Учебный курс. Часть 7. Системы счисления

Автор: xrnd | Рубрика: Учебный курс | 19-03-2010 |
Распечатать запись

Эта статья по большей части для совсем начинающих. Если вы хорошо разбираетесь в системах счисления, можете обратить внимание лишь на особенности синтаксиса ассемблера FASM в конце статьи.

На самом деле процессор работает только с двоичными числами, состоящими из единиц и нулей 🙂 В виде двоичных чисел хранятся и обрабатываются все данные и команды любой программы. Однако, двоичная запись чисел слишком громоздка и неудобна для человека, поэтому в программах на ассемблере используются и другие системы счисления: десятичная, шестнадцатеричная и восьмеричная.

Немного теории

Прежде всего разберёмся, в чём различие между системами счисления. Любое десятичное число можно представить в таком виде:

$$123_{10} = 1 \cdot 10^2 + 2 \cdot 10^1 + 3 \cdot 10^0$$

Индекс внизу обозначает, что это десятичное число. Цифра каждого разряда умножается на 10 в степени, равной номеру разряда, если считать с нуля справа налево. В более общем виде:

$$abc_r = a \cdot r^2 + b \cdot r^1 + c \cdot r^0,$$

где a, b и c — какие-то цифры, а r — основание системы счисления. Для десятичной системы $r = 10$, для двоичной — $r = 2$, для троичной $r = 3$ и т.д. Например, то же самое число в других системах:

$443_5 = 4 \cdot 5^2 + 4 \cdot 5^1 + 3 \cdot 5^0 = 4 \cdot 25 + 4 \cdot 5 + 3 \cdot 1 = 123_{10}$ (пятеричная система)

$173_8 = 1 \cdot 8^2 + 7 \cdot 8^1 + 3 \cdot 8^0 = 1 \cdot 64 + 7 \cdot 8 + 3 \cdot 1 = 123_{10}$ (восьмеричная система)

$1111011_2 = 1 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 1 \cdot 64 + 1 \cdot 32 + 1 \cdot 16 + 1 \cdot 8 + 0 \cdot 4 + 1 \cdot 2 + 1 \cdot 1 = 123_{10}$ (двоичная система)

Шестнадцатеричная система

В шестнадцатеричной системе для обозначения цифр больше 9 используются буквы A = 10, B = 11, C = 12, D = 13, E = 14, F = 15. Например:

$C7_{16} = 12 \cdot 16^1 + 7 \cdot 16^0 = 12 \cdot 16 + 7 \cdot 1 = 199_{10}$

Удобство шестнадцатеричной системы в том, что в неё очень легко можно переводить двоичные числа (и в обратную сторону тоже). Четыре разряда двоичного числа (тетрада) представляются одним разрядом шестнадцатеричного. Для перевода достаточно разбить число на группы по 4 бита и заменить каждую тетраду соответствующей шестнадцатеричной цифрой.

Двоичная тетрада	Шестнадцатеричная цифра
0000	0

0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Для записи одного байта требуется всего
2 шестнадцатеричные цифры:

$0101\ 1011_2 = 5B_{16}$

$0110\ 0000_2 = 60_{16}$

$1111\ 1111_2 = FF_{16}$

Восьмеричная система

Восьмеричная система также удобна для представления двоичных чисел, хотя она используется намного реже, чем шестнадцатеричная. Для быстрого перевода надо разбить

двоичное число на группы по 3 разряда (триплеты или триады).

Двоичная триада	Восьмеричная цифра
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Например: $001\ 110\ 101_2 = 165_8$

Синтаксис ассемблера FASM

По умолчанию, число в программе воспринимается ассемблером как десятичное. Чтобы обозначить двоичное число, необходимо к нему в конце добавить символ *'b'*. Восьмеричное число обозначается аналогично с помощью символа *'o'*. Для записи шестнадцатеричного числа FASM поддерживает 3 формы записи:

- перед числом записываются символы *'0x'* (как в C/C++);
- перед числом записывается символ *'$'* (как в Pascal);
- после числа записывается символ *'h'*. Если шестнадцатеричное число начинается с буквы, необходимо добавить в начале ноль (иначе непонятно, число это или имя метки).

Этот синтаксис используется как при объявлении данных, так и в командах. Вот примеры записи чисел во всех четырёх системах:

```
mov ax,537          ;Десятичная система

mov bl,11010001b    ;Двоичная система

mov ch,57o          ;Восьмеричная система

mov dl,$C2          ;\
mov si,0x013A       ; \
mov ah,18h          ; / Шестнадцатеричная система
mov al,0FFh         ;/

mov al,FFh          ;Ошибка!
```

Следующая часть »

Комментарии:

RoverWWWorm
19-03-2010 20:28

изучаю асм(компилятор FASM)и хочу продолжить самообучение, ну а так я новичок, жду появления других уроков, удачи автору сайта и пусть развивает свой сайт по FASM'u.
Буду рад если автор не будет большое внимание уделять на изучение окон и графических элементов винды, а только их основы… .
Если автору потребуются справочники и другие файлы, примеры по ассемблеру, то буду рад помочь.

[Ответить]

xrnd
21-03-2010 18:59

Сначала будут основы ассемблера, команды процессора без привязки к особенностям какой-то операционной системы. А потом и до окон доберёмся.

[Ответить]

RoverWWWorm
22-03-2010 15:01

Вот только побыстрей бы изучение окон закончились, а начались уроки типа с файлами или с железом что нибудь или др…

[Ответить]

ekslamer
25-03-2010 16:39

Так держать! С нетерпением жду новых уроков! СПАСИБО

[Ответить]

xrnd
26-03-2010 04:10

Рад, что хоть кто-то читает 🙂 Скоро будет следующая часть.

