西北工业大学
NORTHWESTERN POLYTECHNICAL UNIVERSITY

C程序设计

Programming in C

1011014

主讲：姜学锋，计算机学院

信息在计算机中的表示

2、数值数据的表示与存储

3、字符数据的表示与存储

4、其他数据的表示与存储

1.2.3 数值数据的表示

- 1．整数在计算机中的表示
- 由于计算机只有0和1的数据形式，因此数的正（+）、负（–）号也要用0和1编码。通常将一个数的最高二进制位定义为符号位，称为数符，用0表示正数、1表示负数，其余位表示数值。

1.2.3 数值数据的表示

- 在计算机中，作为整体参与运算、处理和传送的一串二进制的位数称为字长，字长一般是8的倍数，例如8位、16位、32位、64位等。一个数在计算机中的表示形式称为机器数。假定字长为8位，5的机器数为00000101，–5的机器数为10000101。

1.2.3 数值数据的表示

- 下面介绍原码、反码和补码，为了简单起见，以下假定字长为8位。

1.2.3 数值数据的表示

- （1）原码
- 整数X的原码是数符位0表示正，1表示负，数值部分是X绝对值的二进制表示，记为(X)原。原码表示的计算公式为：

$$(X)_{原}=\begin{cases} X & 0\le X<2^{n-1} \\ 2^{n-1}+|X| & -2^{n-1}<X\le 0 \end{cases}$$

式1-8

- 其中n为字长，原码表示数的范围是 $-(2^{n-1}-1)\sim 2^{n-1}-1$

1.2.3 数值数据的表示

- 例如：

$$(+1)_{原}=00000001 \quad (+127)_{原}=01111111 \quad (+0)_{原}=00000000$$

$$(-1)_{原}=10000001 \quad (-127)_{原}=11111111 \quad (-0)_{原}=10000000$$

- 由此可知，8位原码表示的最大值为127，最小值为–127，表示数的范围是–127～127，其中0有两种表示形式。
- 原码表示法编码简单，但它的缺点是运算时要单独考虑符号位和判别0，增加了运算规则的复杂性。

1.2.3 数值数据的表示

- （2）反码
- 整数X的反码是对于正数，反码就是原码；对于负数，数符位为1，其数值位为原码中的数值位按位取反，记为(X)反。反码表示的计算公式为：

$$(X)_{反}=\begin{cases} X & 0\le X<2^{n-1} \\ 2^{n}-1-|X| & -2^{n-1}<X\le 0 \end{cases}$$

式1-9

- 其中n为字长，反码表示数的范围是 $-(2^{n-1}-1)\sim 2^{n-1}-1$ 。

1.2.3 数值数据的表示

- 例如：

$$(+1)_{反}=00000001 \quad (+127)_{反}=01111111 \quad (+0)_{反}=00000000$$

$$(-1)_{反}=11111110 \quad (-127)_{反}=10000000 \quad (-0)_{反}=11111111$$

- 由此可知，8位反码表示的最大值、最小值和数的范围与原码相同，其中0也有两种表示形式。
- 反码运算也不方便，很少使用，一般用来求补码。

1.2.3 数值数据的表示

- （3）补码
- 整数X的补码是：对于正数，补码与反码、原码相同；对于负数，数符位为1，其数值位为反码加1，记为(X)补 。补码表示的计算公式为：

$$(X)_{补}=\begin{cases} X & 0\le X<2^{n-1} \\ 2^{n}-|X| & -2^{n-1}<X\le 0 \end{cases}$$

式1-10

- 其中n为字长，补码表示数的范围是 $-2^{n-1}\sim 2^{n-1}-1$ 。

1.2.3 数值数据的表示

- 例如：

$$(+1)_{补}=00000001 \quad (+127)_{补}=01111111 \quad (+0)_{补}=(-0)_{补}=00000000$$

$$(-1)_{补}=11111111 \quad (-127)_{补}=10000001 \quad (-128)_{补}=10000000$$

- 由此可知，8位补码表示的最大值为127，最小值为–128，表示数的范围是–128～127，其中0有唯一的编码形式。

1.2.3 数值数据的表示

【例1.4】

计算（－9）+9的值。

解：

```
    11110111     ……  －9的补码
+   00001001     ……   9的补码
 ①00000000     ……  最高位进位丢弃
```

