机电系统控制软件设计

孙志辉　闫晓强　程伟　编著

机械工业出版社
CHINA MACHINE PRESS

机电系统控制软件设计

孙志辉　闫晓强　程　伟　编著

机 械 工 业 出 版 社

本书主要介绍机电控制系统中计算机控制的软硬件开发过程和相关技术。全书内容包括：各种编码、进制、数据存储格式及其相互转换的程序实现，各类接口技术及其程序实现，机电系统的各类控制对象及其控制方法，机电系统的各类输入、输出控制信号及实现，机电系统控制软件的开发工具和控制算法。

本书从实际应用出发，结合大量程序实例，采用基本概念与程序示例相结合的形式编写，给读者提供大量的经验和技巧。

本书可作为高校机电专业本科生和研究生的相关课程教材或课题研究参考书，也可作为从事机电设备控制系统开发工程技术人员的参考书。

图书在版编目（CIP）数据

机电系统控制软件设计/孙志辉，闫晓强，程伟编著．北京：机械工业出版社，2009.1

ISBN 978-7-111-25427-0

Ⅰ.机… Ⅱ.①孙…②闫…③程… Ⅲ.机电系统-计算机控制-软件设计 Ⅳ.TH-39 TP273

中国版本图书馆 CIP 数据核字（2008）第 165734 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）

责任编辑：靳 平 版式设计：张世琴 责任校对：樊钟英

封面设计：赵颖哲 责任印制：杨 曦

北京瑞德印刷有限公司印刷（三河市胜利装订厂装订）

2009 年 1 月第 1 版第 1 次印刷

169mm×239mm · 14.75 印张 · 287 千字

0001-4000 册

标准书号：ISBN 978 7-111-25427-0

定价：25.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

销售服务热线电话：（010）68326294

购书热线电话：（010）88379639 88379641 88379643

编辑热线电话：（010）88379767

封面无防伪标均为盗版

前　言

机电设备的数字控制技术越来越普及，而数字控制技术涉及到机、电、液、计算机等多方面技术，特别是计算机技术。无论是机电专业还是自动化专业的学生，往往局限于某一方面的知识，有关数字控制的概念可能比较清楚，但如何用计算机程序实现这些概念和功能却无从下手。许多高校都设有机械电子或相关专业，学生们对从事计算机控制方面的研究都具有很大的兴趣，但真正动起手来很困难，通常要花费大量的时间从最基本的概念开始学习，希望本书能够帮助他们较快入门并掌握运用计算机实现机电系统控制的技术。

目前，市场上有关计算机编程的书籍基本都是介绍具体某种语言的语法及其编程方法，有关硬件操作与控制程序设计的内容却很少，而本书内容恰好填补这方面空白，从实际应用出发，结合大量的程序实例，给从事计算机控制研究的工程技术人员提供大量的经验和技巧，避免他们走弯路。

本书采用基本概念与程序示例相结合的形式编写，涉及与计算机控制有关的编码、进制、接口、控制对象、信号获取与产生、编程基本方法等方面，讲解计算机控制系统中软硬件的开发过程和相关技术。

本书由北京科技大学孙志辉统稿，3.8 节由闫晓强编写，其余内容由孙志辉编写，程伟整理并提供了本书的部分参考资料。

本书是一本理论与实际相结合，并以实际应用为主的参考书籍，可以作为高校机电专业本科生和研究生的相关课程的教材或课题研究参考书，也可作为从事机电设备开发的工程技术人员的参考书。

由于编者水平有限，书中难免会存在一些错误，殷切希望广大读者提出批评指正意见，联系邮箱为 hyj57@yahoo.cn

编著者

目　录

第1章　计算机基础知识

1.1　进制与编码

1.1.1　计算机中的进制

自从1946年世界上第一台电子计算机ENIAC诞生以来，其主要目的就是为了进行数值计算，计算机中的各种数据信息采用数字电路的开关状态表示，而这种开关状态正对应于二进制的表示形式，所以计算机内部的数值均采用二进制。1946年，John Von Neumann在计算机体系结构中明确提出采用二进制。

二进制的格式与计算机内部信息格式一致，但在采用高级语言进行程序设计时，二进制格式存在两个问题：

1）不符合人们的日常使用习惯。

2）书写比较麻烦，容易出错，也不便于记忆。

因此，在高级语言程序设计中，除了二进制以外，还采用其他进制形式表示数值，如十进制、八进制、十六进制，在编译成机器语言时，由编译系统自动将它们转换成计算机能够识别的二进制格式。

1.1.2　进位计数法

二进制、八进制、十进制、十六进制都属于进位计数法，即某位逢几进1，如二进制逢2进1，十进制逢10进1等。

进位计数法中，有两个重要的概念：进位基数与权。

所谓进位基数，是指某种进制中表示一位所需要的符号数量。二进制需要0和1共两个符号；八进制需要0、1、2…7共8个符号；十进制需要0、1、2…9共10个符号；而十六进制需要16个符号，除了0～9这10个符号外，还采用A～F这6个符号（或a～f），分别表示10～15。

权是进制中某一位所代表的数值大小，如十进制中的个位代表1（10^0），十位代表10（10^1），小数点后各位分别代表10^{-1}、10^{-2}…，以此类推。二进制小数点左侧从右向左分别代表2^0、2^1…，小数点右侧从左向右分别代表2^{-1}、2^{-2}…；十六进制小数点左侧从右向左分别代表16^0、16^1…，小数点右侧从左向右分别代表16^{-1}、16^{-2}…。

