Лекция 2. Позиционные системы счисления. Представление чисел в микропроцессорах.

Как представлять и хранить данные.
Как ими оперировать?

В микропроцессорной технике со времен фон Неймана в основном используется двоичная система счисления. При этом для человека привычнее десятичная система счисления по числу пальцев на руках. Поэтому представляют интерес правила перевода из двоичной системы счисления в десятичную и обратно и реализация операций с двоичным кодом. Помимо двоичной системы счисления в микропроцессорной технике часто применяется шестнадцатеричная, ее роль также будет рассмотрена в данной лекции.

Позиционные системы счисления

В этой системе счисления число есть строка цифр, каждой цифре приписан вес. Значение числа – взвешенная сумма его разрядов:

$1734 = 1\cdot10^3 + 7\cdot10^2 + 3\cdot10^1 + 4\cdot10^0$

Для представления дробных чисел можно использовать отрицательные степени числа 10:

$187.25 = 1\cdot10^2 + 8\cdot10^1 + 7\cdot10^0 + 2\cdot10^{-1} + 5\cdot10^{-2}$

Величина r = 10 называется основанием (base, radix) системы счисления. В общем случае:

$$D = \sum_{i=-n}^{p-1} d_i \cdot r^i = d_{p-1}d_{p-2}\ldots d_0 \;.\; d_{-1}d_{-2}\ldots d_{-n} \qquad \textbf{(1)}$$

где d_i – разряды числа D, $0 \le d_i < r$ или $d_i \in [0, r - 1]$,
d_{p-1} – старший разряд числа, most significant digit (MSD),
d_{-n} – старший разряд числа, least significant digit (LSD),
p – количество разрядов целой части числа,
n – количество разрядов дробной части числа.

Если n = 0, то число D – целое. Если n > 0, p > 0 и фиксированы, то такой формат представления называется числом с фиксированной точкой. Если n и p могут изменяться, то такой формат чисел называется числами с плавающей точкой. Формат чисел с плавающей точкой может быть разным. В процессорах x86 и AVR используется стандарт IEEE-754.

Роль двоичной системы счисления в микропроцессорной технике

Поговорим более подробно про двоичную систему счисления, т.е. рассмотрим случай r = 2.

Пример:

$110110_{(2)} = \boldsymbol{1}\cdot2^5 + \boldsymbol{1}\cdot2^4 + \boldsymbol{0}\cdot2^3 + \boldsymbol{1}\cdot2^2 + \boldsymbol{1}\cdot2^1 + \boldsymbol{0}\cdot2^0 = 32 + 16 + 4 + 2 + 1 = 55_{(10)}$

Значение двоичной системы в цифровой технике настолько велико, что разряды двоичного числа получили свое собственное название – бит (bit – BInary digiT).

Очевидно, что электронный регистр является естественным хранилищем двоичных чисел. **Таким образом, с помощью регистров можно хранить целые и вещественные числа (в виде чисел с фиксированной и плавающей точкой).**

Но есть особенности.

Пример представление числа $0.1_{(10)}$

Если представить $0.1_{(10)}$ как двоичную дробь, то она окажется периодической: $0.0(0011)_{(2)}$. Подробно рассмотреть пример дальше, когда будет про перевод.

Получается надо бесконечное число разрядов, а их нет. Очевидно, что при хранении данных в регистре величины p и n конечны. Сумма n + p называется разрядностью числа.

Исторически в микропроцессорной технике возникло понятие байта. Байт – единица хранения информации, соответствующая 8 двоичным разрядам, т.е. 8 битам. Кроме того, существует понятие полубайт или тетрада – 4 бита.

Слово – характерный размер данных, измеряемый в битах, для которого аппаратно реализована операция сложения (которое автоматически дает вычитании при использовании *дополнительного кода*). Помимо операции сложения, слово характеризует и другие важные характеристика процессора, о которых речь пойдет позже. Слово зависит от архитектуры процессора и является важной его характеристикой. Так, процессоры Intel Pentium I-IV 32-разрядные, Intel Core 2 – 64-разрядные. Микроконтроллер AVR, изучаемый в данном курсе – 8-разрядный или, что то же самое, 8-битный.

