Array and Addressing Mode

Chapter 10

Prepared by Shareef Ahmed Tamal
Modified(Only Slightly) by Abdus Salam Azad

One Dimensional Array

- Declaration

W db 10, 20, 30, 40

Offset Address	Content
0000h	10
0001h	20
0002h	30
0003h	40

One Dimensional Array

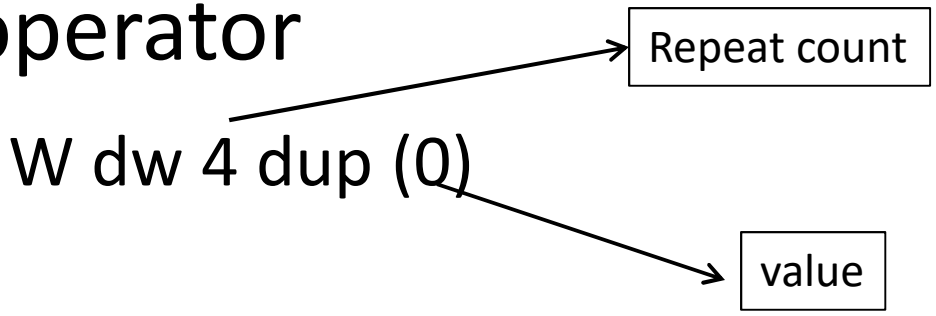
- Declaration

W dw 10, 20, 30, 40

Offset Address	Content
0000h	10
0002h	20
0004h	30
0006h	40

One Dimensional Array

- Dup operator



Offset Address	Content
0000h	0
0002h	0
0004h	0
0006h	0

One Dimensional Array

- Dup operator

W dw 3, 2, 2 dup (0, 3 dup (1), 5)

3, 2, 0, 1, 1, 1, 5, 0, 1, 1, 1, 5

One Dimensional Array

W dw 3, 2, 5, 8, 1, 6, 8

- Where is the location of 3rd element?

w + (3-1)*2

One Dimensional Array

W dw 3, 2, 5, 8, 1, 6, 8

- Calculate sum of 4^{th} and 6^{th} element

  - C : sum = w[3]+w[5]
  - Assembly : mov ax, w+6
    add ax, w+10

One Dimensional Array

W dw 3, 2, 5, 8, 1, 6, 8

- Calculate sum of all elements
  - C : sum=0;

    for(i=0;i<7;i++) sum+=w[i];
  - Assembly: ?

    Need a way to iterate over all elements in the array

Addressing Modes

The way an operand is specified

So Far we have seen…..

- Register Mode:
  - When an operand is a register
  - Example: **Mov Ax, Bx** ; src & dest both in register mode
- Immediate Mode:
  - When an operand is a constant
  - Example: **Mov Ax, 1**
- Direct Mode:
  - When an operand is a variable
  - Example: **Mov Ax, C**

There are 4 additional addressing modes

- Register Indirect
- Based
- Indexed
- Based Indexed

Types of Addressing Modes

- Register Indirect Mode:
  - mov ax, [si]
  - The register SI acts as a pointer to a memory location.
  - Offset address of the operand is contained in SI register.

SI	200h
AX	10

Offset Address	Content
0200h	10
0202h	20
0204h	30
0206h	40

Types of Addressing Modes

- Register Indirect Mode:

  - BX, SI, DI and BP register can be used to hold offset address.
  - For BX, SI and DI, segment number is contained in DS.
  - For BP, segment number is contained in ES.

W dw 3, 2, 5, 8, 1, 6, 8

- Calculate sum of all elements

```
W dw 3, 2, 5, 8, 1, 6, 8
Mov ax, @data
Mov ds, ax
Lea  SI, W    ; SI points to W array
Xor ax,ax   ; clear AX
Mov cx, 7

Sum:
      Add ax , [SI]
      Add  SI, 2  ; inc SI by 2 bytes/ 1 word as W is an word array
      Loop Sum
```

