

# 计算机中的数据表示

彭飞

东南大学电气工程学院



- 在计算机内部，数据都是以二进制的形式存储和运算的。

- 位

- 位(bit)简写为b，音译为比特，是计算机存储数据的最小单位，是二进制数据中的一个位，一个二进制位只能表示0或1两种状态，要表示更多的信息，就得把多个位组合成一个整体，每增加一位，所能表示的信息量就增加一倍。

- 字节

- 字节(Byte)简记为B，规定一个字节为8位，即 $1\text{Byte} = 8\text{bit}$ 。字节是计算机数据处理的基本单位，并主要以字节为单位解释信息。每个字节由8个二进制位组成。通常，一个字节可存放一个ASCII码，两个字节存放一个汉字国际码

- 字

- 字(Word)是计算机进行数据处理时，一次存取、加工和传送的数据长度。一个字通常由一个或若干个字节组成，由于字长是计算机一次所能处理信息的实际位数，所以，它决定了计算机数据处理的速度，是衡量计算机性能的一个重要标识，字长越长，性能越好。
  - 计算机型号不同，其字长是不同的，常用的字长有8位、16位、32位和64位。



## • 机器数

- 在计算机内部，任何信息都以二进制代码表示(即0与1的组合来表示)。一个数在计算机中的表示形式，称为机器数。

## • 真值

- 机器数所对应的原来的数值称为真值



- **原码**

- 原码表示法即用机器数的最高位代表符号(若为0, 则代表正数, 若为1, 则代表负数), 数值部分为真值的绝对值的一种表示方法。

- **反码**

- 正数的反码不变, 负数的反码是对应正数的原码取反

- **补码 (计算机中使用的编码)**

- 正数的补码不变, 负数的补码是其对应正数的反码+1。



## 1.3 计算机中的数据表示

十进制	+73	-73	+127	-127	+0	-0
二进制(真值)	+1001001B	-1001001B	+1111111B	-1111111B	+0000000B	-0000000B
原码	01001001B	11001001B	01111111B	11111111B	0000000B	1000000B
反码	01001001B	10110110B	01111111B	10000000B	00000000B	11111111B
补码	01001001B	10110111B	01111111B	10000001B	0000000B	00000000B



# • 十进制与二进制之间的转换

例如把52换算成二进制数，计算结果如图：

2	52	.....	0
2	26	.....	0
2	13	.....	1
2	6	.....	0
2	3	.....	1
	1	.....	1

例如要把-52换算成二进制：

- 1.先取得52的二进制：00110100
- 2.对所得到的二进制数取反：11001011
- 3.将取反后的数值加一即可：11001100

# • 二进制与十六进制之间的转换



# • 十进制和十六进制转换



- 
- **使用计算器进行数制转换**
  - **MODE**
  - **BASE-N**
  - **SHIFT + DEC十进制**
  - **SHIFT + HEX十六进制**
  - **SHIFT + BIN二进制**



- **BCD码**

- 常用8421 BCD码。
- Gray码(相邻的2个数只有一位不同)

