16 bit er displacement/direct-memory address er lower 8 bits jaabe byte 3 te and upper 8 bits jaabe byte 4 e

https://en.wikibooks.org/wiki/X86_Assembly/Machine_Language_Conversion

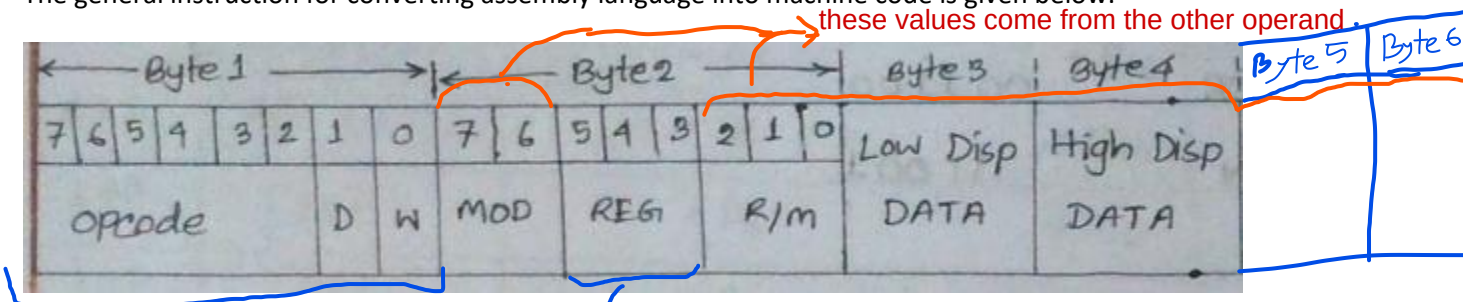
visit this page for more information ^^

Converting assembly language to machine language

max. 6 bytes

The general instruction for converting assembly language into machine code is given below:

these values come from the other operand



these values are taken from destination (if destination considered) or source (if source considered)

Opcode: Operation code

Operation	**Opcode**
MOV	100010
ADD	000000
SUB	010010
XOR	001100
IN	111001

D: Direction to register or from register

=1 if destination is considered

=0 if source is considered

W: Word or byte

=0 for 8 bit register

=1 for 16 bit register

MOD: Register mode or memory mode with displacement

MOD	**Indication**
00	Memory mode, no displacement
01	Memory mode, with 8 bit displacement. For example: [BX]+12H
10	Memory mode, with 16 bit displacement. For example: [BX]+1234H
11	Register mode, no displacement

REG: Identifies a register which is one of the instruction operands.

REG	**W=0**	**W=1**
000	AL	AX
001	CL	CX
010	DL	DX
011	BL	BX, DS
100	AH	SP
101	CH	BP
110	DH	SI
111	BH	DI

R/M: Register/Memory coding

– Depends on the MOD field

• If MOD = 11, then R/M field acts as a REG field (used for register to-register operations and other cases).

• If MOD ≠ 11, then R/M indicates how the effective address of the operand is calculated.

in 8086 emulator, you'll get same machine code for [BX] + [SI] & [BX+SI]
and it will add memory with it's displacement ie. treat [1234h] + [1234h] as 2468h (which is a value and not a memory location !)

MOD / R/M	00	01	10	11	
				W=0	W=1
000	[BX]+[SI] (=P)	→ P + D_8	→ P + D_{16}	AL	AX
001	[BX]+[DI]	+ D_8	+ D_{16}	CL	CX
010	[BP]+[SI]	+ D_8	+ D_{16}	DL	DX
011	[BP]+[DI]	+ D_8	+ D_{16}	BL	BX
100	[SI]	+ D_8	+ D_{16}	AH	SP
101	[DI]	+ D_8	+ D_{16}	CH	BP
110	Direct address	[SP] + D_8	[SP] + D_{16}	DH	SI
111	[BX]	+ D_8	+ D_{16}	BH	DI

Example-1: Convert the following assembly language into machine language.

MOV BL,AL

Solution:

Here, opcode: MOV

Assume, destination register is considered (BL)

hence, **Opcode:** 100010 [MOV]

D=1 [destination register is considered]

W=0 [Destination register (BL) is not a word (2 Bytes = 16 bits) size register, rather an 8 bit register]

REG: 011 [Destination is considered, so the register is BL]

MOD: 11 [The other operand (source) is AL, which is a register. .'. register mode, no displacement]

R/M: 000 [The other operand (source) is AL]

Also write the hex value.

Byte 1								Byte 2							
1	0	0	0	1	0	1	0	1	1	0	1	1	0	0	0
Opcode						D	W	Mod		REG			R/M		

8AD8

Example-2: Repeat example-1 considering register as source.

Solution:

Here, AL is source register.

Here, opcode: MOV

Assume, source register is considered (AL)

hence, **Opcode:** 100010 [MOV]

D=0 [Source register is considered]

W=0 [Source register (BL) is not a word sized register]

REG: 000 [Source register is AL]

MOD: 11 [The other operand (destination) is BL, which is a register. .'. register mode, no displacement]

R/M: 011 [The other operand (destination) is BL]

Byte 1								Byte 2							
1	0	0	0	1	0	0	0	1	1	0	0	0	0	1	1
Opcode						D	W	Mod		REG			R/M		

Example-3: ADD AX,[SI]

Solution:

Here, opcode: ADD

Assume, destination register is considered (AX)

Hence, **Opcode:** 000000 [ADD]

D=1 [Destination is register]

W=1 [Destination register is word size]

REG: 000 [Destination register is AX]

MOD: 00 [The other operand, source, is in memory mode with no displacement]

R/M: 100 [The other operand, source, is [SI]]

Byte 1								Byte 2							
0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0
Opcode						D	W	Mod		REG			R/M		

Example-4: ADD AX, [SI] + 12H

Solution:

Here, opcode: ADD

Assume, destination register is considered (AX)

Hence, **Opcode:** 000000 [ADD]

D=1 [Destination is register]

W=1 [Destination register is word size]

REG: 000 [Destination register is AX]

MOD: 01 [The other operand, source, is in memory mode with 8 bit displacement ([12], here 2 hex digit is worth 2*4=8 bits)]

R/M: 100 [The other operand, source, is [SI]+D8; here, D8 suggests 8 bit displacement]

Byte 1								Byte 2								Byte 3							
0	0	0	0	0	0	1	1	0	1	0	0	0	1	0	0	0	0	0	1	0	0	1	0
Opcode						D	W	Mod		REG			R/M			Displacement							

12 h = 0001 0010

Assignment:

Example-5: ADD AX, [SI]+1234H

Example-6: XOR CL, [1234H]

Solution: Here, opcode: XOR

Destination register is CL

Hence, **Opcode:** 001100 [XOR]

D=1 [Destination is register]

W=0 [Destination register is not word size]

REG: 001 [Destination register is CL]

MOD: 00 [other operand, source, is in memory mode, no displacement]

R/M: 110 [other operand, source, is direct address]

Byte 1							
0	0	0	0	0	0	1	1
Opcode						D	W

Byte 2							
0	1	0	0	0	1	0	0
Mod		REG			R/M		

Byte 3							
0	0	1	1	0	1	0	0
Lower byte							

Byte 4							
0	0	0	1	0	0	1	0
Higher byte							

12 h = 0001 0010

34 h = 0011 0100