Types of Addressing Modes

- Based and Indexed Addressing Mode:
  - Operands offset address = displacement + contents in a register
  - Displacement can be:
    - A variable (Offset address of a variable)
    - Constant
    - Variable + Offset Address
  - Register can be:
    - BX, BP, SI, DI
    - Segment register rule same as Register Indirect Mode
    - If Bx or BP is used, then the mode is called Based
    - If SI or DI is used, then called Indexed

Types of Addressing Modes

- Based and Indexed Addressing Mode:
  - Syntax of an operand
    - [register + displacement]
    - [displacement + register]
    - [register] + displacement
    - displacement + [register]
    - Displacement[register]
  - Example:

```
W dw 3, 2, 5

Mov ax, [W+bx]
```

Types of Addressing Modes

- Based and Indexed Addressing Mode:
  - Syntax of an operand
    - [register + displacement]
    - [displacement + register]
    - [register] + displacement
    - displacement + [register]
    - Displacement[register]
  - Example:

```
W dw 3, 2, 5

Mov ax, [W]+bx
```

Another Addressing Mode

- Based and Indexed Addressing Mode:
  - Syntax of an operand
    - [register + displacement]
    - [displacement + register]
    - [register] + displacement
    - displacement + [register]
    - Displacement[register]
  - Example:

```
W dw 3, 2, 5

Mov ax, W[bx]
```

W dw 3, 2, 5, 8, 1, 6, 8

- Calculate sum of all elements

```
W dw 3, 2, 5, 8, 1, 6, 8

Mov ax, @data
Mov ds, ax
Xor ax,ax   ; clear AX
Xor bx,bx
Mov cx, 7

Sum:
      Add ax ,W[bx]
      Add  bx, 2  ; inc bx by 2 bytes/ 1 word as W is an word array
      Loop Sum
```

W dw 3, 2, 5, 8, 1, 6, 8

- Calculate sum of all elements

```
W dw 3, 2, 5, 8, 1, 6, 8

Mov ax, @data
Mov ds, ax
Xor ax,ax   ; clear AX
Xor bp,bp
Mov cx, 7

Sum:
      Add ax ,W[bp]
      Add  bp, 2  ; inc bp by 2 bytes/ 1 word as W is an word array
      Loop Sum
```

What is the problem with this code?

For bp, SS is the segment register

2 Dimensional Array

- 2 Ways to store 2D array.

1	2	3
4	5	6

  - Row Major Order

Offset	0000h	0002h	0004h	0006h	0008h	000Ah
Content	1	2	3	4	5	6

  - Column Major Order

Offset	0000h	0002h	0004h	0006h	0008h	000Ah
Content	1	4	2	5	3	6

2 Dimensional Array

- 2 Ways to store 2D array.

1	2	3
4	5	6

  - Row Major Order

```
W  dw   1 ,2, 3
   dw   4, 5, 6
```

  - Column Major Order

```
W  dw  1 ,4
   dw  2, 5
   dw  3, 6
```

2 Dimensional Array

- Locating an element
  - A is an *M*x*N* array
  - Locate A[i][j]

1. If A stored in row major order then,

   Address of A[i][j] = A + ((i-1)xN + (j-1)) x S

2. If A stored in column major order then,

   Address of A[i][j] = A + ((i-1) + (j-1)xM) x S

where S=1 if A is byte array and 2 if A is word array

Based Indexed Addressing Mode

- Elements addressed as

  Variable [BX or BP] [SI or DI]

  [BX or BP] + [SI or DI] + variable + constant

  Variable[BX or BP + SI or DI + constant]

  Constant [BX or BP + SI or DI + variable]

- Whatever the format, the address is sum of all components

```
W dw 10, 20, 30, 40, 50  ;  bx has 2, SI has 4

Mov ax, w[BX + SI] ; ax = 40
Mov ax, w[BX + SI + 2] ; ax = 50
```

PTR Operator

- Mov Ax,1 ; Legal
- Mov Bh,5 ; Legal
- Mov [Bx],1 ; Legal or Illegal?

Ans: Illegal

- Mov **BYTE PTR** [BX],1
- Mov **WORD PTR** [BX],1

PTR Operator

- A db 1
- B db 2
- Mov ax, A ; Illegal

- Mov ax, Word PTR A

More

- Label pseudo-op
- Xlat Instruction