Разрядность задает диапазон возможных значений. Для целых 8-разрядных чисел возможны коды:

0000 0000, 0000 0001, …, 1111 1111

Сколько здесь кодовых комбинаций?

0, 1, … 255

Получается максимальный код плюс 1, с учетом комбинации 0000 0000. В двоичном коде:

1111 1111 + 1 = 1 0000 0000 = 2^8

В общем случае при n-разрядном представлении двоичного числа возможно 2^n кодовых комбинаций.

Задание.

Сколько комбинаций в слове у 32-битных процессоров?

Шестнадцатиразрядная система счисления

Разряд шестнадцатеричной системы счисления лежит в диапазоне от 0 до 15:

$d \in [0, 16) = [0, 15]$

Записи разрядов для диапазона 0-9 соответствуют десятичной системе, а цифры 10-15 задаются латинскими буквами A-F.

Шестнадцатеричный разряд	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
Десятичное значение разряда	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**

Пример

$AF3_{(16)} = A \cdot 16^2 + F \cdot 16^1 + 3 \cdot 16^0 = ?$

<u>Связь двоичной и шестнадцатиричной систем счисления</u>

$b_8 \cdot 2^8 + b_7 \cdot 2^7 + b_6 \cdot 2^6 + b_5 \cdot 2^5 + b_4 \cdot 2^4 + b_3 \cdot 2^3 + b_2 \cdot 2^2 + b_1 \cdot 2^1 + b_0 \cdot 2^0 =$
$= (b_8 \cdot 2^0) \cdot 2^8 + (b_7 \cdot 2^3 + b_6 \cdot 2^2 + b_5 \cdot 2^1 + b_4 \cdot 2^0) \cdot 2^4 + (b_3 \cdot 2^3 + b_2 \cdot 2^2 + b_1 \cdot 2^1 + b_0 \cdot 2^0) \cdot 2^0 =$
$= (\ \dots\ ) \cdot 16^2 + (\ \dots\ ) \cdot 16^1 + (\dots) \cdot 16^0$

Можно показать, что выражения в скобках, например $(b_3 \cdot 2^3 + b_2 \cdot 2^2 + b_1 \cdot 2^1 + b_0 \cdot 2^0)$ лежат в диапазоне [0, 15], т.е. удовлетворяет требованию к разрядам 16-разрядного числа (см. формулу (1))

Пример

$1101\ 0101_{(2)} = 13_{(10)}\ \ 5_{(10)} = D_{(16)}\ \ 5_{(16)} = D5_{(16)}$

Очевидно, легко сделать и обратное преобразование.

Пример

$AF3_{(16)} = 1010\ 1111\ 0011_{(2)}$

Из примеров ясно, что ценность 16-разрядной системы счисления – в ее близости в двоичной. Фактически, она в основном используется программистами (не микропроцессорами!), когда надо задать двоичное число большой разрядности в компактном виде.

В общем случае для перевода ***n*** – разрядного двоичного число надо $\lceil n/4 \rceil$ шестнадцатеричных разрядов. Общая формула перевода имеет вид:

$$\sum_{i=0}^{n-1} b_i \cdot 2^i = \sum_{i=0}^{\lceil n/4 \rceil - 1} \left(\sum_{k=0}^{3} b_{k+4i} \cdot 2^{k+4i} \right) \cdot 16^i$$

Тетрады двоичного числа $\left(\sum_{k=0}^{3} b_{k+4i} \cdot 2^{k+4i} \right)$, записанные в 16-ричном виде, образуют запись 16-ричного представления числа.

Общий подход к преобразованию чисел между системами счисления.

Мы рассмотрели 3 системы счисления с основаниями 2, 10, 16. В работе надо уметь переводить из каждой системы в любую другую, т.е. каждой клетке таблицы 1 должно соответствовать некоторое правило перевода (кроме диагональных клеток). Некоторые клетки мы уже можем заполнить.