丢弃高位1，运算结果为0。

1.2.3 数值数据的表示

【例1.5】

计算（－9）+8的值。

解：

$$
\begin{array}{rll}
 & 11110111 & \cdots\cdots \text{ －9的补码} \\
+ & 00001000 & \cdots\cdots \text{ 8的补码} \\
\hline
 & 11111111 & \cdots\cdots \text{ －1的补码}
\end{array}
$$

运算结果为–1。

1.2.3 数值数据的表示

【例1.6】

计算65+66的值。

解：

$$
\begin{array}{rll}
 & 01000001 & \cdots\cdots \ 65\text{的补码} \\
+ & 01000010 & \cdots\cdots \ 66\text{的补码} \\
\hline
 & 10000011 & \cdots\cdots \ -125\text{的补码}
\end{array}
$$

两个正数相加，从结果的符号位可知运算结果是一个负数（–125），其原因是结果（131）超出了数的有效表示范围（–128～127）。

1.2.3 数值数据的表示

【例1.7】

求补码10000000对应的十进制数。

解：从符号位判断该数为一个负数，根据式（1–10）可知：

$$(|X|)_{补}+(-|X|)_{补}=2^n$$

则：

$$(|X|)_{补}=2^n-(-|X|)_{补}=100000000\mathrm{B}-10000000\mathrm{B}=10000000\mathrm{B}=(128)_D$$

所以补码10000000对应的十进制数为–128。

- （4）无符号整数
- 无符号整数是指没有正负之分的整数。无符号整数总是大于等于0的，其数的表示范围是 $0 \sim 2^n - 1$ ，即二进制的每一位都是数值位。显然，在一定字长情况下，无符号整数的数值比有符号整数的数值大。

1.2.3 数值数据的表示

【例1.8】

计算无符号整数65+66的值。

解：从前面得到 $65+66=(10000011)_B$ ，由于是无符号整数，故直接转换成十进制数为131。

1.2.3 数值数据的表示

- 2．浮点数在计算机中的表示
- 数学中的实数在计算机中称为浮点数，是指小数点不固定的数。浮点数用二进制表示，但表示方法比整数复杂得多。

1.2.3 数值数据的表示

- 为便于软件的移植，目前大多数计算机都遵守1985年制定的**IEEE754浮点数标准**（最新标准为IEEE754–2008），主要有单精度浮点数（float或single）和双精度浮点数（double）格式。

- 按二进制数据形式，单精度格式具有24位有效数字，总共占用32位；双精度格式具有53位有效数字精度，总共占用64位，相对应的十进制有效数字分别为7位和17位。

1.2.3 数值数据的表示

- 下面以单精度浮点数为例，介绍浮点数在计算机中的表示。
- 按IEEE754的规定，浮点数使用下列形式的规格化表示：

$$规格化数 = (-1)^s \times 2^E \times 1.f$$

- 其中s为符号，E为指数，f为小数。

1.2.3 数值数据的表示

- 单精度浮点数存储时占用4个字节，即32位，各位的意义和布局如图所示。

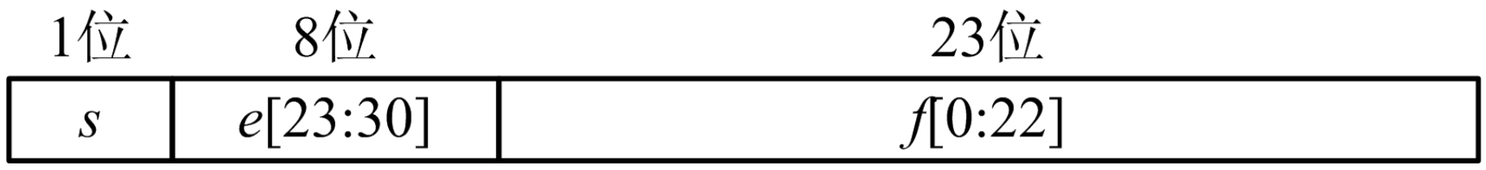
图1.5 单精度浮点数存储格式

1位	8位	23位
s	e[23:30]	f[0:22]