- **ASCII**

- 7位二进制编码
- 表示字母、数字

- **Unicode**

- 16位编码
- 对世界上所有语言
- <https://unicode-table.com/>





# • ASCII码表

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	<b>NUL</b> (null)	32	20	040	&#32;	<b>Space</b>	64	40	100	&#64;	<b>@</b>	96	60	140	&#96;	<b>`</b>
1	1	001	<b>SOH</b> (start of heading)	33	21	041	&#33;	<b>!</b>	65	41	101	&#65;	<b>A</b>	97	61	141	&#97;	<b>a</b>
2	2	002	<b>STX</b> (start of text)	34	22	042	&#34;	<b>"</b>	66	42	102	&#66;	<b>B</b>	98	62	142	&#98;	<b>b</b>
3	3	003	<b>ETX</b> (end of text)	35	23	043	&#35;	<b>#</b>	67	43	103	&#67;	<b>C</b>	99	63	143	&#99;	<b>c</b>
4	4	004	<b>EOT</b> (end of transmission)	36	24	044	&#36;	<b>\$</b>	68	44	104	&#68;	<b>D</b>	100	64	144	&#100;	<b>d</b>
5	5	005	<b>ENQ</b> (enquiry)	37	25	045	&#37;	<b>%</b>	69	45	105	&#69;	<b>E</b>	101	65	145	&#101;	<b>e</b>
6	6	006	<b>ACK</b> (acknowledge)	38	26	046	&#38;	<b>&amp;</b>	70	46	106	&#70;	<b>F</b>	102	66	146	&#102;	<b>f</b>
7	7	007	<b>BEL</b> (bell)	39	27	047	&#39;	<b>'</b>	71	47	107	&#71;	<b>G</b>	103	67	147	&#103;	<b>g</b>
8	8	010	<b>BS</b> (backspace)	40	28	050	&#40;	<b>(</b>	72	48	110	&#72;	<b>H</b>	104	68	150	&#104;	<b>h</b>
9	9	011	<b>TAB</b> (horizontal tab)	41	29	051	&#41;	<b>)</b>	73	49	111	&#73;	<b>I</b>	105	69	151	&#105;	<b>i</b>
10	A	012	<b>LF</b> (NL line feed, new line)	42	2A	052	&#42;	<b>*</b>	74	4A	112	&#74;	<b>J</b>	106	6A	152	&#106;	<b>j</b>
11	B	013	<b>VT</b> (vertical tab)	43	2B	053	&#43;	<b>+</b>	75	4B	113	&#75;	<b>K</b>	107	6B	153	&#107;	<b>k</b>
12	C	014	<b>FF</b> (NP form feed, new page)	44	2C	054	&#44;	<b>,</b>	76	4C	114	&#76;	<b>L</b>	108	6C	154	&#108;	<b>l</b>
13	D	015	<b>CR</b> (carriage return)	45	2D	055	&#45;	<b>-</b>	77	4D	115	&#77;	<b>M</b>	109	6D	155	&#109;	<b>m</b>
14	E	016	<b>SO</b> (shift out)	46	2E	056	&#46;	<b>.</b>	78	4E	116	&#78;	<b>N</b>	110	6E	156	&#110;	<b>n</b>
15	F	017	<b>SI</b> (shift in)	47	2F	057	&#47;	<b>/</b>	79	4F	117	&#79;	<b>O</b>	111	6F	157	&#111;	<b>o</b>
16	10	020	<b>DLE</b> (data link escape)	48	30	060	&#48;	<b>0</b>	80	50	120	&#80;	<b>P</b>	112	70	160	&#112;	<b>p</b>
17	11	021	<b>DC1</b> (device control 1)	49	31	061	&#49;	<b>1</b>	81	51	121	&#81;	<b>Q</b>	113	71	161	&#113;	<b>q</b>
18	12	022	<b>DC2</b> (device control 2)	50	32	062	&#50;	<b>2</b>	82	52	122	&#82;	<b>R</b>	114	72	162	&#114;	<b>r</b>
19	13	023	<b>DC3</b> (device control 3)	51	33	063	&#51;	<b>3</b>	83	53	123	&#83;	<b>S</b>	115	73	163	&#115;	<b>s</b>
20	14	024	<b>DC4</b> (device control 4)	52	34	064	&#52;	<b>4</b>	84	54	124	&#84;	<b>T</b>	116	74	164	&#116;	<b>t</b>
21	15	025	<b>NAK</b> (negative acknowledge)	53	35	065	&#53;	<b>5</b>	85	55	125	&#85;	<b>U</b>	117	75	165	&#117;	<b>u</b>
22	16	026	<b>SYN</b> (synchronous idle)	54	36	066	&#54;	<b>6</b>	86	56	126	&#86;	<b>V</b>	118	76	166	&#118;	<b>v</b>
23	17	027	<b>ETB</b> (end of trans. block)	55	37	067	&#55;	<b>7</b>	87	57	127	&#87;	<b>W</b>	119	77	167	&#119;	<b>w</b>
24	18	030	<b>CAN</b> (cancel)	56	38	070	&#56;	<b>8</b>	88	58	130	&#88;	<b>X</b>	120	78	170	&#120;	<b>x</b>
25	19	031	<b>EM</b> (end of medium)	57	39	071	&#57;	<b>9</b>	89	59	131	&#89;	<b>Y</b>	121	79	171	&#121;	<b>y</b>
26	1A	032	<b>SUB</b> (substitute)	58	3A	072	&#58;	<b>:</b>	90	5A	132	&#90;	<b>Z</b>	122	7A	172	&#122;	<b>z</b>
27	1B	033	<b>ESC</b> (escape)	59	3B	073	&#59;	<b>;</b>	91	5B	133	&#91;	<b>[</b>	123	7B	173	&#123;	<b>{</b>
28	1C	034	<b>FS</b> (file separator)	60	3C	074	&#60;	<b>&lt;</b>	92	5C	134	&#92;	<b>\</b>	124	7C	174	&#124;	<b> </b>
29	1D	035	<b>GS</b> (group separator)	61	3D	075	&#61;	<b>=</b>	93	5D	135	&#93;	<b>]</b>	125	7D	175	&#125;	<b>}</b>
30	1E	036	<b>RS</b> (record separator)	62	3E	076	&#62;	<b>&gt;</b>	94	5E	136	&#94;	<b>^</b>	126	7E	176	&#126;	<b>~</b>
31	1F	037	<b>US</b> (unit separator)	63	3F	077	&#63;	<b>?</b>	95	5F	137	&#95;	<b>_</b>	127	7F	177	&#127;	<b>DEL</b>



