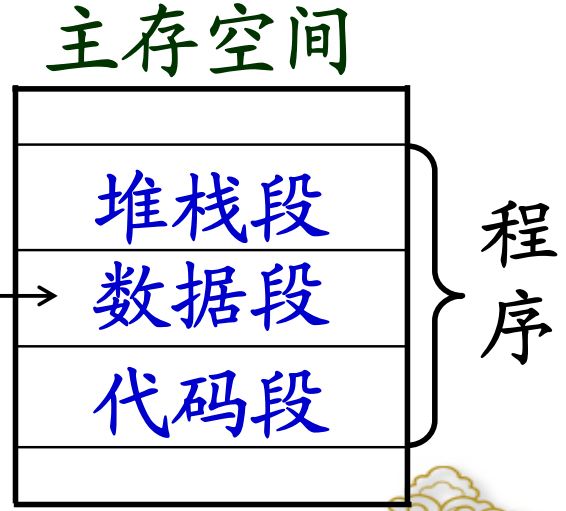
汇编语言的变量定义

什么是变量

- 变量（**Variable**）
  - 随程序运行会发生变化的数据
  - 保存在可读可写的主存空间
- 变量的实质是主存单元的数据，因而可以改变
  - 变量需要事先定义才能使用
  - 变量具有属性方便应用

变量表达主存数据，即存储器操作数



变量的定义

变量名　　变量定义伪指令　　初值表

变量名是用户标识符，表示首元素逻辑地址

伪指令助记符：byte、word、dword...，表示变量类型

初值表是用逗号分隔的一个或多个参数，表示变量初值

主要的变量定义伪指令

助记符：**BYTE** 变量类型：字节	分配一个或多个字节单元 每个数据是8位、字节量
助记符：**WORD** 变量类型：字	分配一个或多个字单元 每个数据是16位、字量
助记符：**DWORD** 变量类型：双字	分配一个或多个双字单元 每个数据是32位、双字量

对应C语言变量类型：**char　short　long**

变量定义的初值表

- 变量定义是申请存储空间
  - 同时还可以进行存储单元初始化
  - 即用初值表赋予变量初值
- 初值表可以有一个或多个参数，多个参数用逗号分隔
  - 各种形式的常量
  - 使用“？”表示初值不确定，即未赋初值
  - 使用复制操作符**DUP**表示多个同样数值

复制操作符格式：　重复次数　DUP(重复参数)

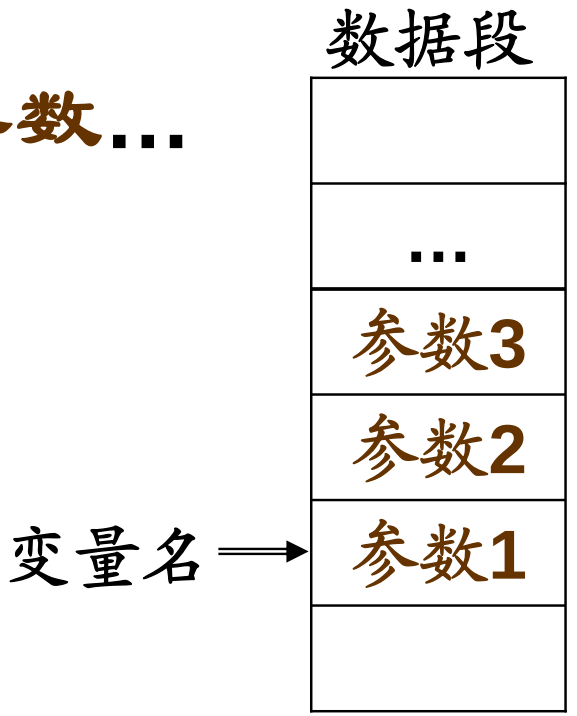
本讲总结

- 变量需要先定义才能使用
- 变量定义的格式

  变量名　变量定义伪指令　参数, 参数...

