Учебный курс. Часть 11. Умножение и деление

Автор: xrnd | Рубрика: Учебный курс | 01-04-2010 |
Распечатать запись

Умножение и деление выполняются по-разному для чисел со знаком и без, поэтому в системе команд процессора x86 есть отдельные команды умножения и деления для чисел со знаком и для чисел без знака.

Умножение чисел без знака

Для умножения чисел без знака предназначена команда MUL. У этой команды только один операнд — второй множитель, который должен находиться в регистре или в памяти. Местоположение первого множителя и результата задаётся неявно и зависит от размера операнда:

Размер операнда	Множитель	Результат
Байт	AL	AX
Слово	AX	DX:AX

Отличие умножения от сложения и вычитания в том, что разрядность результата получается в 2 раза больше, чем разрядность сомножителей. Также и в десятичной системе — например, умножая двухзначное число на двухзначное, мы можем получить в результате максимум четырёхзначное. Запись «DX:AX» означает, что старшее слово результата будет находиться в DX, а младшее — в AX. Примеры:

```
    mul bl    ;AX = AL * BL
```

```
    mul ax    ;DX:AX = AX * AX
```

Если старшая часть результата равна нулю, то флаги *CF* и *OF* будут иметь нулевое значение. В этом случае старшую часть результата можно отбросить. Это свойство можно использовать в программе, если результат должен быть такого же размера, как множители.

Умножение чисел со знаком

Для умножения чисел со знаком предназначена команда IMUL. Эта команда имеет три формы, различающиеся количеством операндов:

- *С одним операндом* — форма, аналогичная команде MUL. В качестве операнда указывается множитель. Местоположение другого множителя и результата определяется по таблице.
- *С двумя операндами* — указываются два множителя. Результат записывается на место первого множителя. Старшая часть результата в этом случае игнорируется. Кстати, эта форма команды не работает с операндами размером 1 байт.
- *С тремя операндами* — указывается положение результата, первого и второго множителя. Второй множитель должен быть непосредственным значением. Результат имеет такой же размер, как первый множитель, старшая часть результата игнорируется. Это форма тоже не работает с однобайтными множителями.

Примеры:

```
    imul cl           ;AX = AL * CL
    imul si           ;DX:AX = AX * SI
    imul bx,ax        ;BX = BX * AX
    imul cx,-5        ;CX = CX * (-5)
    imul dx,bx,134h   ;DX = BX * 134h
```

CF = *OF* = 0, если произведение помещается в младшей половине результата, иначе *CF* = *OF* = 1. Для второй и третьей формы команды *CF* = *OF* = 1 означает, что произошло переполнение.

Деление чисел без знака

Деление целых двоичных чисел — это всегда деление с остатком! По аналогии с умножением, размер делителя, частного и остатка должен быть в 2 раза меньше размера делимого. Деление чисел без знака осуществляется с помощью команды DIV. У этой команды один операнд — делитель, который должен находиться в регистре или в памяти. Местоположение делимого, частного и остатка задаётся неявно и зависит от размера операнда:

Размер операнда (делителя)	**Делимое**	**Частное**	**Остаток**
Байт	AX	AL	AH
Слово	DX:AX	AX	DX

При выполнении команды DIV может возникнуть *прерывание* (о прерываниях я подробно расскажу потом, пока старайтесь избегать таких случаев):

- если делитель равен нулю;
- если частное не помещается в отведённую под него разрядную сетку (например, если при делении слова на байт частное больше 255).

Примеры:

```
    div cl   ;AL = AX / CL, остаток в AH
    div di   ;AX = DX:AX / DI, остаток в DX
```

Деление чисел со знаком

Для деления чисел со знаком предназначена команда IDIV. Единственным операндом является делитель. Местоположение делимого и частного определяется также, как для команды DIV. Эта команда тоже генерирует прерывание при делении на ноль или слишком большом частном.

