Курс «Основы программирования» Занятие №0 (продолжение)

Григорин Александр

Санкт-Петербург 2017 г.

Кодирование информации (продолжение)

- Компьютер оперирует потоком 0 и 1:
- 10101110001111000000111000001010001110011 0
- Этот поток нужно превратить в понятную для человека форму, например в числовую: 1234567890
- Правила преобразования потока 0 и 1 в понятный вид называется «Кодированием»

Позиционные системы счисления ч.1

- Человек оперирует 10 цифрами: 0, 1 .. 9
- Допустим есть число 5479, что это значит?
- 5479 = **5** * 1000 + **4** * 100 + **7** * 10 + **9** * 1
- Т.е. в числе 5 тысяч, 4 сотни, 7 десятков и 9 единиц
- Можно записать по другому:
- $5479 = 5 * 10^3 + 4 * 10^2 + 7 * 10^1 + 9 * 10^0$

Степень	3	2	1	0
Число	5	4	7	9

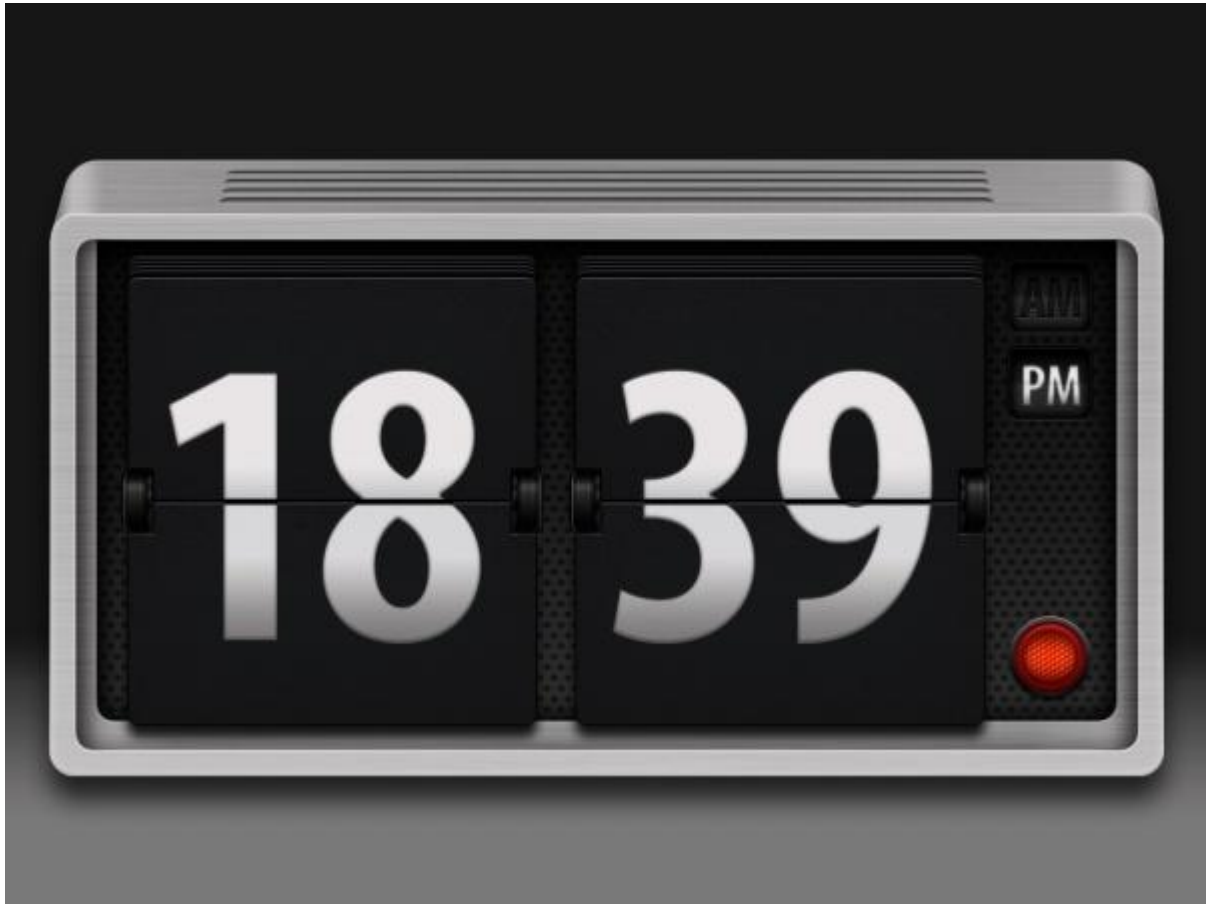
- Число можно представить в следующем виде: $a_n*10^n + a_{n-1}*10^{n-1}+\ldots+a_1*10^1+a_0*10^0$
- 10 – количество цифр в алфавите – основание системы счисления
- Вес цифры зависит от позиции в числе, следовательно система эта позиционная.

Позиционные системы счисления ч.2

- Представьте, что вы к цифре 0 прибавляете цифру 1 несколько раз 0+1=1, 1+1=2, 2+1=3 и т.д.
- Когда вы сделаете 9+1, то получите 10.
- Что это значит?
- После девятки цифр нет. Т.е. некий счётчик единиц переполнен. И вы переходите к счётчику десятков, добавив туда единицу (сделали перенос). Теперь у вас 0 единиц и 1 десяток.
- $10 = 1 * 10^1 + 0 * 10^0$

Позиционные системы счисления ч.3

- Близкая аналогия это перекидные часы



Позиционные системы счисления ч.4

- 10 – основание системы счисления (базис) = количеству цифр. Это удобно для человека.
- Для компьютера удобно основание системы счисления 2. Т.е. две цифры – 0 и 1.
- Базис разный, но правила работы одни и те же!
- Есть число 0_2. Будем прибавлять к нему 1_2.
- $0_2+1_2=1_2$, $1_2+1_2=10_2$
- После 1 нет цифры. Т.е. произошло «переполнение сверху». И мы снова по кругу пришли к нулю. При этом в следующей позиции прибавилась 1.
- Число 010_2 это не 10 в понимании человека!
- Число 010_2 это 2 в десятичной системе!

