

Компьютерная арифметика

Алексей Владыкин

СПбГУ ИТМО

21 сентября 2009

Биты, байты

- Бит — единица информации: 0 или 1
- Байт — 8 бит, минимальная адресуемая ячейка памяти
- 32-битный процессор оперирует числами шириной 1, 2, 4 байт
- 64-битный — 1, 2, 4, 8 байт

Перевод числа в P -ную систему счисления и обратно

- Дано число в десятичной системе счисления; хотим перевести в P -ную.
- Делим с остатком на P , выписывая остатки от деления справа налево.
- Дано число в P -ной системе счисления; хотим перевести в десятичную.
- Суммируем цифры $a_i|_{i=0}^k$ с весами P^i .

Двоичная и шестнадцатеричная системы счисления

- Двоичная система счисления: язык компьютера, но слишком громоздка
- Шестнадцатеричная система счисления: удобно переводить в двоичную и обратно, хорошо «ложится» на байты, компактна

0	0000	4	0100	8	1000	C	1100
1	0001	5	0101	9	1001	D	1101
2	0010	6	0110	A	1010	E	1110
3	0011	7	0111	B	1011	F	1111

Логические операции

- Сколько всего одноместных логических операций?

Логические операции

- Сколько всего одноместных логических операций?
- Ответ: 4

Логические операции

- Сколько всего одноместных логических операций?
- Ответ: 4

- Сколько всего двуместных логических операций?

Логические операции

- Сколько всего одноместных логических операций?
- Ответ: 4

- Сколько всего двуместных логических операций?
- Ответ: 16

Логические операции

- Сколько всего одноместных логических операций?
- Ответ: 4

- Сколько всего двуместных логических операций?
- Ответ: 16

- Используются: not, and, or, xor

Работа с отдельными битами числа

Инвертировать : value xor mask
Установить в 0 : $\text{value and not mask}$
Установить в 1 : value or mask
Проверить 1 : value and mask

Представление отрицательных чисел

- Прямой код: прост для понимания, но в нем трудно производить арифметические действия
- Дополнительный код (two's complement): не очевиден, но арифметика реализуется естественным образом
- Алгоритм получения дополнительного кода: инвертировать все биты, прибавить единицу

Представление символьной информации

- Процессор ничего не знает о символах; оперирует только числами.
- Кодлируем символы числами.
- ASCII — American Standard Code for Information Interchange

	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	.A	.B	.C	.D	.E	.F
0.	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	TAB	LF	VT	FF	CR	SO	SI
1.	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2.		!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3.	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4.	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5.	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6.	'	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7.	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

ASCII art

Star Wars in ASCII: <http://www.asciimation.co.nz/>

```

  /~\
 ( oo|   He says he's
  _\=/-   found the main
         controls to
 /       \   the power beam
//|/. \|\ \
|| \_/_/ || holding the
|| \| /| ||   ship here.
#  \_ _/ #
  | | |
  | | |
  [] | []
  | | |
  /_ ] _[ \_
-----

      _ _ _
    / ( ) \
  _| _ _ _ | _
  | | == | |
  | _ | 0 | _ | /
  | | 0 | | | |-----
  | | _ * _ | | | (*)
  | ~ \ _ _ / ~ | |
  / = \ / = \ / = \ |
  [ ] _ [ ] _ [ ] _ |-----

```

Представление символьной информации

- Разнообразные варианты для кодов 0x80–0xFF:
CP866, KOI-8, Windows-1251
- Проблема неправильной кодировки: «бНОПНЯ»
- Стандарт Unicode: кодовые позиции + кодировки UTF
ASCII: U+0000 – U+00FF,
Кириллица: U+0400 – U+04FF, ...
- Различные подходы к хранению длины строки: C vs Pascal.

UTF-8

U+000000-U+00007F	0.....
U+000080-U+0007FF	110..... 10.....
U+000800-U+00FFFF	1110.... 10..... 10.....
U+010000-U+1FFFFFF	11110... 10..... 10..... 10.....