Пример программы

Допустим, в программе требуется вычислять координату какого-то движущегося объекта по формуле:

$$x = x_0 + v_0 t + at^2/2$$

Все числа в правой части — 8-битные целые без знака, а x — 16-битное целое и тоже без знака. Здесь нужно внимательно следить за размерами операндов.

```
 1 use16                 ;Генерировать 16-битный код
 2 org 100h              ;Программа начинается с адреса 100h
 3
 4     mov al,[v0]       ;AL = v0
 5     mov cl,[t]        ;CL = t
 6     mul cl            ;AX = AL*CL = v0*t
 7     mov bx,ax         ;BX = AX = v0*t
 8
 9     mov al,[a]        ;AL = a
10     mul cl            ;AX = AL*CL = a*t
11     mov ch,0          ;Преобразуем t в слово в регистре CX
12     mul cx            ;DX:AX = AX*CX = a*(t^2)
13     mov cl,2          ;CL = 2 = CX, так как CH = 0
14     div cx            ;AX = DX:AX/2 = a*(t^2)/2
15
16     add ax,bx         ;AX = AX+BX = v0*t + a*(t^2)/2
17     add al,[x0]       ;\
18     adc ah,ch         ;/ AX = AX+x0 = x0 + v0*t + a*(t^2)/2
19
20     mov [x],ax        ;Сохраняем результат в x
```

```
21
22      mov ax,4C00h    ;\
23      int 21h         ;/ Завершение программы
24 ;-------------------------------------------------
25 x0   db 188
26 v0   db 7
27 a    db 3
28 t    db 25
29 x    dw ?
```

В 7-й строке промежуточный результат сохраняется в bx. В 11-й строке происходит преобразование байта в слово, путём добавления нулевой старшей части. Такой метод подходит для чисел без знака, но приведёт к ошибке для чисел со знаком (в случае отрицательного числа). Прибавление x_0 происходит в два этапа (строки 17 и 18) с учётом переноса, так как мы складываем слово и байт.

Упражнение

Напишите программу для вычисления формулы
$z = (x \cdot y) / (x + y)$. Все числа 16-битные целые со знаком.

Сложное упражнение

Напишите программу для умножения двух 32-битных целых без знака. В результате должно получиться 64-битное целое без знака. Подсказка: используйте комбинацию умножения по частям и сложения, как в способе умножения столбиком. (Мой вариант решения вы можете посмотреть здесь).

Следующая часть »

Комментарии:

RoverWWWorm
02-04-2010 18:44

;посмотрите правильно или нет

```
use16
org 100h

mov ax,[x]
mov cx,[y]
imul cx ;DX:AX=AX*CX

add cx,[x]

idiv cx ;AX=DX:AX/CX, остаток в DX

mov [z],ax

mov ax,4c00h
int 21h

;_____________
x dw 6
y dw (-4)
z dw ?
```

[Ответить]

xrnd
03-04-2010 01:00

Почему-то твою программу сайт принял за спам 🙂

Да, здесь всё верно написано! Единственное, что круглые скобки в строке «y dw (-4)» ставить не обязательно, можно просто писать «-4». Попробуй ещё решить упражнение из предыдущего урока. И можно на ты, мне так привычнее))

[Ответить]

RoverWWWorm
03-04-2010 08:34

Уф, чуть не вступил в ряды спамщиков:)))
Ну постараюсь решить упражнение из предыдущего урока Лады, будем не ты))

[Ответить]

RoverWWWorm
06-04-2010 07:51

А когда будет следующий урок?

[Ответить]

xrnd
06-04-2010 16:09

В ближайшие дни 🙂

[Ответить]

RoverWWWorm
16-04-2010 20:41

НУ где же уроки, ааа(((

[Ответить]

xrnd
18-04-2010 18:17

Написал ещё одну часть 🙂

[Ответить]

ВИКА
05-12-2010 21:19

какую-же?)

[Ответить]

xrnd
06-12-2010 13:40

На момент написания комментария — 12-ю часть.