自然二进制码与格雷码的对照表：

十进制数	自然二进制数	格雷码	十进制数	自然二进制数	格雷码
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000



# • 32位ARM C语言编程中所用的数据类型

数据类型	位数	范围（有符号）	范围（无符号）
char, int8_t, uint8_t	8	-128 – 127	0 – 255
short, int16_t, uint16_t	16	-32768 – 32767	0 – 65535
int, int32_t, uint32_t	32	$-2^{31} - 2^{31}-1$	$0 - 2^{32} - 1$
long	32	$-2^{31} - 2^{31}-1$	$0 - 2^{32} - 1$
Long long, int64_t, uint64_t	64	$-2^{63} - 2^{63}-1$	$0 - 2^{64} - 1$
float	32	$-3.403 \times 10^{38} - 3.403 \times 10^{38}$	
Double, long double	64	$-1.798 \times 10^{308} - 1.798 \times 10^{308}$	



## • 小数的表示方法:

### – 直观的小数表示法

### – 定点数

- 约定数值的小数点固定在某一位置，称为定点表示法，简称为定点数。

### – 浮点数

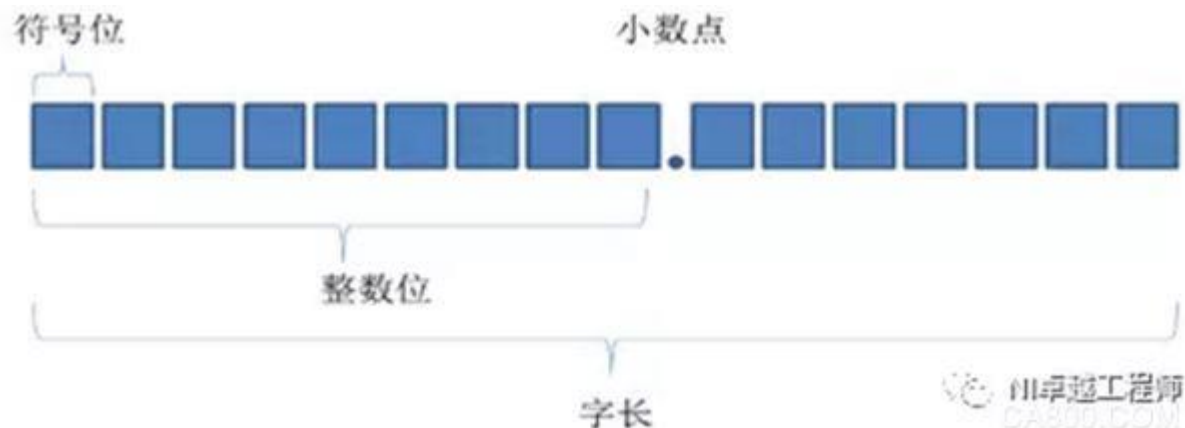
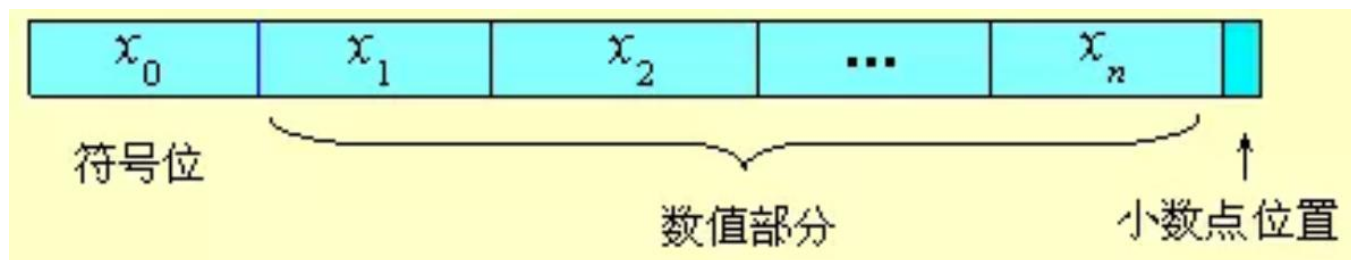
- 小数点位置可以任意浮动，称为浮点表示法，简称为浮点数