- （1）0:22位是23位小数f，其中第0位是小数的最低有效位，第22位是最高有效位。小数中的“1.”不用存储，目的是为了节省存储空间。23位小数加上隐含前导有效位提供了24位精度。

1.2.3 数值数据的表示

图1.5 单精度浮点数存储格式



- （2）23:30位是8位e，其中第23位是e的最低有效位，第30位是最高有效位，$0<e<255$ 。指数 $E=e-127$ ，其范围为 $-126\sim127$ 。
- （3）最高的第31位是符号位s，0表示正，1表示负。

1.2.3 数值数据的表示

表1-3 单精度存储格式位模式及其IEEE值

通用名称	位模式（十六进制）	十进制值
+0	00000000	0.0
–0	80000000	–0.0
1	3F800000	1.0
最大正数	7F7FFFFF	$3.40282347 \times 10^{+38}$
最小正数	00800000	$1.17549435 \times 10^{-38}$
+∞	7F800000	正无穷
–∞	FF800000	负无穷
非数	7FC00000	NAN

1.2.3 数值数据的表示

【例1.9】

求单精度浮点数50.0在计算机中的表示。

解：格式化表示：$50.0 = 110010.0\text{B} = (-1)^0 \times 2^5 \times 1.100100\text{B}$，因 $s = 0$ ，$E = 5$ ，

$f = 0.100100$

指数：$e = E + 127 = 132 = 10000100\text{B}$

所以50.0在计算机中的表示为42480000（十六进制），其存储格式如图所示。

0	10000100	10010000000000000000000

1.2.3 数值数据的表示

【例1.10】

求单精度浮点数-2.5在计算机中的表示。

解：格式化表示：$-2.5=-10.1\text{B}=(-1)^1\times2^1\times1.01\text{B}$ ，因此 $s=1$，$E=1$ ，$f=0.01$

指数：$e=E+127=128=10000000\text{B}$

所以 在计算机中的表示为C0200000（十六进制），其存储格式如图所示。

1	10000000	01000000000000000000000

1.2.3 数值数据的表示

- 双精度浮点数在计算机中的表示与单精度浮点数类似，只有两点区别：一是双精度浮点数存储时占用8个字节，即64位。其中 占1位， 占11位， 占52位；二是指数 。

1.2.4 非数值数据的表示

- 1．西文字符
- 西文字符包含英文字符、数字、各种符号，是不可做数学运算的数据。西文字符按特定的规则进行二进制编码才能进入计算机，最常用的是美国信息交换标准代码ASCII（american standard code for information interchange）。