[Ответить]

mc-black
11-05-2010 23:13

```
[source]use16 ;Генерировать 16-битный код
org 100h ;Программа начинается с адреса 100h

mov ax,word[value1]
mul word[value2]
mov word[result],ax
mov word[result+2],dx

mov ax,word[value1+2]
mul word[value2]
add word[result+2],ax
adc dx,0
mov word[result+4],dx

mov ax,word[value1]
mul word[value2+2]
add word[result+2],ax
adc word[result+4],dx
```

mov ax,0
adc ax,0
mov word[result+6],ax

mov ax,word[value1+2]
mul word[value2+2]
add word[result+4],ax
adc word[result+6],dx

mov ax,4C00h
int 21h
;————————————————————————————————————-
value1 dd 6F6056EFh
value2 dd 2h
result rd 2[/source]

32-битное умножение. Сначала ошибочно запостил в предыдущий урок.

[Ответить]

xrnd
12-05-2010 00:50

Я проверил 🙂 Всё правильно работает.
Для сравнения вот мой вариант программы (я его написал, когда придумал упражнение)

```
 1 use16                 ;Генерировать 16-битный код
 2 org 100h              ;Программа начинается с адреса 100h
 3
 4     mov ax,word[a]
 5     mul word[b]
 6     mov di,ax
 7     mov si,dx
 8     sub cx,cx
 9     sub bx,bx
10
```

```
11      mov ax,word[a+2]
12      mul word[b]
13      add si,ax
14      adc cx,dx
15
16      mov ax,word[a]
17      mul word[b+2]
18      add si,ax
19      adc cx,dx
20      adc bx,0
21
22      mov ax,word[a+2]
23      mul word[b+2]
24      add cx,ax
25      adc bx,dx
26
27      mov word[c],di
28      mov word[c+2],si
29      mov word[c+4],cx
30      mov word[c+6],bx
31
32      mov ax,4C00h    ;
33      int 21h         ;/ Завершение программы
34 ;-------------------------------------------------------
35 align 4
36 a   dd $FFFFFFFF
37 b   dd $FFFFFFFF
38 c   dq ?
```

Вместо sub cx,cx я обычно использую xor. Но в этой части ещё не изучены логические команды.

[Ответить]

bor1k
30-07-2010 11:26

так правильно?

```
use16 ;Ãåíåðèðîâàòü 16-áèòíûé êîä
org 100h ;Ïðîãðàììà íà÷èíàåòñÿ ñ àäðåñà 100h
;z = (x·y) / (x + y)

mov ax,[x]
mov cx,[y]
imul cx
mov bx,[x]
add bx,[y]
idiv bx
mov [z],ax
mov ax,4c00h
int 21h

;---------------------------------------------------

x dw 343
y dw 531
z dw ?
```

[Ответить]

xrnd
30-07-2010 13:33

Да. Всё правильно.

[Ответить]

bor1k
31-07-2010 09:18

xrnd сделай пожалуйста листинг с комментариями к заданию с 32-битномым умножением.

[Ответить]

xrnd
31-07-2010 11:18

Я даже написал небольшое объяснение алгоритма.
http://asmworld.ru/isxodniki/32-bitnoe-umnozhenie/

[Ответить]

bor1k
31-07-2010 20:29

по идее на 64-битном процессоре, эти операции будут не нужны

[Ответить]

xrnd
31-07-2010 21:29

Даже на 32-битном можно обычной командой умножать 32 на 32 бита и получать 64-битное значение. Но там также придётся умножать 64 на 64 бита.

Важен сам принцип умножения чисел, больших чем разрядность процессора. Например, есть 8-битные микроконтроллеры и на них просто так не умножишь большие числа.

