24. x86 Instruction Prefix Bytes

- x86 instruction can have up to 4 prefixes.
- Each prefix adjusts interpretation of the opcode:

String manipulation instruction prefixes (prefixes provided EXPLICITLY by the programmer !)

F3h = REP, REPE
F2h = REPNE

where

- REP repeats instruction the number of times specified by *iteration count* ECX.
- REPE and REPNE prefixes allow to terminate loop on the value of **ZF** CPU flag.

- 0xF3 is called REP when used with MOVS/LODS/STOS/INS/OUTS (instructions which don't affect flags)
  0xF3 is called REPE or REPZ when used with CMPS/SCAS
  0xF2 is called REPNE or REPNZ when used with CMPS/SCAS, and is not documented for other instructions.
  Intel's insn reference manual REP entry only documents F3 REP for MOVS, not the F2 prefix.

Instruction prefix - REP MOVSB F3:A4

Related string manipulation instructions are:

- MOVS, move string ; STOS, store string
- SCAS, scan string ; CMPS, compare string, etc.

See also string manipulation sample rogram: rep_movsb.asm

Segment override prefix causes memory access to use *specified segment* instead of *default segment* designated for instruction operand.
(these prefixes are provided EXPLICITLY by the programmer).

2Eh = CS	**26h = ES**
36h = SS	**64h = FS**
3Eh = DS	**65h = GS**

---

26 : D7
Segment override prefix - ES xlat
(ES:EBX)

mov eax, [ebx] - 8B03 **mov eax, [SS:ebx] – 36:8B03**

mov ax, [DS:ebx] – implicit prefixing with 66 3E : 66 : 8B03
- **Explicit prefixing by the programmer with DS**

(3E – prefix DS, 66 – prefix op.size, 8B – cod mov Gv, Ev, 03 – EBX)

mov eax, [CS:ebx] – explicit prefixing by the progr. with CS 2E : 8B03
(2E – prefix CS, 8B – cod mov Gv, Ev, 03 – EBX)

ES LODSB – 26 : AC ; LODS byte ptr ES:[ESI]
ES CMPSB - 26 : A6 ; CMPS byte ptr ES:[ESI], byte ptr ES:[EDI]

ES STOSB - 26 : AA ; STOS byte ptr ES:[EDI] – superfluous segment override prefix

MOVSB - A4 ; MOVS byte ptr ES:[EDI], byte ptr DS:[ESI]
ES MOVSB - 26 : A4 ; MOVS byte ptr ES:[EDI], byte ptr ES:[ESI]

ES SCASB - 26 : AE ; SCAS byte ptr ES:[EDI] - superfluous segment override prefix

Operand override, 66h. Changes size of data expected by default mode of the instruction e.g. 16-bit to 32-bit and vice versa.

Address override, 67h. Changes size of address expected by the default mode of the instruction. 32-bit address could switch to 16-bit and vice vs

These two last prefix types appear as a result of some particular ways of using the instructions (examples below), which will cause the generation of these prefixes by the assembler in the internal format of the instruction. These prefixes are NOT provided EXPLICITLY by the programmer by keywords or reserved words of the assembly language.

Operand size prefix – 0x66

Bits 32 - default mode of the below code

cbw ; 66:98 - because rez is on 16 bits (AX)
cwd ; 66:99 - because rez is composed by 2 reg on 16 bits (DX:AX)
cwde ; 98 - because we follow the default mode on 32 biti – rez in EAX
push ax ; 66:50 – because a 16 bits value is loaded onto the stack, the stack being organized on 32 bits
push eax ; 50 - ok – consistent usage with default mode
mov ax, a ; 66:B8 0010 – because rez is on 16 bits

Bits 16 - default mode of the below code

cbw ; 98 - ok, because result is on 16 bits (AX)
cwd ; 99 - ok, because result is composed by a combination of 2 registers on 16 bits (DX:AX)
cwde ; 66:98 - because here the 16 bits default mode is not followed – result in EAX

Address size prefix – 0x67

Bits 32
mov eax, [bx] ; 67:8B07 - because DS:[BX] is 16 bits addressing

Bits 16
mov BX, [EAX] ; 67:8B18 – because DS:[EAX] is 32 bits addressing

Bits 16

push dword[ebx] ; 66:67:FF33 – Here the default mode is Bits 16; because the addressing [EBX] is on 32 bits 67 will be generated and because a push is made to a DWORD operand instead of one on 16 bits 66 will be generated as a prefix

push dword[CS:ebx] ; 2E:66:67:FF33
rep push dword[CS:ebx] ; F3:2E:66:67:FF33
(rep push word ptr CS:[BP+DI] - superfluous REPxx fix ! - OllyDbg)

Bits 32
67:8B07 mov eax, [bx]; Offset_16_biti = [BX|BP] + [SI|DI] + [const]
(mov eax, dword ptr DS:[BX]) **67 – address size override prefix**

Bits 16
66:8B07 mov eax, [bx]; mov ax, word ptr DS:[edi]
(66 – operand size override prefix)

<u>Definition</u>
Instruction prefixes are assembly language constructs that appear optionally in the composition of a source line (explicit prefixes) or in the internal format of an instruction (prefixes generated implicitly by the assembler in two cases) and that modify the standard behavior of those instructions (in the case of explicit prefixes) or which signals the processor to change the default representation size of operands and/or addresses, sizes established by assembly directives (BITS 16 su BITS 32).