- 变量定义之后
  - 分配了存储空间
  - 赋予了初值（也可以没有初值）
  - 创建了变量名

  （可以获知变量的地址和类型）



8位变量定义

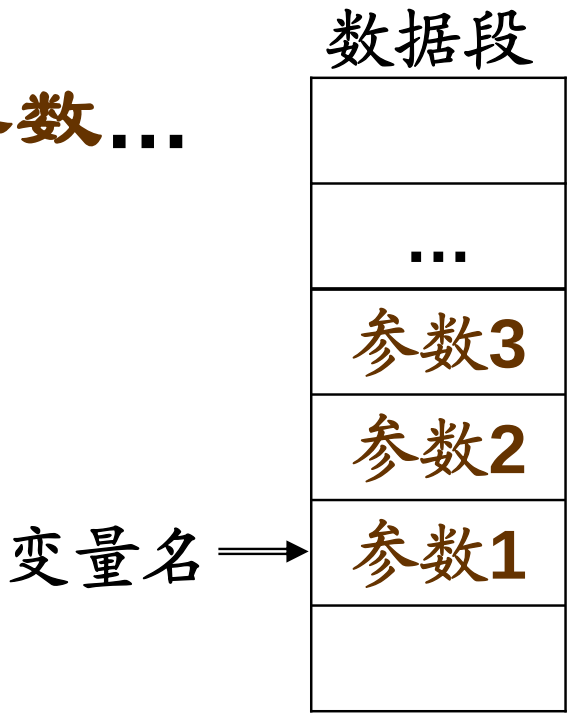
汇编语言的变量定义

- 变量需要先定义才能使用
- 变量定义的格式

  变量名　变量定义伪指令　参数, 参数...

- 变量定义之后
  - 分配了存储空间
  - 赋予了初值（也可以没有初值）
  - 创建了变量名

  （可以获知变量的地址和类型）



主要的变量定义伪指令

助记符：**BYTE** 变量类型：字节	分配一个或多个字节单元 每个数据是8位、字节量
助记符：**WORD** 变量类型：字	分配一个或多个字单元 每个数据是16位、字量
助记符：**DWORD** 变量类型：双字	分配一个或多个双字单元 每个数据是32位、双字量

对应C语言变量类型：**char　short　long**

BYTE：定义字节量(8位)数据的变量

- 8位无符号整数：0～255
- 8位补码表示的有符号整数：-128～+127
- 一个字符（ASCII码值）
- 压缩BCD码：0～99
- 非压缩BCD码：0～9
- ……

8位（字节量、Byte-sized）

```
msg      byte  'Hello',13,10,0
const1   byte  100,64h,'d'
const6   byte  4*4,34h+34
```

定义字符串要使用字节变量定义BYTE

字节变量程序一1

```
00000000        00 80 FF 80 00 7F
bvar1   byte 0,128,255,-128,0,+127
```