1.2.4 非数值数据的表示

附录A　ASCII码对照表

DEC	HEX	字符	控制字符	DEC	HEX	字符	DEC	HEX	字符	DEC	HEX	字符	DEC	HEX	字符	DEC	HEX	字符	DEC	HEX	字符	DEC	HEX	字符
0	00	(null)	NUL	32	20	空格	64	40	@	96	60	`	128	80	Ç	160	A0	á	192	C0	└	224	E0	α
1	01	☺	SOH	33	21	!	65	41	A	97	61	a	129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
2	02	☻	STX	34	22	"	66	42	B	98	62	b	130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
3	03	♥	ETX	35	23	#	67	43	C	99	63	c	131	83	â	163	A3	ú	195	C3	├	227	E3	π
4	04	♦	EOT	36	24	$	68	44	D	100	64	d	132	84	ä	164	A4	ñ	196	C4	─	228	E4	Σ
5	05	♣	ENQ	37	25	%	69	45	E	101	65	e	133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
6	06	♠	ACK	38	26	&	70	46	F	102	66	f	134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
7	07	•	BEL	39	27	'	71	47	G	103	67	g	135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
8	08	◘	BS	40	28	(	72	48	H	104	68	h	136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
9	09	○	TAB	41	29	)	73	49	I	105	69	i	137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
10	0A	◙	LF	42	2A	*	74	4A	J	106	6A	j	138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
11	0B	♂	VT	43	2B	+	75	4B	K	107	6B	k	139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
12	0C	♀	FF	44	2C	,	76	4C	L	108	6C	l	140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
13	0D	♪	CR	45	2D	-	77	4D	M	109	6D	m	141	8D	ì	173	AD	¡	205	CD	═	237	ED	ø
14	0E	♫	SO	46	2E	.	78	4E	N	110	6E	n	142	8E	Ä	174	AE	«	206	CE	╬	238	EE	∈
15	0F	☼	SI	47	2F	/	79	4F	O	111	6F	o	143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
16	10	►	DLE	48	30	0	80	50	P	112	70	p	144	90	É	176	B0	░	208	D0	╨	240	F0	≡
17	11	◄	DC1	49	31	1	81	51	Q	113	71	q	145	91	æ	177	B1	▒	209	D1	╤	241	F1	±
18	12	↕	DC2	50	32	2	82	52	R	114	72	r	146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
19	13	‼	DC3	51	33	3	83	53	S	115	73	s	147	93	ô	179	B3	│	211	D3	╙	243	F3	≤
20	14	¶	DC4	52	34	4	84	54	T	116	74	t	148	94	ö	180	B4	┤	212	D4	╘	244	F4	⌠
21	15	§	NAK	53	35	5	85	55	U	117	75	u	149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
22	16	▬	SYN	54	36	6	86	56	V	118	76	v	150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
23	17	↨	ETB	55	37	7	87	57	W	119	77	w	151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
24	18	↑	CAN	56	38	8	88	58	X	120	78	x	152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	°
25	19	↓	EM	57	39	9	89	59	Y	121	79	y	153	99	Ö	185	B9	╣	217	D9	┘	249	F9	•
26	1A	→	SUB	58	3A	:	90	5A	Z	122	7A	z	154	9A	Ü	186	BA	║	218	DA	┌	250	FA	•
27	1B	←	ESC	59	3B	;	91	5B	[	123	7B	{	155	9B	¢	187	BB	╗	219	DB	█	251	FB	√
28	1C	∟	FS	60	3C	<	92	5C	\	124	7C	|	156	9C	£	188	BC	╝	220	DC	▄	252	FC	^{n}
29	1D	↔	GS	61	3D	=	93	5D	]	125	7D	}	157	9D	¥	189	BD	╜	221	DD	▌	253	FD	^{2}
30	1E	▲	RS	62	3E	>	94	5E	^	126	7E	~	158	9E	₧	190	BE	╛	222	DE	▐	254	FE	■
31	1F	▼	US	63	3F	?	95	5F	_	127	7F	⌂	159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	

DEC：十进制ASCII值；HEX：十六进制ASCII值；128～255为扩展ASCII码。

1.2.4 非数值数据的表示

- 计算机存储与处理一般以字节为单位，因此西文字符的一个字符在计算机内部实际是用8位表示的。

1.2.4 非数值数据的表示

- 2．汉字字符
- 汉字字符种类多，编码上比西文字符复杂。在汉字处理系统中，需要在输入、内部处理、输出对汉字字符编码及转换。因此汉字字符编码有输入码、字形码、国标码、机内码之分。输入码是键盘输入汉字时所用的编码，字形码用于汉字的显示和打印输出。

1.2.4 非数值数据的表示

- 汉字国标码是指我国在1980年发布的《中华人民共和国国家标准信息交换汉字编码》GB2312-80，它把最常用的6763个汉字和682个非汉字图形符号按汉语拼音顺序和偏旁部首排列。