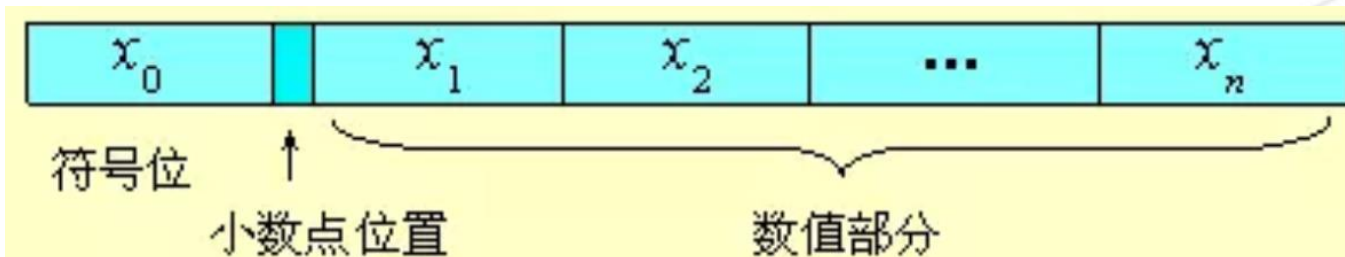
- **直观的小数表示法**
- **用10000表示10.000， 10001表示10.001**
- **用10000表示100.00， 10001表示100.01**
- **.....**
- **表示方法自己定义，不统一，注意取值范围**
- **加减运算，不受影响**
- **乘除运算，受影响，需要多余乘除法**



# • 定点数



**Qf:** 称作“Q 格式”，Qn 表示 n 个小数字。



_iq15	-65536	65535.999 969 482	0.000 030 518
_iq14	-131072	131071.999 938 965	0.000 061 035
_iq13	-262144	262143.999 877 930	0.000 122 070
_iq12	-524288	524287.999 755 859	0.000 244 141
_iq11	-1048576	1048575.999 511 719	0.000 488 281
_iq10	-2097152	2097151.999 023 437	0.000 976 563
_iq9	-4194304	4194303.998 046 875	0.001 953 125
_iq8	-8388608	8388607.996 093 750	0.003 906 250
_iq7	-16777216	16777215.992 187 500	0.007 812 500
_iq6	-33554432	33554431.984 375 000	0.015 625 000
_iq5	-67108864	67108863.968 750 000	0.031 250 000
_iq4	-134217728	134217727.937 500 000	0.062 500 000
_iq3	-268435456	268435455.875 000 000	0.125 000 000
_iq2	-536870912	536870911.750 000 000	0.250 000 000
_iq1	-1073741824	1 073741823.500 000 000	0.500 000 000

```

typedef long    _iq12;    /* Fixed point data type: Q12 format    */
typedef long    _iq11;    /* Fixed point data type: Q11 format    */
typedef long    _iq10;    /* Fixed point data type: Q10 format    */
typedef long    _iq9;     /* Fixed point data type: Q9 format     */
typedef long    _iq8;     /* Fixed point data type: Q8 format     */
typedef long    _iq7;     /* Fixed point data type: Q7 format     */
typedef long    _iq6;     /* Fixed point data type: Q6 format     */
typedef long    _iq5;     /* Fixed point data type: Q5 format     */
typedef long    _iq4;     /* Fixed point data type: Q4 format     */
typedef long    _iq3;     /* Fixed point data type: Q3 format     */
typedef long    _iq2;     /* Fixed point data type: Q2 format     */
typedef long    _iq1;     /* Fixed point data type: Q1 format     */