相对地址　　汇编语句　　机器指令

BYTE伪指令定义的每个数据都是8位、1个字节

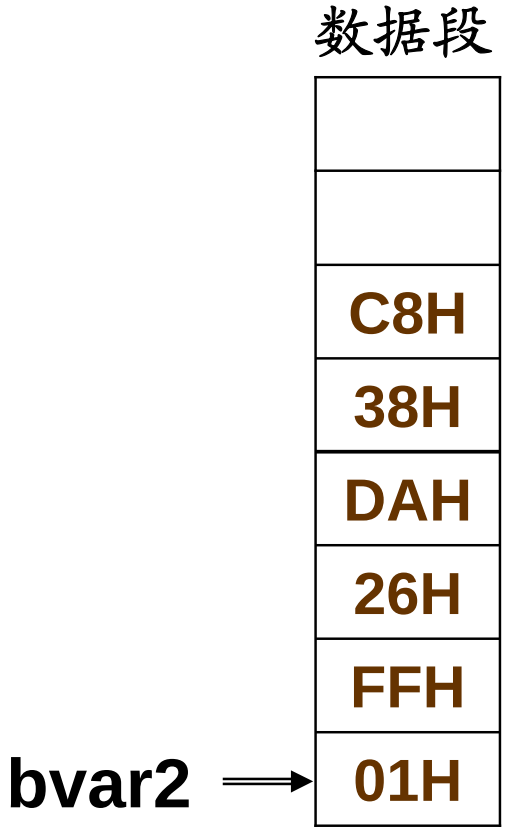
字节变量程序—2

00000006　　01 FF 26 DA 38 C8

bvar2　byte　1,-1,38,-38,38h,-38h

38和-38，十进制数

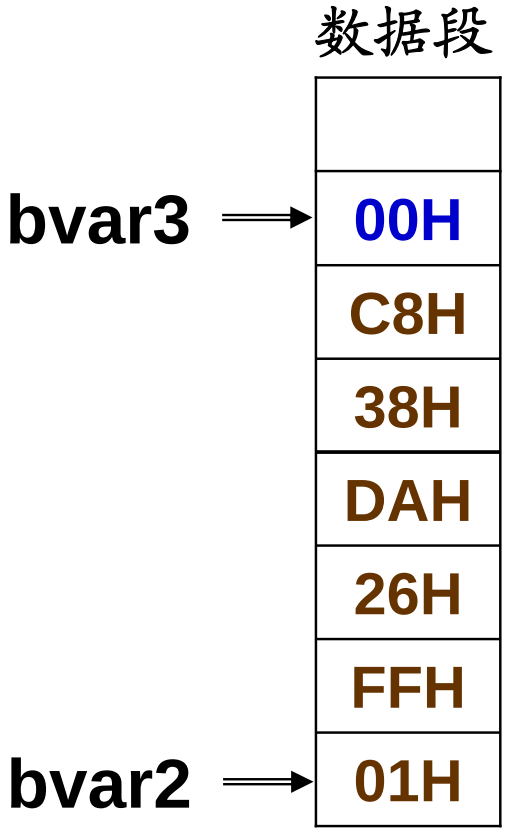
38H和-38H，十六进制数



字节变量程序—3

```
00000006        01 FF 26 DA 38 C8
bvar2   byte  1,-1,38,-38,38h,-38h
0000000C        00
bvar3   byte ?
```

变量BVAR3无初值
只在主存保留存储空间
实际上用0填充



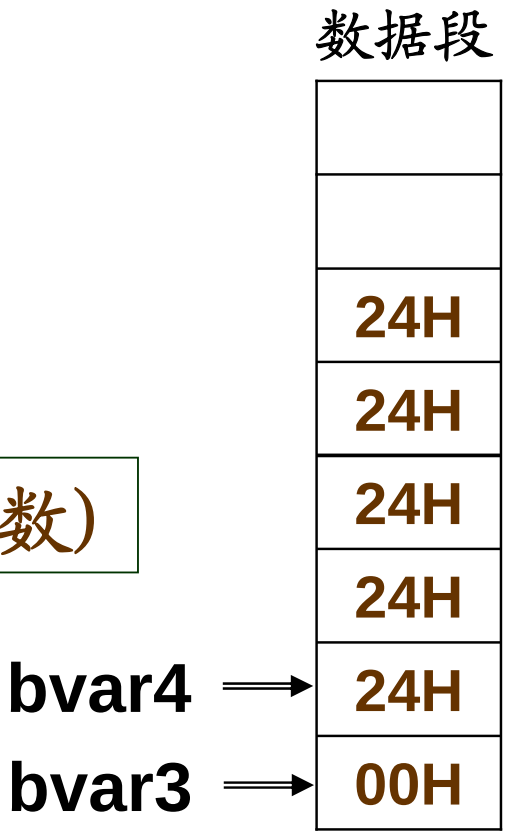
字节变量程序—4

0000000D　　00000005 [24]

bvar4　byte 5 dup ('$')

变量BVAR4定义了5个相同的数据

复制操作符格式：　重复次数 DUP(重复参数)



字节变量程序—5

=0000000A　　minint = 10

00000012　　0000000A [00] 0000000A [0A 00]

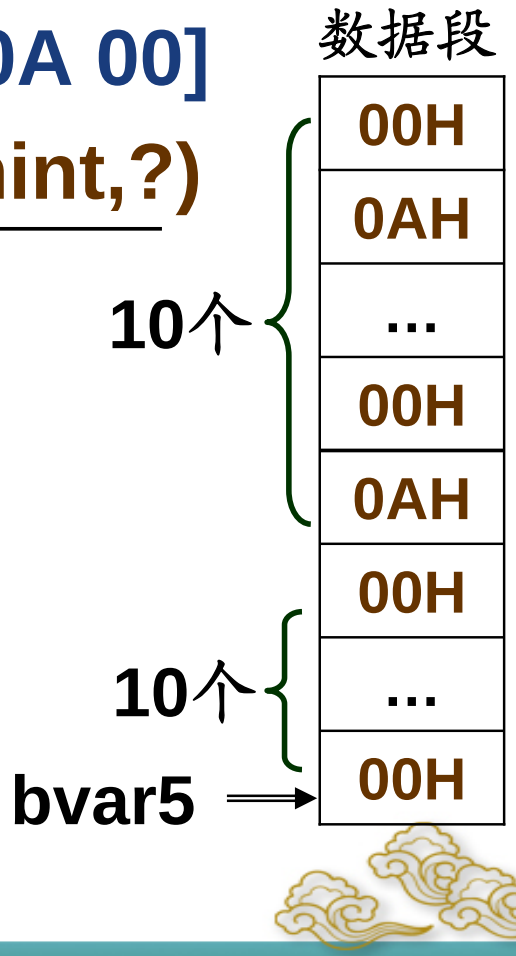
bvar5　byte minint dup(0),minint dup(minint,?)

实际上就是

bvar5　byte 10 dup(0),10 dup(10,?)