[Ответить]

bebe
08-04-2010 22:04

Очень помогли :))) Я некоторые нюансы только сейчас понял :)))))

[Ответить]

xrnd
10-04-2010 17:26

Отлично, я очень рад 🙂

[Ответить]

mc-black
08-05-2010 22:51

А вот и системы счисления. Есть одно пожелание — сделать на эту страницу перекрестную ссылку с «первой программы», которая ничего не делала, кроме как двигала значения по регистрам. Там не каждому новичку будет ясно, почему 255d = FFh

[Ответить]

xrnd
09-05-2010 16:55

Это хорошая идея. Я так и сделаю. Сначала мне не хотелось вообще писать про системы счисления, но потом я решил всё же написать. Вдруг кто-то не знает 🙂

[Ответить]

oleg
14-11-2010 23:49

У вас опечатка, для новичков повод потренироваться и не идти слепо за гуру 🙂

C3 = 12·16 + 3·16 = 12·16 + 3·1 = 195

[Ответить]

xrnd
15-11-2010 16:52

Исправил, спасибо.

[Ответить]

oleg
22-11-2010 00:51

Вот чуток добавлю, как переводить из десятичного в шестнадцатеричное число.

http://pixs.ru/showimage/dectohexpn_7538793_1237657.png

Для перевода из десятичного в двоичное, принцип ещё проще. Делим на 2, есть остаток пишим 1, нет пишем 0. А так всё тоже.

[Ответить]

xrnd
22-11-2010 13:12

Ага, или можно ещё стандартным калькулятором Windows воспользоваться 🙂

[Ответить]

oleg
22-11-2010 15:40

Так это понятно, но вот если на бумаге посчитать, то многие в панике. 🙂
Лучше знать, как это делать вручную на бумаге, а калькулятор

лишь средство убыстрения процесса. Тем более если курс призван дать познания ассемблера, пожалуй стоит понимать все процессы пересчёта между системами.

[Ответить]

xrnd
25-11-2010 02:09

Да, я согласен. Сам раньше так считал. И знать, как это делается, полезно.

[Ответить]

Миха
26-06-2011 02:23

Не убыстрения, а ускорения 🙂

[Ответить]

NimRoen
13-04-2011 11:37

Из книги Криса Касперски «Техника и философия хакерских атак»:

Как можно перевести произвольное число в двоичное? Для этого нужно поделить
его на 2 и записать остаток в младший разряд. И так до тех пор, пока делить
станет нечего. Хорошо, а как разделить, если нет калькулятора и даже счетов?
Разумеется, в столбик. Однако, этот способ несколько утомителен.

Куда проще запомнить (или вычислить в уме) ряд квадратов:

1 2 4 8 16 32 64 128

Ясно, что любое число от нуля до 255 представляет собой их сумму. Причем
каждая степень может встречаться только один раз. Покажем это на примере.
Допустим, нам необходимо узнать двоичное представление числа 99. Начинаем с
конца. Число нечетное, значит, в сумме фигурирует единица, т.е. младший бит
равен единице. Отнимаем от 99 один и получим 98. Если отнять еще и двойку,
то получим 96, а 96 == 32 + 64 как легко можно видеть. Итого в двоичном виде
это — 1100011. Конечно, это потребует определенных навыков устного счета, но
все же достаточно просто, чтобы не обращаться каждый раз к калькулятору.

Аналогично можно любое число из двоичного перевести в десятичное. Например:

1001b == 1+2*0+4*0+8*1 == 1+8 == 9

Все вышесказанное не в меньшей мере применимо и к шестнадцатеричным числам:

0x1 0x2 0x4 0x8 0x10 0x20 0x40 0x80

Причем все математические операции с такими круглыми числами делать в уме на
порядок проще

[Ответить]

xrnd
13-04-2011 19:36

Хорошая книга, да.

[Ответить]

Миха
26-06-2011 02:20

Думал, ничего нового не узнаю, а нет — нужно нолик перед 16-ричным числом ставить, спасибо 🙂 Собираюсь все уроки прочитать, вспомнить ассемблер захотелось 🙂

[Ответить]

Евгений
26-06-2011 10:03

Слушайте есть же программа наверно которая переводит мигом числа из двоичной системы в десятеричную и тд.?

[Ответить]

xrnd
02-07-2011 01:51

Есть. Например, обычный калькулятор Windows. Достаточно выбрать пункт меню «Вид->Инженерный» и появляются переключатели системы счисления 🙂

[Ответить]

Cross
10-07-2011 15:42

блин точняк 🙂

[Ответить]

Cross
10-07-2011 15:44

Люди а у всех же ассемблеров (fasm,masm,tasm) одинаковый синтаксис да?
и отличаются тока способом компиляции?

[Ответить]

xrnd
17-09-2011 14:51

Нет, синтаксис разный. Названия команд такие же, а в остальном много отличий. На мой взгляд, синтаксис FASM самый удачный.

[Ответить]

ArtemHermeus
21-06-2014 21:16

Замечательный сайт! Только сейчас я понял, почему именно 8-тиричная, и особенно 16-тиричная системы счисления так важны и полезны. Спасибо вам за Ваш труд.

[Ответить]

Ваш комментарий

Имя *

Почта (скрыта) *

Сайт

Добавить

Уведомить меня о новых комментариях по email.

Уведомлять меня о новых записях почтой.