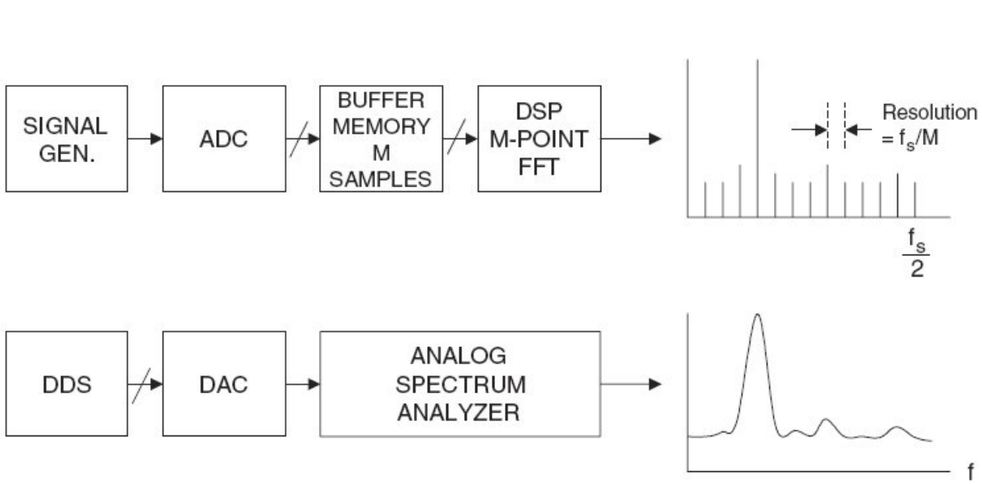
1.2.4 非数值数据的表示

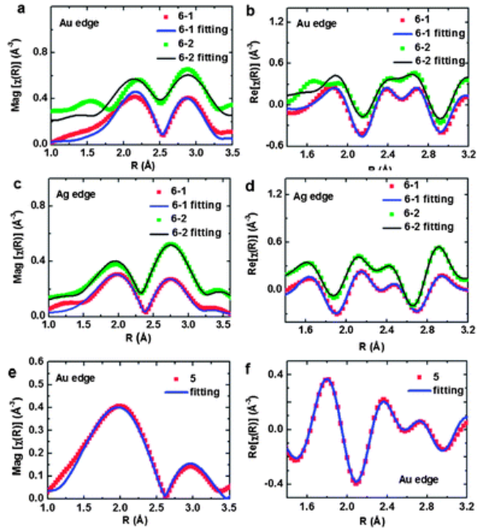
- 汉字国标码与区位码的关系
- 为了在计算机内部方便区分汉字编码和ASCII码，将国标码的每个字节的最高位设置成1，变换后的国标码称为汉字机内码，即：