符号常量使用等价的数值替代

变量BVAR5定义了10个0，再10个10和0



字节变量程序—6

00000030　　00000002 [02 03 00000002 [04]]

byte 2 dup(2,3,2 dup(4))

复制操作符DUP可以嵌套

主存中分配了变量初值，但没有变量名

复制操作符格式：重复次数 DUP(重复参数)

数据段

04H
04H
03H
02H
04H
04H
03H
02H

本讲总结

- **8位变量定义，使用byte伪指令**
  - 每个数据是一个字节量，占用一个存储单元
- 字符（串）变量定义也使用**byte**伪指令
  - 每个字符是一个**8位ASCII码**
- 变量定义的参数不区别有无符号
  - 可以是无符号数，也可以是有符号数

对应C语言变量类型：**char**

16位变量定义

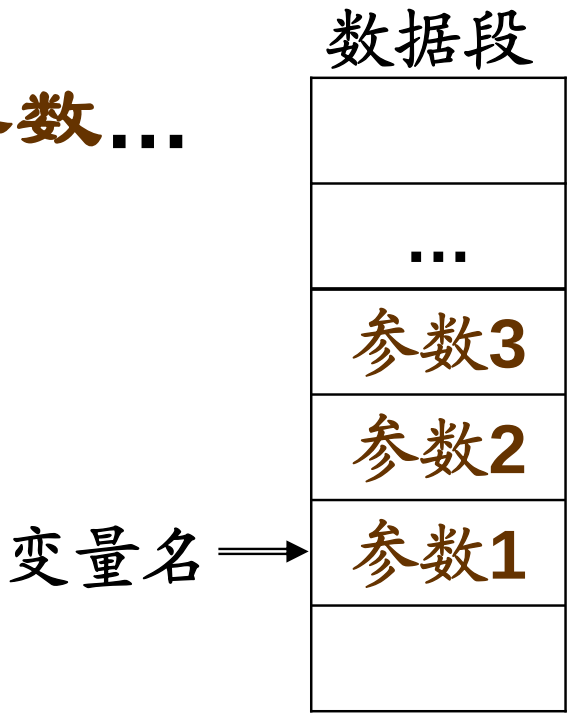
汇编语言的变量定义

- 变量需要先定义才能使用
- 变量定义的格式

  变量名　变量定义伪指令　参数, 参数...

- 变量定义之后
  - 分配了存储空间
  - 赋予了初值（也可以没有初值）
  - 创建了变量名

  （可以获知变量的地址和类型）



主要的变量定义伪指令

助记符：**BYTE** 变量类型：字节	分配一个或多个字节单元 每个数据是8位、字节量
助记符：**WORD** 变量类型：字	分配一个或多个字单元 每个数据是16位、字量
助记符：**DWORD** 变量类型：双字	分配一个或多个双字单元 每个数据是32位、双字量

对应C语言变量类型：**char　short　long**

WORD：定义字量(16位)数据的变量

- 16位无符号整数：0～65535
- 16位补码表示的有符号整数：-32768～+32767
- 16位段地址
- 16位偏移地址
- ……

16位（字量、Word-sized）

```
wvar1   word 0,-32768,65535
wvar2   word ?
minint  = 10
wvar3   word 5 dup(minint)
```