1. **Bin -> Dec**, уже рассмотрен выше, просто расписать по формуле (1)
2. **Bin -> Hex** – есть простой способ, рассмотренный выше
3. Попробуем **Dec -> Hex** расписать по формуле (1)

$135_{(10)} = 1 \cdot A^2 + 3 \cdot A^1 + 5 \cdot A^0$

Получается сложно, потому что для перевода придется вести вычисления в шестнадцатеричной системе счисления. Другой способ – подгонка. Нам необходимо подобрать разряды d_2, d_1, d_0 в разложении:

$135_{(10)} = d_2 \cdot 16^2 + d_1 \cdot 16^1 + d_0 \cdot 16^0$

Сколько раз по 16^2=256 содержится в 135? Ни одного, поэтому разряд $d_2 = 0$. Разряд d_1 указывает сколько раз по 16^1 содержится в 135. Очевидно, 8. Дальше легко определить d_0.

(Почему надо идти сверху и брать максимальное значение?)

Табл. 1. Перевод между системами счисления

Из	В		
	Bin(ary), 2	Dec(imal), 10	Hex(ademical), 16
Bin(ary), 2	x	формула (1)	Связь 2<->16
Dec(imal), 10	?	x	подбор
Hex(ademical), 16	Связь 2<->16	формула (1)	x

Остался перевод dec->bin. Рассмотрим для

Общее правило перевода

Пусть задано число в некоторой системе счисления. Требуется его перевести в систему счисления с основанием r. Рассмотрим отдельно перевод его целой и дробной части. В соответствии с (1) *p* разрядов задают целую часть числа, а *n* разрядов – дробную. Целая часть числа в нужной нам системе с основанием *r* в соответствии с (1) имеет вид:

$$D = \sum_{i=0}^{p-1} d_i \cdot r^i = d_{p-1} \cdot r^{p-1} + d_{p-2} \cdot r^{p-2} + \ldots + d_1 \cdot r^1 + \quad \textbf{(2)}$$

Задача перевода сводится к отысканию величин d_i. Разделим D на r и определим чему равны частное и остаток.

$$\frac{D}{r} = \underbrace{d_{p-1} \cdot r^{p-2} + d_{p-2} \cdot r^{p-3} + \ldots + d_1 \cdot r^0}_{Q} + \frac{d_0}{r}$$

Поскольку $d_0 < r$, то остаток от деления равен d_0 (именно d_0, а не дробь $\frac{d_0}{r}$), а частное равно Q. Следовательно, при делении числа в «старой» системе счисления на основание «новой» системы счисления, остаток от деления равен младшему разряду числа в «новой» системе счисления. Аналогично d_1 будет остатком от деления Q на r. Повторяя процедуру, пока $Q \neq 0$, получим все разряды числа в системе счисления с основанием r в обратном порядке.

Рассмотрим перевод дробной части числа:

$$D = \sum_{i=-1}^{-n} d_i \cdot r^i = d_{-1} \cdot r^{-1} + d_{-2} \cdot r^{-2} + \ldots + d_{-n} \cdot r^{-n} \quad \textbf{(3)}$$

Умножим D на r:

$$D \cdot r = P = d_{-1} + d_{-2} \cdot r^{-1} + \ldots + d_{-n} \cdot r^{-n+1} \quad \textbf{(4)}$$

Можно показать, что сумма всех слагаемых выражения, кроме d_{-1} строго меньше 1. Следовательно, целая часть числа P равна d_{-1}. Обнулим целую часть P и снова умножим на r, получим d_{-2} и так далее.

Отметим, что мы не сделали никаких предположений относительно основания системы r, поэтому описанный способ подходит не только для dec -> bin, но и, например, для dec -> hex.

Примеры:

$13_{(10)} = ?_{(2)}$

$0.1_{(10)} = ?_{(2)}$ – убедиться, что это действительно периодический код