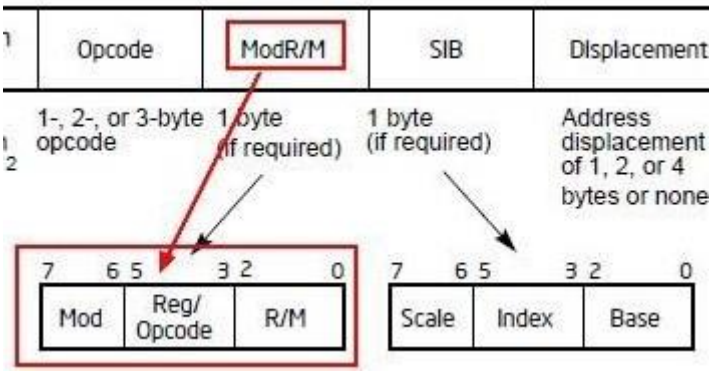
C14 - -01_Reprezentarea instr. masina - coloana 2 OllyDbg,

- 02_coduri de instructiuni (SetOpcode Table) + 02_BIS_Opcode_Extension Table

-03_Addressing Methods Codes - lămureşte cum să înţelegem OPERANZII instrucţiunilor din tabelele SetOpcode Table

-04 + 05 - Mod R/M byte + SIB bytes – lămuresc modul de CODIFICARE în cadrul unei instrucţiuni (vezi coloana 2 din OllyDbg) al octeţilor Mod R/M şi SIB (structura lor este detaliată în 01_Reprezentarea instr. masina) – tabelele 2.2 (Mod R/M byte) şi 2.3 (SIB byte)



Cum să citim tabelul corespunzător octetului ModR/M:

The second and third columns in Tables 2-1 and 2-2 gives the binary encodings of the Mod and R/M fields in the ModR/M byte, respectively, required to obtain the associated effective address listed in the first column. All 32 possible combinations of the Mod and R/M fields are listed.

Mod - 2 biţi - sunt 4 valori posibile, prezentate pe COLOANA 2 (00, 01, 10, 11) – acest câmp indică moduri de combinaţie de tipuri de operanzi: 00 – "memorie (doar bază), registru" sau "registru, memorie(doar bază), 01 - "memorie (bază+disp8), registru" sau invers, 10 - "memorie (bază+disp32), registru" sau invers, 11 – "registru, registru"

Reg/Opcode - 3 biţi – DACA E REGISTRU sunt 8 valori posibile (0-7, 000-111) prezentate pe LINIILE 6 şi 7 - din prima linie de sus a tabelului (REG=) – acest element **ALEGE AL DOILEA OPERAND - cel din dreapta, adică SURSA !!!** (this denotes the second operand register if an instruction requires one !)

R/M - 3 biti - sunt 8 valori posibile (000-111) prezentate pe COLOANA 3 - **ALEGE PRIMUL OPERAND - cel din stânga, adica SURSA !!! (the first operand)**

Combinatiile celor 3 configuraţii de biti furnizeaza valoarea octetului R/M - prezentat in tabel in sectiunea din dreapta - **Value of Mod R/M Byte (in Hexadecimal)**

The ModR/M byte follows many opcodes to specify the addressing mode. This byte is fairly complicated but I'll try to explain it in this section. The diagram below shows how the byte is split into three fields**: mod selects the overall mode, reg selects a register, and r/m selects either a register or memory mode.**

Sa luam câteva exemple de valoare de octet Mod R/M (32-Bit Addressing Forms with the ModR/M Byte !!!!!!) si sa incercam sa descifrăm ce reprezintă:

a). mov [ebp+a], ebx ; 89**9D** 02204000 **MOV Ev,Gv (89)**

9Dh = 1001 1101 = 10 011 101 (Mod R/M)

Deci Mod = **10** - deci ma uit pe linia coresp. valorii 10 pt MOD ([...] + disp32)

Reg/Opcode = **011** - gasesc valoarea 011 pe coloana 4 a tabelului din dreapta (BL, BX, **EBX**, ...), care este alegerea depinde de tipul instructiunii (byte, word, dword - dat de octetul OPCODE). – **al DOILEA** operand

r/M = 101 - identifica linia [EBP]+disp32 din sectiunea Mod=10 (acesta este PRIMUL OPERAND - DESTINATIA !!!!)

Deci daca Mod R/M = 9Dh, setul de operanzi va fi:

instructiune [EBP]+disp32, EBX adica o instructiune de tipul "instr memorie, registru"

E - A ModR/M byte follows the opcode and specifies the operand. The operand is either a general-purpose register or a memory address. If it is a memory address, the address is computed from a segment register and any of the following values: a base register, an index register, a scaling factor, a displacement.

G - The reg field of the ModR/M byte selects **a general register** (for example, AX (000), EBX(011) etc.)