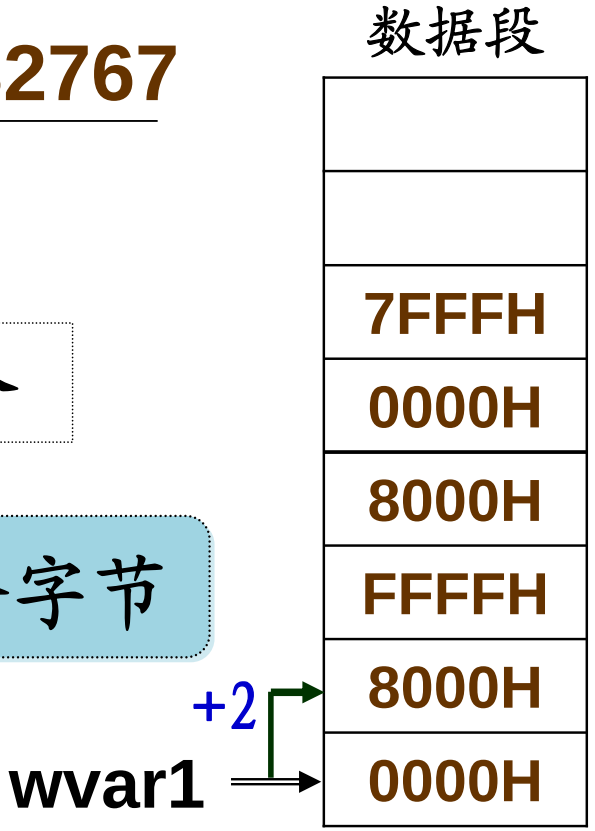
字变量程序—1

00000000　　0000 8000 FFFF 8000 0000 7FFF

wvar1　word 0,32768,65535,-32768,0,+32767

相对地址　　汇编语句　　机器指令

WORD伪指令定义的每个数据都是16位、2个字节



字变量程序—2

0000000C　　0001 FFFF 0026 FFDA 0038 FFC8

wvar2　word 1,-1,38,-38,38h,-38h

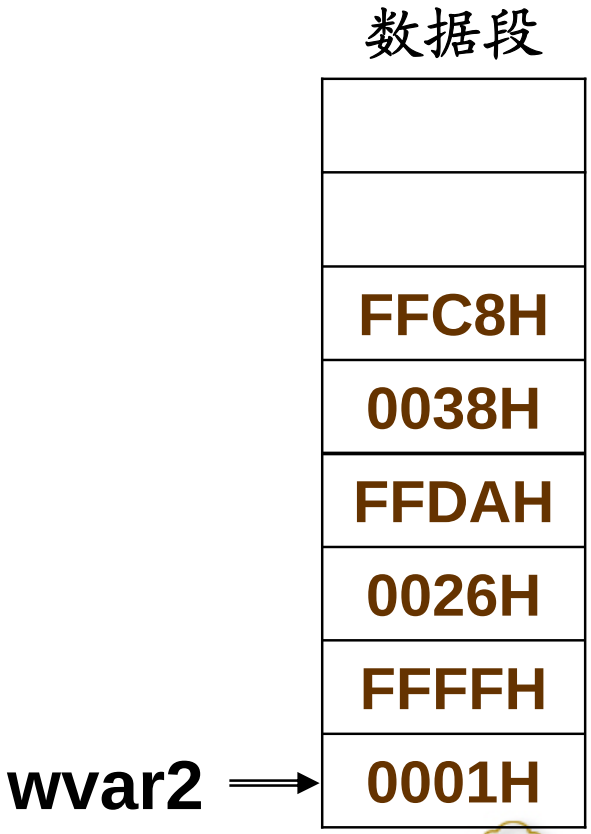
1和－1，字节量01H和FFH

字量是0001H和FFFFH

负数－38H，

字节量补码：C8H(＝[1]00H－38H)

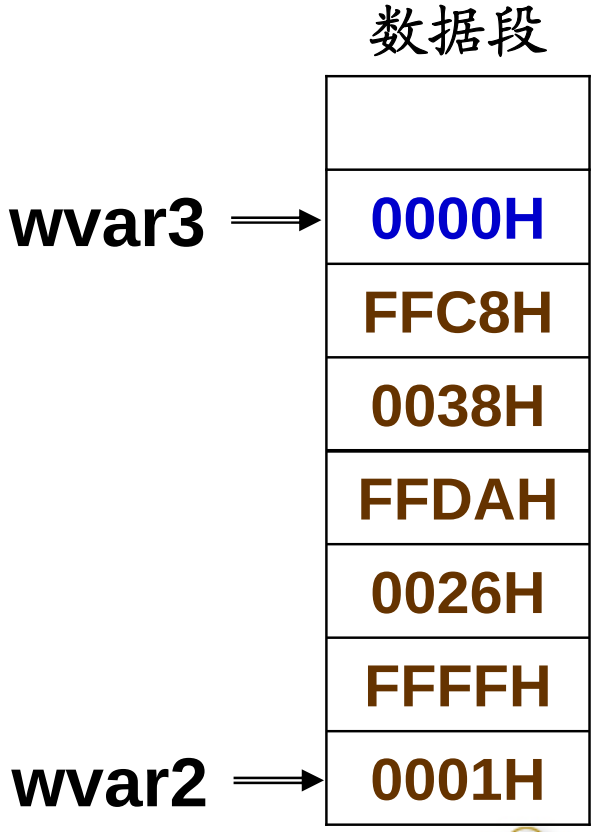
字量补码：FFC8H(＝[1]0000H－0038H)



字变量程序—3

```
0000000C      0001 FFFF 0026 FFDA 0038 FFC8
wvar2  word 1,-1,38,-38,38h,-38h
000000180000
wvar3  word ?
```