  汉字机内码＝汉字国标码+8080H=区位码+A0A0H
- 这样在文字处理系统中，字节值大于128的字符是汉字机内码，字节值小于128的字符是ASCII码。

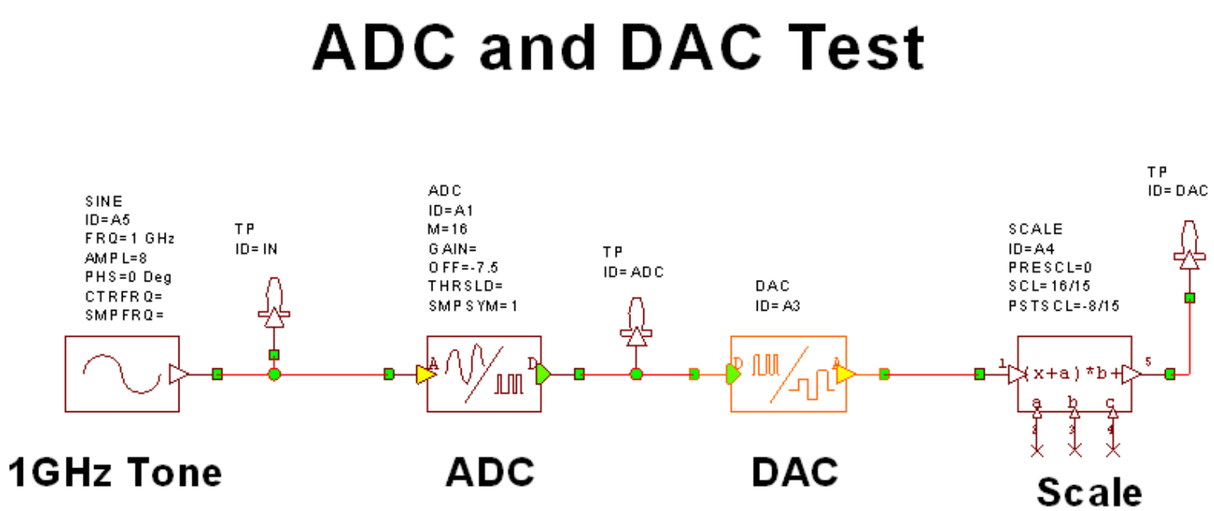
1.2.4 非数值数据的表示

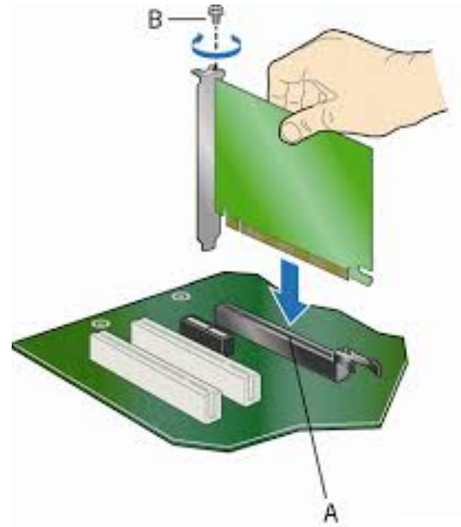
- 3．多媒体信息
- 除数值、文字数据外，计算机也可以处理图形、图像、音频和视频信息。这些媒体信息的表现方式可以说是多种多样，但是在计算机中它们都是通过二进制编码表示的。