[Ответить]

IgorKing
31-08-2010 16:58

32-битное правильно?

use16
org 100h

```
mov ax,word[a]
imul word[b]
mov cx,dx
mov word[c],ax

mov ax,word[a]
imul word[b+2]
add cx,ax
adc dx,0
mov bx,dx

mov ax,word[a+2]
imul word[b]
add cx,ax
adc bx,dx
mov word[c+2],cx

mov ax,word[a+2]
imul word[b+2]
add bx,ax
adc dx,0
mov word[c+4],bx
mov word[c+6],dx

mov ax,4c00h
int 21h
;---------------------------
a dd 0xF3D1C7E6
b dd 0xA6C012BC
c dq ?
```

[Ответить]

xrnd
02-09-2010 20:48

Принцип ты правильно понял, но команда IMUL здесь будет неправильно работать.
Я не случайно предлагал в задании умножать числа без знака. При умножении старший бит в каждом слове будет обрабатываться как знаковый и это приведёт к ошибке в результате.

[Ответить]

Борис
19-12-2010 14:20

1-е задание

```
use16 ;16-bit cod
org 100h ;begin 100h
;z = (x·y) / (x + y)

mov ax,[x]
imul [y] ;dx:ax=x*y
mov bx, [x]
add bx, [y] ;bx=x+y
idiv bx ;ax=(xy)/(x+y) dx — ostatok
mov [z], ax
mov [z0], dx
mov ax,4C00h ;\
int 21h ;/ end
;————————————————————————————————-
x dw -3
y dw 2
z dw ?
z0 dw ?
```

2-е задание

```
use16 ;16-bit cod
org 100h ;begin 100h
;Z = (Xh:Xl*Yh:Yl)
```

```
mov ax, word[x]
mul word[y] ;Dx:Ax = Xl*Yl
mov word[z],ax
mov word[z+2],dx

mov ax, word[x+2]
mul word[y] ;Dx:Ax=Xh*Yl
add word[z+2],ax
adc word[z+4],dx

mov ax, word[x]
mul word[y+2] ;Dx:Ax=Xl*Yh
add word[z+4], ax
adc word[z+6], dx

mov ax, word[x+2]
mul word[y+2] ;Dx:Ax=Xh*Yh
add word[z+6], ax
adc word[z+8], dx

mov ax,4C00h ;\
int 21h ;/ end
;-------------------------------------------------
x dd 0xff32
y dd 0x32ff
```

Что-то во втором я сомневаюсь.

Спасибо!!!!

[Ответить]

Борис
19-12-2010 14:26

О посмотрел ваше решение и поправил последние 2 блока так

```
mov ax, word[x]
mul word[y+2] ;Dx:Ax=Xl*Yh
add word[z+2], ax
adc word[z+4], dx

mov ax, word[x+2]
mul word[y+2] ;Dx:Ax=Xh*Yh
add word[z+4], ax
adc word[z+6], dx
```

помоему должно получиться

[Ответить]

Борис
19-12-2010 14:29

с поправкой в дебагере получил правильный результат

[Ответить]

xrnd
20-12-2010 19:36

Поздравляю, первое упражнение правильно, второе вроде тоже (с поправкой) 🙂

[Ответить]

Amator
04-01-2011 00:35

Напишите программу для вычисления формулы z = (x·y) / (x + y). Все числа 16-битные целые со знаком.

```
use16 ;Генерировать 16-битный код
org100h ;програма начинается с адреса 100h
```

mov ax, [x] ;AX = x
mov bx, [y] ;BX = y
imul bx ;DX:AX = AX * BX = x * y
add cx, [x] ;CX = y + x
idiv cx ;AX = DX:AX \ CX = x*y \ y+x
mov [z], ax

mov 4cooh
inc 21h;
;---
x dw 14
y dw (-12)
z dw ?

проверь пожалуйста

[Ответить]

xrnd
10-01-2011 22:54

Похож на первый, но есть ошибки.

```
    add cx, [x] ;CX = y + x
```

В CX ничего не записывалось, поэтому там неизвестно что.
Непонятно, к чему прибавляется x.