WVAR3无初值（被填入0）
占用16位（2个字节）空间



字变量程序—4

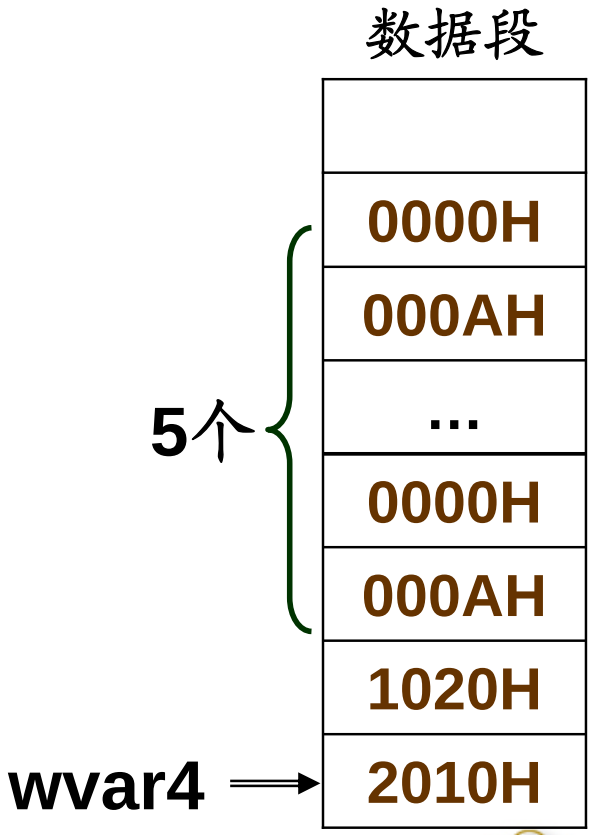
```
0000001A        2010 1020
wvar4   word 2010h,1020h
=0000000A       minint = 10
0000001E        00000005 [000A 0000]
```

word 5 dup(minint,?)

实际上就是

word 5 dup(10,?)

符号常量使用等价的数值替代



本讲总结

- **16位变量定义，使用word伪指令**
  - 每个数据是一个字量，占用2个存储单元
- 变量定义的参数不区别有无符号
  - 可以是无符号数，也可以是有符号数

对应C语言变量类型：**short**

还有一个问题：

16位数据占2个存储单元，哪个在前、哪个在后呢？

32位变量定义

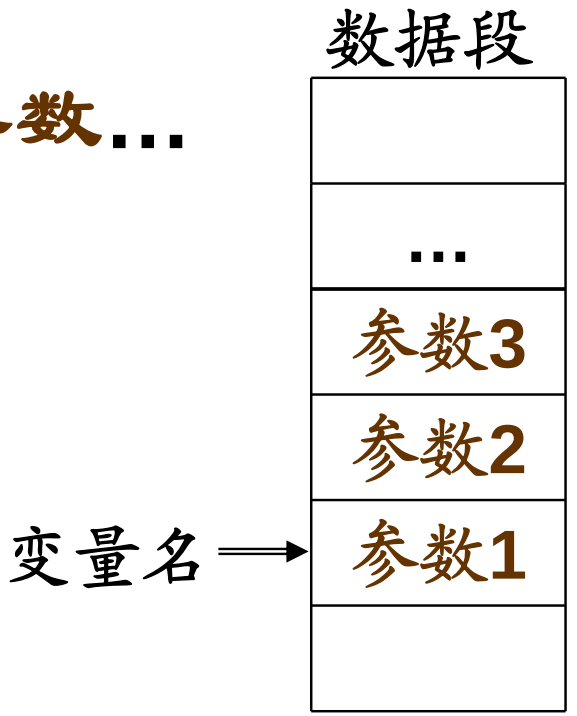
汇编语言的变量定义

- 变量需要先定义才能使用
- 变量定义的格式

  变量名　变量定义伪指令　参数, 参数...

- 变量定义之后
  - 分配了存储空间
  - 赋予了初值（也可以没有初值）
  - 创建了变量名

  （可以获知变量的地址和类型）



主要的变量定义伪指令

助记符：**BYTE** 变量类型：字节	分配一个或多个字节单元 每个数据是8位、字节量
助记符：**WORD** 变量类型：字	分配一个或多个字单元 每个数据是16位、字量
助记符：**DWORD** 变量类型：双字	分配一个或多个双字单元 每个数据是32位、双字量

对应C语言变量类型：**char　short　long**

DWORD：定义双字量(32位)数据的变量

- 32位无符号整数：$0 \sim 2^{32}-1$
- 32位补码表示的有符号整数：$-2^{31} \sim +2^{31}-1$
- 32位逻辑地址（含16位段地址和16位偏移地址）
- ……

32位（双字量、Doubleword-sized）

```
dvar1   dword 0,80000000h,0ffffffffh
dvar2   dword ?
array   dword 5 dup(0)
```