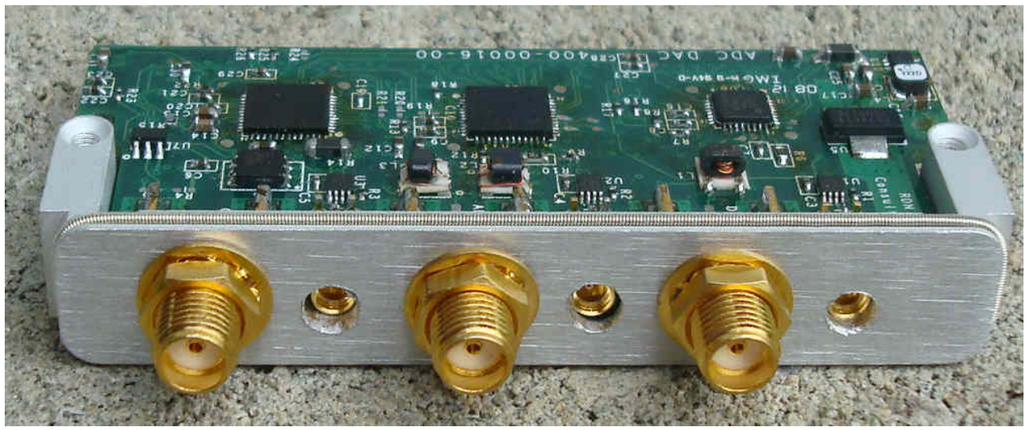




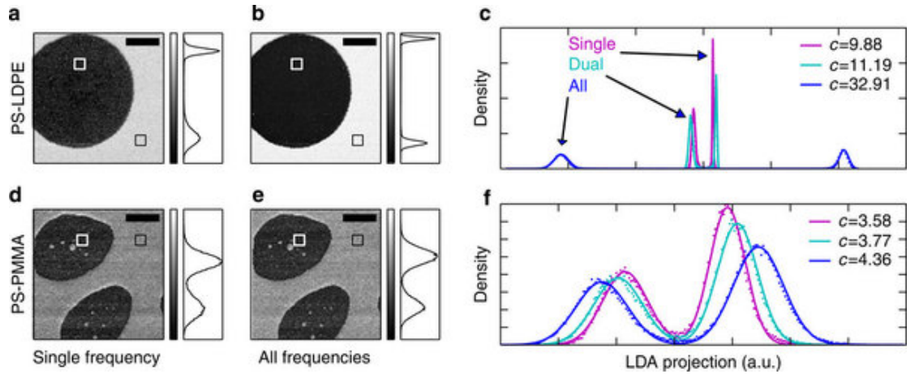
1.2.4 非数值数据的表示

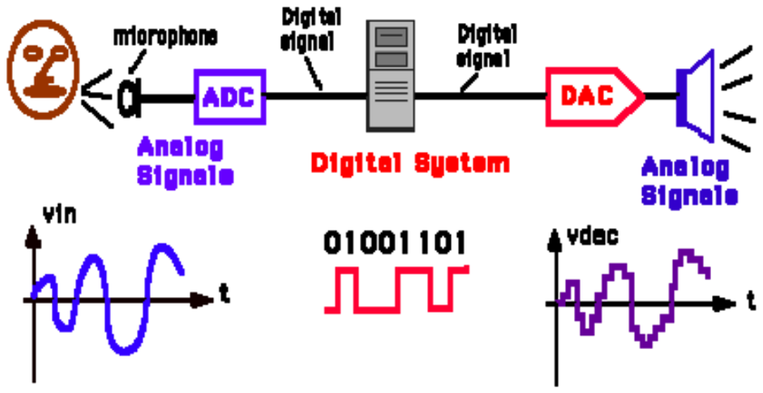






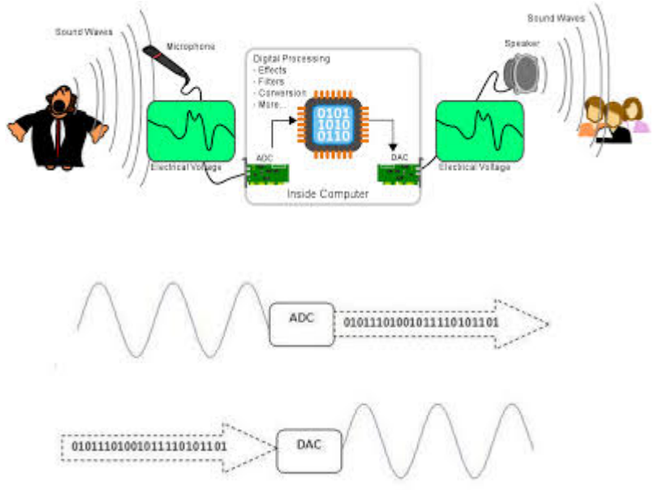
1.2.4 非数值数据的表示

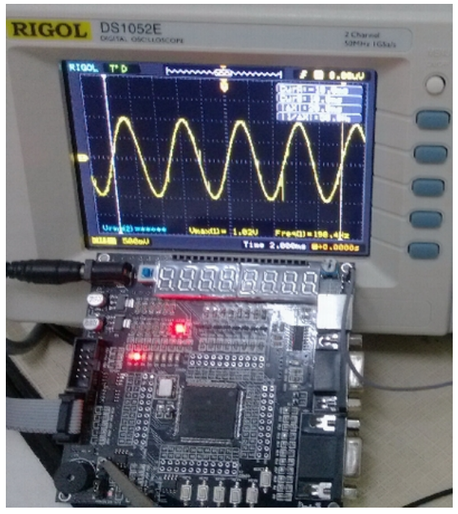


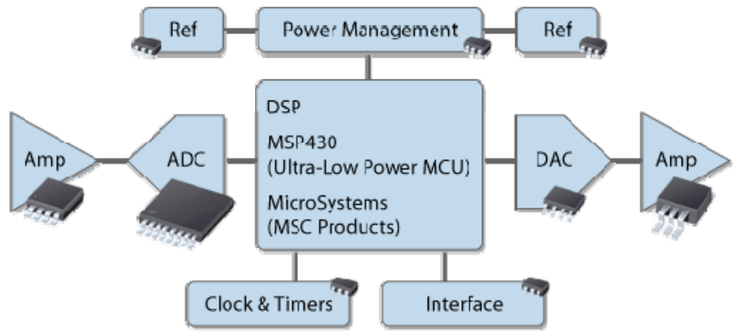


