Machine instruction representation

the internal format of an instruction

[prefixes] + code + [Mode R/M] + [SiB] + [displacement] + [immediate]

(displacement → pointer! ; immediate → immediate value)

(SiB → scale, index, base)

	[prefixes]	code	[Mode R/M]	[SiB]	[displacement]	[immediate]
# bytes	0 - 4	1/2	0/1	0/1	0, 1, 2, 4	0, 1, 2, 4
		only mandatory				

prefixes can be from 0 up to 4 and occupy 1 byte each

code = the operation to be run

Structure of Mod R/M:

Mod	Reg/Op Code	R/M
7 6	5 4 3	2 1 0

SiB structure:

Scale	Index	Base
7 6	5 4 3	2 1 0

→ offset

max 8 values ⟷ 8 registers

} Must Know Struct.

```
mov eax, 17     ; has Mod R/M byte
                  and the immediate byte

mov eax, [v]    ; displacement
```

* displacement can happen on 8 bits in short jumps! but only by the processor

[Mode R/M] → | Mod | Reg/OpCode | R/M |

lines!

Table 2-2. 32-Bit Addressing Forms with the ModR/M Byte

r8(/r)	AL	CL	DL	BL	AH	CH	DH	BH
r16(/r)	AX	CX	DX	BX	SP	BP	SI	DI
r32(/r)	EAX	ECX	EDX	EBX	ESP	EBP	ESI	EDI
mm(/r)	MM0	MM1	MM2	MM3	MM4	MM5	MM6	MM7
xmm(/r)	XMM0	XMM1	XMM2	XMM3	XMM4	XMM5	XMM6	XMM7
/digit (Opcode)	0	1	2	3	4	5	6	7
REG =	000	001	010	011	100	101	110	111

Effective Address	Mod	R/M	Value of ModR/M Byte (in Hexadecimal)							
[EAX]	00	000	00	08	10	18	20	28	30	38
[ECX]		001	01	09	11	19	21	29	31	39
[EDX]		010	02	0A	12	1A	22	2A	32	3A
[EBX]		011	03	0B	13	1B	23	2B	33	3B
[--][--][1]		100	04	0C	14	1C	24	2C	34	3C
disp32[2]		101	05	0D	15	1D	25	2D	35	3D
[ESI]		110	06	0E	16	1E	26	2E	36	3E
[EDI]		111	07	0F	17	1F	27	2F	37	3F
disp8[EAX][3]	01	000	40	48	50	58	60	68	70	78
disp8[ECX]		001	41	49	51	59	61	69	71	79
disp8[EDX]		010	42	4A	52	5A	62	6A	72	7A
disp8[EBX];		011	43	4B	53	5B	63	6B	73	7B
disp8[--][--]		100	44	4C	54	5C	64	6C	74	7C
disp8[EBP]		101	45	4D	55	5D	65	6D	75	7D
disp8[ESI]		110	46	4E	56	5E	66	6E	76	7E
disp8[EDI]		111	47	4F	57	5F	67	6F	77	7F
disp32[EAX]	10	000	80	88	90	98	A0	A8	B0	B8
disp32[ECX]		001	81	89	91	99	A1	A9	B1	B9
disp32[EDX]		010	82	8A	92	9A	A2	AA	B2	BA
disp32[EBX]		011	83	8B	93	9B	A3	AB	B3	BB
disp32[--][--]		100	84	8C	94	9C	A4	AC	B4	BC
disp32[EBP]		101	85	8D	95	9D	A5	AD	B5	BD
disp32[ESI]		110	86	8E	96	9E	A6	AE	B6	BE
disp32[EDI]		111	87	8F	97	9F	A7	AF	B7	BF
EAX/AX/AL/MM0/XMM0	11	000	C0	C8	D0	D8	E0	E8	F0	F8
ECX/CX/CL/MM/XMM1		001	C1	C9	D1	D9	E1	E9	F1	F9
EDX/DX/DL/MM2/XMM2		010	C2	CA	D2	DA	E2	EA	F2	FA
EBX/BX/BL/MM3/XMM3		011	C3	CB	D3	DB	E3	EB	F3	FB
ESP/SP/AH/MM4/XMM4		100	C4	CC	D4	DC	E4	EC	F4	FC
EBP/BP/CH/MM5/XMM5		101	C5	CD	D5	DD	E5	ED	F5	FD
ESI/SI/DH/MM6/XMM6		110	C6	CE	D6	DE	E6	EE	F6	FE
EDI/DI/BH/MM7/XMM7		111	C7	CF	D7	DF	E7	EF	F7	FF

memory addressed operand

Reg/Mem Reg

ADC (10)

Eb Gb ← byte

Ev Gv ← word (11)

Gb Eb (12)

Gv Ev (13)

ex:

B8	04000000	Mov EAX, 4 ✓
BA	78563412	Mov EDX, 12345678 ✓
50		PUSH EAX ✓
52		PUSH EDX ✓
B9	20000000	Mov ECX, 20 ✓
FF35	05104000	PUSH DWORD PTR [DS:401...] X NOT correct

83 C4 10 ADD ESP, 10

B8 MOV EAX iv

* all imediate values are stored on 64 bits

iv ← immediate value

BA MOV EDX iv

50 PUSH EAX

52 PUSH EDX

B9 MOV ECX iv

FF 35

81 C4 ADD Ev Iv

Mod R/M byte

C4 = 11 000 100

Mod | Reg/opCode | R/M

* Examen trebuie stiut semnificatia fiecarui câmp

Mode R/M ex:

9D h = 10011100 → instruction [EBP] + disp. 32, EBX

Mod | Reg/OpCode | R/M

R/M → EBX

Reg/OpCode → EBP

Mod → memory disp. 32

2A h = 00 101 010 → instruction [EDX], EBP (or CH, or BP)

Mod | Reg/opCode | R/M

Reg/opCode → EBP

R/M → EDX

* Hw: F3 h

SiB

9C h = 10 011 100 → memory address operand offset spec. form

ESP

Scale = 4

EBX * 4

[ESP + EBX * 4]