双字变量程序—1

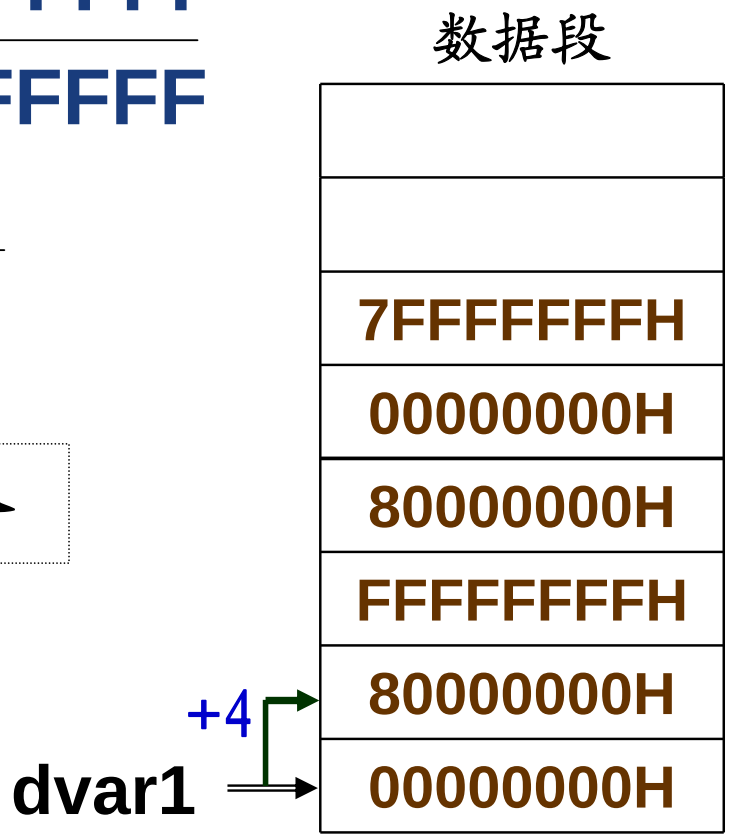
00000000　00000000 80000000 FFFFFFFF

80000000 00000000 7FFFFFFF

```
dvar1   dword 0,80000000h,0ffffffffh
        dword -80000000h,0,7fffffffh
```

相对地址　汇编语句　机器指令

DWORD伪指令定义的
每个数据都是32位、4个字节



双字变量程序—2

00000018　00000001 FFFFFFFF 00000026

FFFFFFDA 00000038 FFFFFFC8

dvar2　dword 1,-1,38,-38,38h,-38h

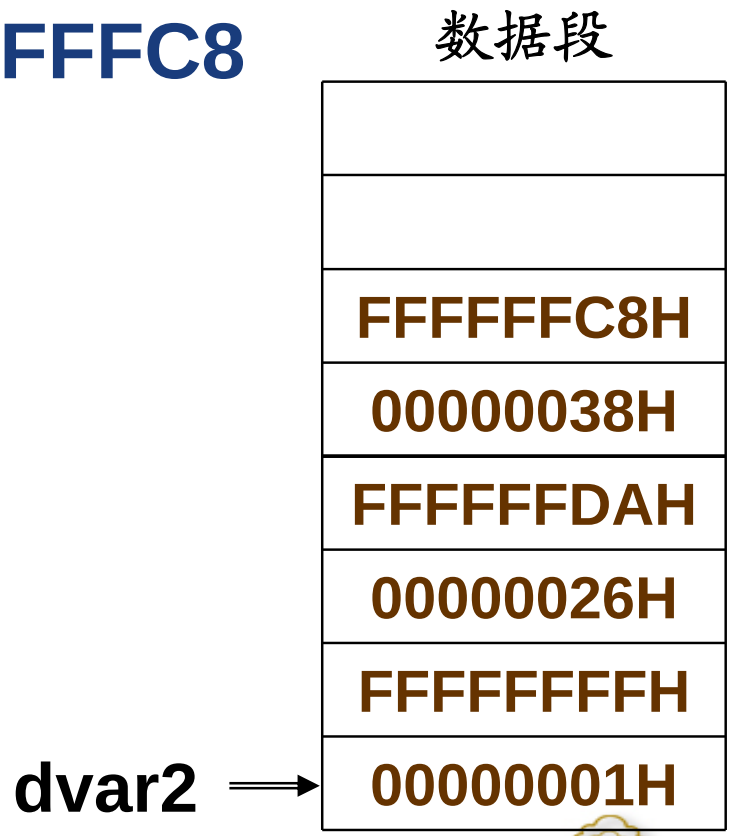
1和－1，字节量01H和FFH

双字量是00000001H和FFFFFFFFH

负数－38H，

字节量补码：C8H(＝[1]00H－38H)