```





# • 使用IQMath库进行运算

Arithmetic Functions							
IQNmpy	1-30	~ 6	32 bits	NA	IQN*IQN	IQN	INTRINSIC
IQNrmpy	1-30	17	32 bits	13 words	IQN*IQN	IQN	
IQNrsmPy	1-30	21	32 bits	21 words	IQN*IQN	IQN	
IQNmpyl32	1-30	~ 4	32 bits	NA	IQN*long	IQN	C-MACRO
IQNmpyl32int	1-30	22	32 bits	16 words	IQN*long	long	
IQNmpyl32frac	1-30	24	32 bits	20 words	IQN*long	IQN	
IQNmpylQX		~ 7	32 bits	NA	IQN*IQN	IQN	INTRINSIC
IQNdiv	1-30	63	28 bits	71 words	IQN/IQN	IQN	

Function Name	IQ Format	Execution Cycles	Accuracy (in bits)	Program Memory (words)	Input format	Output format	Remarks
Trigonometric Functions							
IQNasin	1-29	154		82 words	IQN	IQN	Note A
IQNsin	1-29	46	30 bits	49 words	IQN	IQN	
IQNsinPU	1-30	40	30 bits	41 words	IQN	IQN	
IQNacos	1-29	170		93 words	IQN	IQN	Note A
IQNcos	1-29	44	30 bits	47 words	IQN	IQN	
IQNcosPU	1-30	38	29 bits	39 words	IQN	IQN	
IQNatan2	1-29	109	26 bits	123 words	IQN	IQN	
IQNatan2PU	1-29	117	27 bits	136 words	IQN	IQN	
IQatan	1-29	109	25 bits	123 words	IQN	IQN	





## • 浮点数

float的规格化表示为： $\pm 1.f \times 2^{E-127}$ ，其中， $f$ 是尾数， $E$ 是指数。

[illegible]

比如十进制数123.125，其二进制表示为：1111011.001，规格化表示为： $1.111011001 \times 2^6$   
也就是 $1.111011001 \times 2^{133-127}$ ， $f = 111011001$ ， $E = 133 = 10000101$ ，图示如下：

+	E = 6 + 127 = 133								尾数：111011001（后面补0到23位）																						
0	1	0	0	0	0	1	0	1	1	1	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0		



- **浮点数使用方便**
- **运算非常复杂**
- **需要特殊的浮点运算单元，一般只有高端CPU中才会有，成本较高，而且智能处理加法、减法和乘法。**
- **其他数学计算，比如除法，开方，三角函数等，则计算更加复杂耗时。**
- **定点数与浮点数之间相互转换，也消耗计算时间。**



# • 作业

1. 冯·诺伊曼计算机的基本设计思想是什么？哈佛结构的计算机，与冯·诺伊曼计算机相比，有哪些优缺点？
2. 什么是总线，总线通常有哪3组信号？各组信号的作用是什么？
3. 1) 计算机的字长是什么含义？ 2) 简述处理器中的流水线技术。
4. 将下列十六进制无符号整数，转换为十进制真值。  
1) 0FFH      2) 0H      3) 5EH      4) EFH
5. 如果上题中的十六进制数为8位有符号整数，请将其转换位十进制真值
6. 将下列十进制数转换位压缩BCD码  
1) 12      2) 24      3) 68      4) 99
7. 将下列二级制补码表示的有符号整数转换为十进制真值  
1) 0000 0000b      2) 0111 1111b      3) 1000 0001b      4) 1100 0111b