Вместо:

```
    mov 4cooh
```

Должно быть:

```
    mov ax,4c00h
```

[Ответить]

Amator
13-01-2011 23:59

ТОЧНО ПРОШЛЯПИЛ:

```
use16 ;Генерировать 16-битный код
org100h ;програма начинается с адреса 100h

mov ax, [x] ;AX = x
mov bx, [y] ;BX = y
imul bx ;DX:AX = AX * BX = x * y
add bx, [x] ;BX = y + x
idiv bx ;AX = DX:AX \ BX = x*y \ y+x
mov [z], ax

mov ax, 4cOOh
inc 21h;
;-------------------------------------------------------------------
x dw 14
y dw (-12)
z dw ?
```

[Ответить]

xrnd
15-01-2011 00:58

Теперь правильно 🙂

Только 4C00h надо писать с двумя нулями, а не с буквой «O».

[Ответить]

Amator
04-01-2011 00:38

кажется мой вариант похож на первый

[Ответить]

Георгий
22-01-2011 21:17

; здравствуйте) скажите, есть ли здесь истина ?:) мне кажется, что жутко нерационально

```
; z=(x*y)/(x+y)
use16
org 100h

mov cx,[x]
imul cx,[y]
mov ax,cx

mov bx,word[x]
adc bx,word[y]
mov word[xday],bx

mov bx, word[x+2]
adc bx, word[y+2]
mov word[xiy+2],bx
mov bx,[xiy]

idiv bx
mov [z],ax
mov xiy,0

mov ax, 4C00h
int 21h
; ——— init data————————
x dw 37h
y dw 32h
xiy dw ?
z dw ?
```

[Ответить]

xrnd
22-01-2011 21:59

Истина есть, но её тут мало 🙂

Непонятно, где переменная xday?
Зачем складывать x и y в 2 этапа, если их размер — 16 бит, два байта?

word[x+2] — то же самое, что word[y] или просто [y]

[Ответить]

Георгий
22-01-2011 22:07

```
; упс…а так?
; z=(x*y)/(x+y)
use16
org 100h

mov cx,[x]
imul cx,[y]
mov ax,cx

mov bx,[x]
add bx,[y]

idiv bx
mov [z],ax

mov ax, 4C00h
int 21h
; ——— init data————————
x dw 37h
```

y dw 32h
z dw ?

[Ответить]

xrnd
22-01-2011 22:40

Так намного лучше.

Но одна ошибка осталась. Команда «idiv bx»делит DX:AX на BX. А что в DX? Скорее всего какой-то мусор, который будет мешать делению.

Лучше умножать так, чтобы получалось двойное слово в DX:AX.

[Ответить]

Георгий
05-02-2011 15:05

; тогда вероятно так? если нет, то напишите пожалуйста как должно быть
; z=(x*y)/(x+y)
use16
org 100h

mov ax,[x]
mov bx,[y]
imul bx; DX:AX=ax*bx

add bx,[x]; bx=bx(y)+x

idiv bx; ax= dx:ax / bx

mov [z],ax

```
mov ax, 4C00h
int 21h
; —— init data————
x dw 37h
y dw 32h
z dw ?
; ???
```

[Ответить]

xrnd
09-02-2011 15:25

Да, на этот раз всё правильно 🙂

[Ответить]

anton
02-02-2011 12:39

Привет ! Хочу перевести двойное слово в десятичную систему счисления для последующего вывода в десятичном виде на экран. Алгоритм простой: делю число на 10, остаток записываю как последнюю цифру, частное снова делю на 10, остаток — предпосл. цифра и т.д. пока частное не окажется меньше 10, тогда его записываю первой цифрой. Так вот, при первом же делении возникает переполнение, частное не помещается в слово. Можно ли решить проблему без использования расширенных регистров(eax,edx)? Что-то не соображу.

[Ответить]

xrnd
09-02-2011 14:52

Привет. Такое можно сделать, но придётся делить по частям. Алгоритм такой же, как в этой статье: http://asmworld.ru/isxodniki/preobrazovanie-64-bitnogo-chisla-v-stroku/.
Только нужно будет 2 деления, а не 4.