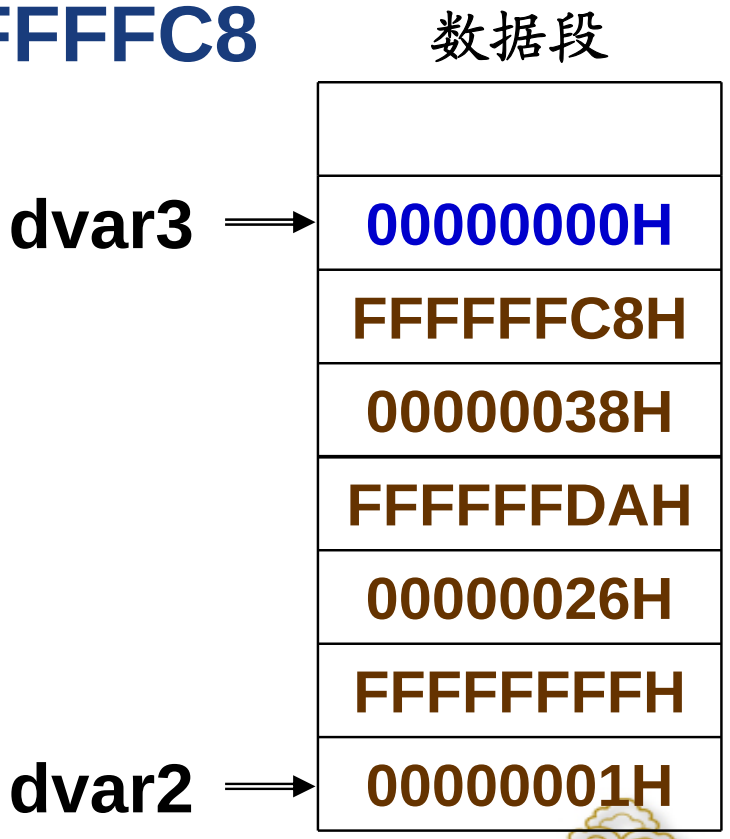
双字量补码：FFFFFFC8H



双字变量程序—3

```
00000018   00000001 FFFFFFFF 00000026
           FFFFFFDA 00000038 FFFFFFC8
dvar2   dword 1,-1,38,-38,38h,-38h
00000030   00000000
dvar3   dword ?
```

DVAR3无初值（被填入0）
占用32位（4个字节）空间

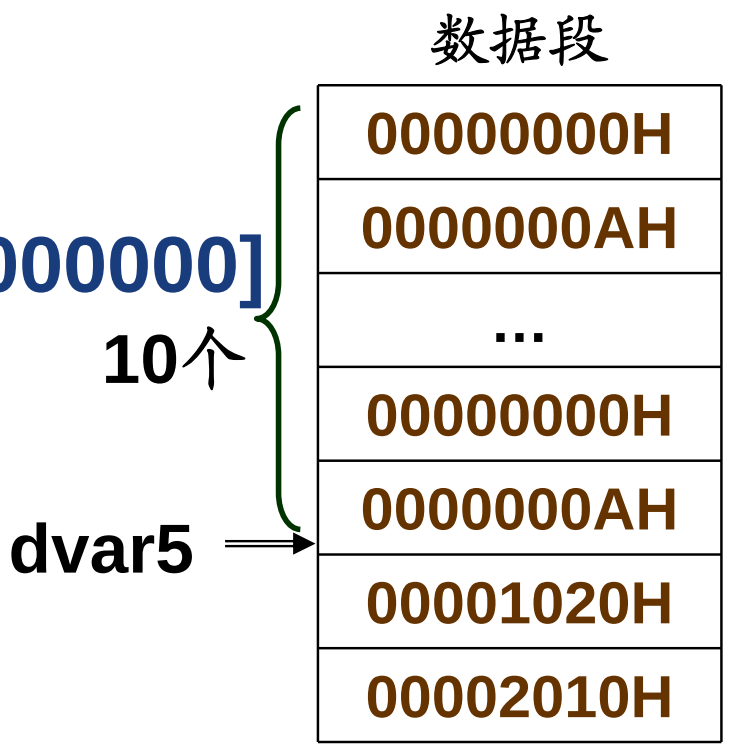


双字变量程序—4

```
00000034   00002010 00001020
           dword 2010h,1020h
=0000000A    minint = 10
0000003C   0000000A [0000000A 00000000]
dvar5   dword minint dup(minint,?)
```

实际上就是

```
dvar5   word 10 dup(10,?)
```



符号常量使用等价的数值替代

本讲总结

- **32位变量定义，使用dword伪指令**
  - 每个数据是一个双字量，占用4个存储单元
- 变量定义的参数不区别有无符号
  - 可以是无符号数，也可以是有符号数

对应C语言变量类型：**long**

还有一个问题：

32位数据占4个存储单元，如何安排这4个单元？