		

- Convert to machine code
 - dec [var]
 - Var is a byte at offset 1000₁₆
- From table 2
 - o dec 8 bit reg / any size mem

0	00	110			
 1111 111w	mod 0	01 r/m	disp-lo	disp-hi]

OPERAND(S)	MOD	REG	R/M	DISP	IMM
1. single register	11	special code	register		*
2. single memory	00	special code	110	mem addr	
3. immediate to register	11	special code	register		data
4. immediate to memory	00	special code	110	mem addr	data
5. register and memory	00	register	110	mem addr	
6. register and register	11	register 1	register 2		

- Convert to machine code
 - dec [var] FE 0E 00 10
 - O Var is a byte at offset 1000₁₆
- From table 2
 - o dec 8 bit reg / any size mem

0	00	110	00 ₁₆	10 ₁₆
1111 111w	mod 0	001 r/m	disp-lo	disp-hi

OPERAND(S)	MOD	REG	R/M	DISP	IMM
1. single register	11	special code	register		*
2. single memory	00	special code	110	mem addr	
3. immediate to register	11	special code	register		data
4. immediate to memory	00	special code	110	mem addr	data
5. register and memory	00	register	110	mem addr	
6. register and register	11	register_1	register_2		

- Convert
 - 0 24 24

Try it.

- Convert
 - 0 24 24
 - \circ 24₁₆ = 0010 0100₂
- Table 3

and reg/mem to reg/mem	0010 00dw	mod reg r/m	disp-lo	disp-hi
and immed to accumulator	0010 010w	data-lo	data-hi	
sub reg/mem from reg/mem	0010 10dw	mod reg r/m	disp-lo	disp-hi

Convert

0 24 24

 \circ 24₁₆ = 0010 0100₂

and reg/mem to reg/mem	0010 00dw	mod reg r/m	disp-lo	disp-hi
and immed to accumulator	0010 010w	data-lo	data-hi	
sub reg/mem from reg/mem	0010 10dw	mod reg r/m	disp-lo	disp-hi

Convert

$$\circ$$
 24₁₆ = 0010 0100₂

Table 3

and reg/mem to reg/mem	0010 00dw	mod reg r/m	disp-lo	disp-hi
and immed to accumulator	0010 010w	data-lo	dat x -hi	
sub reg/mem from reg/mem	0010 10dw	mod reg r/m	disp-lo	disp-hi

 \Rightarrow byte

Convert → and al,24h 24 24 $24_{16} = 0010 \ 0100_{2}$ \Rightarrow byte Table 3 and reg/mem to reg/mem 0010 00dw mod reg r/m disp-lo disp-hi and immed to accumulator dat -hi 0010 010w data-lo mod reg r/m 0010 10dw disp-hi sub reg/mem from reg/mem disp-lo