Первый этап — делится старшее слово, остаток добавляется к младшему слову (точнее становится старшей частью, достаточно оставить его в регистре DX)
Второй этап — делится младшее слово.

```
    mov bx,10               ;Делитель
    xor dx,dx               ;DX = 0
    mov ax,word[x+2]        ;Старшее слово
    div bx                  ;AX = (DX:AX)/10; DX = остаток
    mov word[x+2],ax        ;Старшая часть частного
    mov ax,word[x]          ;Младшее слово
    div bx                  ;AX = (DX:AX)/10; DX = остаток
    mov word[x],ax          ;Младшая часть частного
...
x dd 12345678
```

[Ответить]

anton
09-02-2011 20:38

Большое спасибо, xrnd ! 🙂 Как оказалось все просто, а я-то голову ломал.

[Ответить]

Philin
03-02-2011 11:59

Привет… по логике, вроде должно работать…)) ?
;z=(x*y)/(x+y)

```
use16
org 100h

mov ax,[x] ;ax=x=150
mov cx,[y] ;cx=y=-5
imul cx ;dx:ax=ax*cx=150*(-5)

add cx,[x] ;cx=cx+x=(-5)+150

div cx ;ax=dx:ax/cx
mov [z],ax

mov ax,4c00h
int 21h
;_____
x dw 150
y dw -5
z dw ?
```

[Ответить]

xrnd
09-02-2011 14:59

Привет. Практически всё правильно.
Только команду DIV нужно заменить на IDIV. Всё-таки числа со знаком 🙂

[Ответить]

plan4ik
31-03-2011 17:10

```
use16
org 100h
```

```
Start:
mov ax, word [x]
add ax, word [y]
mov word [z], ax

mov ax, word [x]
imul word [y]

idiv word [z]
mov [z], ax

mov ax, 4c00h
int 21h

; z = (x·y) / (x + y)

z dw 0
x dw 0x1122
y dw 0x3344
```

[Ответить]

xrnd
01-04-2011 15:59

Правильно 🙂

[Ответить]

annihilator
16-04-2011 16:51

Не могу понять, MUL работает только с регистрами AX и CX ?
У меня не получается заставить его работать с другими регистрами.

[Ответить]

xrnd
16-04-2011 16:58

Первый операнд должен быть в AL или AX (используется неявно)
Второй может быть в любом регистре или в памяти.
MUL BH ; AX = AL * BH
MUL SI ; DX:AX = AX * SI

[Ответить]

annihilator
16-04-2011 17:06

Почему при команде

mul cx

обнуляется значение регистра DX ?

[Ответить]

xrnd
16-04-2011 17:48

В DX записывается старшая часть произведения. Умножаешь 16×16 бит — в результате получается 32 бита.

[Ответить]

stud
25-04-2011 13:05

Если есть возможность. напишите программу с действиями 2-х байтных чисел. ADUC 812

XZ+2+XY-4X

[Ответить]

stud
26-04-2011 06:34

извините ошибся даже не 2 байтных , а 1 байтных чисел.

[Ответить]

Zipfer
18-05-2011 01:33

Не знаю как других, а меня сильно сбило, в плане понимания, вот эта строка
mov ch,0 ;Преобразуем t в слово в регистре CX
Я долго сидел и втыкал для чего это нужно. Если бы я знал что в след главе будет об этом рассказано (:
Еще, меня запутала запись, где вначале идет умножение, а только потом возведение в квадрат:
mul cx ;DX:AX = AX*CX = a*(t^2)
я никак не мог понять, как же так происходит, когда числа были умножены, а возведение происходит потом

[Ответить]

xrnd
19-05-2011 16:51

В CL находится t.
Сначала умножается AL на CL, потом ещё раз на CX.
CH обнуляется, поэтому получается 2 раза умножение на t 🙂

Не очень понятно получилось, но ты же разобрался ? 🙂

[Ответить]

Faltek
20-05-2011 23:01

Добрый день. Есть 1 проблемка: с умножением все понятно, а что делать с 32 разрядным делением? Можете написать алгоритм, пожалуйста?

[Ответить]

Ваш комментарий

Имя *

Почта (скрыта) *

Сайт

Добавить

Уведомить меня о новых комментариях по email.

Уведомлять меня о новых записях почтой.