

Daten

```
Tabelle DW 4 DUP (?) ; Array Definition
- - -
MOV BX, Index ; get Index
SHL BX, 1 ; Groesse: 2 Byte
; shift left = * 2
MOV AX, [BX + OFFSET Tabelle] ; Daten lesen 1)
MOV Res, AX ; Resultat speichern
```

**Beispiele:**

```
MOV AX, Zahl ;direkt
MOV DX, [BX] ;indirekt mit basis
MOV AL, [BX+4] ;indirekt mit basis und displ.
MOV AX, [BP-6] ;indirekt mit basis und displ.
MOV CX, [BX+SI] ;indirekt mit basis und index
MOV CH, [BX+DI-2] ;indirekt mit basis, index und displ.
```

```
.CODE
;
; index = 17;
1 mov word ptr _index,17
;
; charArray[index] = 0;
;
2 mov bx, word ptr _index
3 mov byte ptr _charArray[bx],0

.DATA
_charArray db 20 dup (?)
_index dw ?
```

**MOV BX, 2800h**  
**MOV SI, 0010h**  
**MOV AX, [BX+SI+2]** <sup>1)</sup>

Adressierung

Register	AX
Immediate	1234h
Direkt	DS:[1234h]
Register-Indirekt	[BX] oder [BX+SI+12h] oder 12h[BX+SI]
Implizit	

I/O	nur über AX und DX
-----	--------------------

**little endian**

Wort Adresse<sup>1)</sup> → lower byte (red) / higher byte (blue) → WORD → höherste Adresse

**big endian**

Wort Adresse<sup>1)</sup> → higher byte (blue) / lower byte (red) → WORD → tiefste Adresse

7 0 7 0

0 1 0 0 0 0 1 1 1 0 1 0 0 1 0 1

most significant byte MSB least significant byte LSB

**little endian (x86)**

Adresse	Daten
0x00200	0xA5
0x00201	0x43

Speicher

**big endian**

Adresse	Daten
0x00200	0x43
0x00201	0xA5

Speicher

Befehle

Unäre Instruktionen	Binäre Instruktionen
INC, DEC, NEG(2comp), NOT(bitweise), CBW, CWD	ADD, ADC, SUB, SBB, MUL, IMUL, DIV, IDIV