V - Word or doubleword, depending on operand-size attribute.

b). mov [edx],ebp ; 89**2A** **MOV Ev,Gv (89)**

2Ah - 0010 1010 = 00 101 010 (Mod R/M)

Deci Mod = **00** - deci ma uit pe linia coresp. valorii 00 pt MOD ([...] - fara deplasamente) = **[reg]**

Reg/Opcode = 101 - gasesc valoarea 101 pe coloana 6 a tabelului din dreapta (CH, BP, EBP, ...), care este exact alegerea depinde de tipul instructiunii (byte, word, dword - dat de octetul OPCODE).

r/M = 010 - identifica linia [EDX] din sectiunea Mod=00 (acesta este PRIMUL OPERAND - DESTINATIA !!!!)

Deci daca Mod R/M = 2Ah, setul de operanzi va fi:

instructiune [EDX], EBP (sau CH, sau BP) - adica tot o instructiune de tipul "instr memorie, registru"

c). mov [edx+17], ecx = 894A 11 89 = MOV Ev,Gv (memory address, general register)

4Ah = 0100 1010 = 01 001 010 (Mod R/M)

Mod = 01 = [reg] + disp8

Reg/Opcode = 001 = ECX (operandul sursă – al DOILEA operand !) [EDX + disp8]

r/M = 010 - identifică linia [EDX] + disp8 din sectiunea Mod=01 (acesta este PRIMUL OPERAND - DESTINATIA !!!!)

In codificarea intructiunii mai apare in final 11 = 17 (in baza 10) – "The disp8 nomenclature denotes an 8-bit displacement that follows ModR/M byte"

.

d). mov ebx, esi ; 89F3

F3h = 1111 0011 = 11 110 011 (mod R/M)

Deci **Mod = 11** - deci mă uit pe linia coresp. valorii 11 pt MOD - aici este vorba despre un operand de tip REGISTRU si NU din memorie !

Reg/Opcode = 110 - gasesc valoarea 110 pe coloana 7 a tabelului din dreapta (DH, SI, ESI, ...), care este exact alegerea depinde de tipul instructiunii (byte, word, dword - dat de octetul OPCODE).

r/M = 011 - identifica linia EBX/BX/BL din sectiunea Mod=11 (acesta este PRIMUL OPERAND - DESTINATIA !!!!)

Deci daca Mod R/M = F3h, setul de operanzi va fi:

instructiune EBX (sau BX sau BL), ESI (sau SI, sau DH) - adica o instructiune de tipul "registru, registru"

e). mov ecx, [ebx] ; 8B0B MOV Gv, Ev (8B) mov registru, memorie

0B = 00 001 011 (ModR/M)

Deci Mod = 00 - deci ma uit pe linia coresp. valorii 00 pt MOD ([...] - fara deplasamente)

Reg/Opcode = 001 - gasesc valoarea 001 pe coloana 2 a tabelului din dreapta (CL, CX, ECX, ...), care este exact alegerea depinde de tipul instructiunii (byte, word, dword - dat de octetul OPCODE). Acesta este PRIMUL OPERAND (destinatia).

G - The reg field of the ModR/M byte selects a general register (for example, CX (001))

r/M = 011 - identifica linia [EBX] din sectiunea Mod=00 (acesta este AL DOILEA OPERAND - SURSA !!!!)

f). mov ecx, [esp + ebx*4] ; 8B0C9C MOV Gv, Ev (8B) mov registru, memorie

0C = 00 001 100 (mod R/M byte)

- similar cu cee ace este mai sus , mai putin câmpul r/M:

r/M = **100** - identifica linia [EBX] din sectiunea Mod=00 (1. The [--][--] nomenclature means a SIB follows the ModR/M byte). -

- SIB_Byte Values - Structura lui este SS (SCALE - 2 biti) Index (3 biti) Base (3 biti)

9Ch = 1001 1100 = 10 011 100 (SIB Byte)

SS = 10 (factor de scalare = 4) - vezi coloana SS din tabel si lista de formule de adresare disponibila de pe prima coloana

Index = 011 (vezi coloana 3 din linia coresp. Lui SS=10) deci e vorba despre expresia de index [EBX*4]

Base = 100 (vezi linia 1) deci registrul de baza este ESP

Ca urmare formula de calcul al offsetului este **[ESP + EBX*4]**

Ca verificare, cautam in tabelul **Table 2-3. 32-Bit Addressing Forms with the SIB Byte valoarea 9C si o găsim la intersecţia liniei** [EBX*4] din coloanele SS=10 si Index = 011 si in cadrul coloanei ESP (de valoare 4 = 100)

g). mov ecx, [esp + ebx*4 + a - 7] ; 8B8C9C FB1F4000 MOV Gv, Ev (8B) mov registru, memorie

(mov ecx, dword ptr SS:[ebx*4 + esp + 00401FFB]) 00402002 – 7 = 00401FFB

8C = 10 001 100 (mod R/M byte)

Deci Mod = 10 - deci ma uit pe linia coresp. valorii 10 pt MOD ([...] + disp32)

Reg/Opcode = 001 - gasesc valoarea 001 pe coloana 2 a tabelului din dreapta (CL, CX, ECX, ...). Acesta este PRIMUL OPERAND (destinatia) = **ECX**

r/M = **100** - identifica din secţiunea Mod=10 (1. The [--][--]+disp32 nomenclature means a SIB follows the ModR/M byte).

9C = SIB byte (same as above) FB1F4000 = disp32 (= a-7 = 00401FFB)