1.2.4 非数值数据的表示

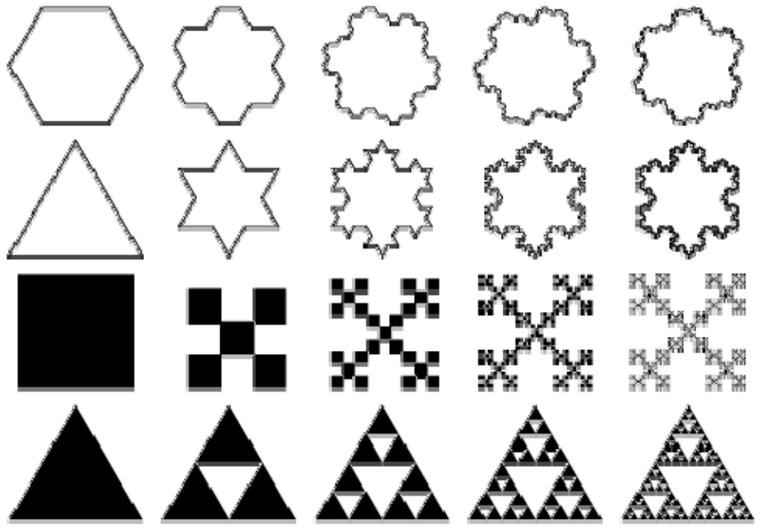
- 数字音频是由A/D（模拟/数字）转换器用一定采样频率采样、量化音频信号，然后使用固定二进制位记录量化值以数字声波文件的形式存储在计算机中。若要输出数字声音，必须通过D/A（数字/模拟）转换器将数字信号转换成模拟信号输出。

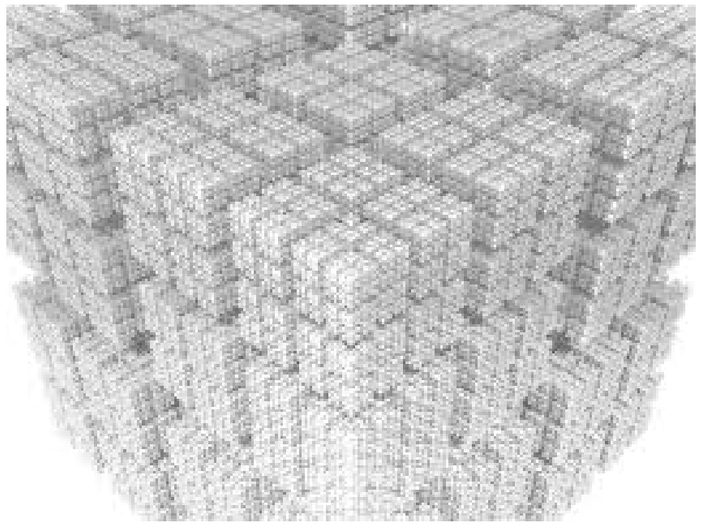


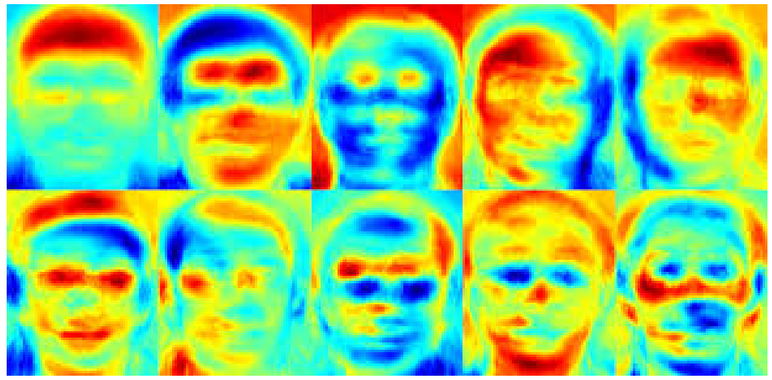




1.2.4 非数值数据的表示








程序设计