Позиционные системы счисления ч.5

Десятичная	Двоичная	Проверка
0	0000_2	
1	0001_2	
2	0010_2	$1*2^1+0*2^0 = 2+0=2$
3	0011_2	$1*2^1+1*2^0 = 2+1=3$
4	0100_2	$1*2^2+0*2^1+0*2^0 = 4+0+0=4$
5	0101_2	
6	0110_2	
7	0111_2	$1*2^2+1*2^1+1*2^0 = 4+2+1=7$
8	1000_2	
9	1001_2	
10	1010_2	$1*2^3+0*2^2+1*2^1+0*2^0 = 8+0+2+0=10$
11	1010_2	

Устройство памяти ч.1

- Память можно представить строкой:

№ ячейки	7	6	5	4	3	2	1	0
Данные	1	1	1	1	1	1	1	1

- Какое число здесь хранится?
- $1*2^7+1*2^6+1*2^5+1*2^4+1*2^3+1*2^2+1*2^1+1*2^0=$

$=128+64+32+16+8+4+2+1=255_{10}$

- Одна ячейка хранит данные, которые могут находится в двух состояниях – 0 или 1, т.е. хранить 1 бит данных. Bit – Binary digit (двоичная цифра)
- Но оперировать с каждым отдельным битом самостоятельно сложно (так как нужно хранить гигантское количество № ячеек, т.е. их адресов)
- Поэтому ячейки битов группируют.

Устройство памяти ч.2

- Биты группируют в байты (byte).
- 1 Байт это 8 битов подряд

Байт	0								1							
Бит	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
Значение	1	1	0	0	1	0	1	0	0	1	1	1	0	1	0	0

- Байт это минимально адресуемая величина.
- Компьютер обращается к байтам и только затем оперирует с битами.
- Диапазон байта – от 00000000 до 11111111.
- Число 11111111_2 это как мы узнали 255_{10}, Т.е. 1 байт может хранить 256 состояний (2^8) считая 0.
- Количество состояний = $2^{\text{кол-во ячеек}}$

Устройство памяти ч.3

- В 1 Байте информации можно сохранить числа от 0 до 255 включительно. Если нужно больше?
- Тогда формируем цепочку из 2-х байтов

0								1							
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- Здесь закодировано число 65 535
- Т.е. в двух байтах (16 бит) можно закодировать 65536 состояний (не забудьте про 0)!
- Сдвоенный байт (16 бита) называется «слово» (Word)
- Четыре байта (32 бита) это «двойное слово» (Double word)
- 8 байт это «четверное слово» (Quad word)

Устройство памяти ч.4

Кол-во бит	Кол-во байт				
8	1				Byte
16	2				Word
32	4			Double word	
64	8	Quad word			

1 бит информации хранит 2 состояния (0 или 1) - 2^1
8 бит – $2^8 = 256$ состояний
16 бит – $2^{16} = 65\,536$
32 бит – $2^{32} = 4\,294\,967\,296$
64 бит – $2^{64} = 18\,446\,744\,073\,709\,551\,616$
ВАЖНО ПОМНИТЬ, что 0 это тоже число и диапазон чисел всегда равен:
От 0 до 2^n-1, где n это число бит. Байт не может хранить число 256!!!

№	X	7	6	5	4	3	2	1	0
+	X	1	1	1	1	1	1	1	1
	X	0	0	0	0	0	0	0	1
=	1	0	0	0	0	0	0	0	0

ПЕРЕПОЛНЕНИЕ СВЕРХУ

Устройство памяти ч.5

Адрес байта	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Данные	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
Тип	b	w		w		qw								dw				b	b	dw				w		w	

- Byte – b
- Word – w
- Double word – dw
- Quad word – qw
- В ячейке по адресу 0 начинаются данные типа byte
- В ячейке по адресу 5 начинаются данные типа quad word
- Компьютер обращается к ячейке памяти по адресу
- Чтобы определить протяжённость ячейки - к типу.

Устройство памяти ч.6 (пример)

Адрес байта	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**15**	**15**
Данные	133		*	*	11 246 576 877								4 294 967 296			
Тип	w		*	*	qw								dw			

В таблице есть ошибки. Ищем их самостоятельно. Это и есть Д/з.

Устройство памяти ч.7

8 бит это $2^8 = 256$ состояний

К примеру я хочу сохранить в памяти компьютера некую величину и выделяю под её хранение 1 байт. Значит я могу там хранить целые числа от 0 до 255. Число $255_{10} = 11111111_2$. Если к 11111111_2 прибавить 1_2, то все данные обнулятся и в последнем не существующем знакоместе будет единица. Это «переполнение сверху» - overflow

№	X	7	6	5	4	3	2	1	0
+	X	1	1	1	1	1	1	1	1
	X	0	0	0	0	0	0	0	1
=	1	0	0	0	0	0	0	0	0

Если из числа 00000000_2 типа byte вычесть 1_2, то тогда результат будет 11111111_2, произошло «переполнение снизу» - downflow.

Устройство памяти ч.8