对于各种进制，了解各位的权是把它们转换成十进制的关键。对于二进制，一般要求能够记住 1、2、4、8、16、32、64、128、256、512、1024…32768、65536 这些权，1024 往往又称之为 1K。由于计算机是以 8 个二进制位为基本存储单位，数往往是按 8 位、16 位、32 位、64 位二进制组织，256 和 65536 正是二进制中对应 8 位和 16 位的权（也是 8 位二进制和 16 位二进制表示数的数量），在计算机处理信息时使用得很频繁。十六进制中，要求记住 256 和 65536，对应 2 位和 4 位的权。

高级语言中采用十进制是为了方便人们的日常使用习惯，而采用八进制和十六进制则是为了书写方便，大多数计算机编程语言都支持这些进制。主要计算机语言中各进制的表示见表 1-1。

表 1-1 主要计算机语言中各进制的表示

语言	二进制	十进制	八进制	十六进制
汇编语言	10110101B	10	12Q	1AFH
C/C++语言	不支持	10	012	0x1AF
Basic/VB 语言	不支持	10	&O12	&H1AF

进行计算机编程时，如果仅仅进行数值计算，则一般采用十进制即可，而编写计算机控制程序时，往往要使用八进制和十六进制，特别是十六进制，这是由于八进制和十六进制与二进制之间存在 3 次方和 4 次方的关系，1 位八进制或十六进制可以表示 3 位或 4 位二进制。

1.1.3 BCD 编码

对于二进制表示的数的格式，在存储、传输和显示时很不方便，如数 123 在计算机内部表示成二进制 01111011，如果需要在显示器屏幕上显示 1、2、3 这 3 个字符，从 01111011 这 8 个 0 和 1 的组合很难判断显示的顺序和内容。为此，计算机内部在保存数时引入一种特殊的十进制格式，即所谓的二进制编码的十进制数（Binary Coded Decimal，BCD）。

用 BCD 编码时，对于 123 这样的十进制数，每 1 位十进制数均采用 4 位或 8 位二进制编码表示，如 123，可表示为 0001 0010 0011，从 0001 可以知道需要显示 1 这个符号，从 0010 可以知道需要显示 2 这个符号，从 0011 可以知道需要显示 3 这个符号，这样就可以依次在屏幕上显示 1、2、3 这 3 个符号。

采用 4 位二进制对 0～9 这 10 个数进行 BCD 编码时，两位十进制数组成 8 位，正好可以存放在计算机内部的一个 8 位存储单元中，这种 BCD 编码方式又称为组合 BCD 编码或压缩 BCD 编码；而采用 8 位进行 BCD 编码时，一位十进制的编码存放在计算机内部的一个存储单元中，称为非组合 BCD 编码或非压缩

BCD 编码。使用较多的是 4 位的压缩 BCD 码（可以节省存储空间）。

4 位二进制可以对 16 个十进制数编码，对其中 0～9 这十个数编码可以有许多种不同的编码方法。比较常用的压缩 BCD 编码见表 1-2。

表 1-2　主要的压缩 BCD 编码方式

	0	1	2	3	4	5	6	7	8	9
8421BCD 码	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001
格雷 BCD 码	0000	0001	0011	0010	0110	1110	1010	1000	1100	0100
余 3BCD 码	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100
2421BCD 码	0000	0001	0010	0011	0100	0101	0110	0111	1110	1111
5211BCD 码	0000	0001	0011	0101	0111	1000	1001	1011	1101	1111

1. 8421BCD 码

这种 BCD 编码采用二进制的权进行编码，即编码中从左到右 4 位二进制的权分别为 8、4、2、1，其 0～9 的编码见表 1-2。

8421BCD 码的 4 位信息与二进制完全相同，便于进行 BCD 与二进制之间的转换，是实际应用最广泛的一种 BCD 编码。

在机电系统的计算机控制中，经常需要采用 LED（7 段数码二极管）显示检测或计算的数据，送到 LED 的数据往往以 8421BCD 编码的形式送出去，通过芯片 74LS47 将 8421BCD 编码转换成 7 段码送 LED 显示。

2. 格雷 BCD 码

格雷（Gray）BCD 码的特点是相邻两个数的编码只有 1 位不同。这样的编码方式主要用在绝对编码器中（绝对编码器是用于检测电机转动角度的一种传感器，将在第 3 章介绍），可以提高编码器的读数精度。

3. 余 3BCD 码

余 3BCD 码是在 8421BCD 码的基础上加 3 产生的，其特点是两个余 3BCD 码编码的十进制数之和是 8421BCD 码，而 8421BCD 码的十进制数之和需要处理以后才是正确的 8421BCD 码。

例如 8 和 9 求和，采用 8421BCD 码，8 和 9 分别为 1000 和 1001，相加后为 0001 0001，是 11 的 8421BCD 码，需要在低 4 位加上 6，才能变成 17 的 8421BCD 码；而采用余 3 码，8 和 9 分别为 1011 和 1100，相加后为 0001 0111，正好是 17 的 8421BCD 码。

对于机电控制系统，8421BCD 码的使用最多，许多数字化位移传感器或二次仪表的输出往往采用 8421BCD 码，如轧钢机的轧辊辊缝检测传感器等。这些输出可以送给 PC 或 PLC，由它们的开关量或数字量输入模块输入，然后转换成十进制数值进行进一步处理。这种转换没有标准的函数能实现，只能通过自己编

程解决。

1.1.4 ASCII 编码

前面介绍的进制和编码都是针对数的，而计算机的功能除了数值计算外，另一个重要功能就是进行文字信息处理。对文字（或称字符）的处理最早是从对英文字母和常用符号编码开始的，可以解决基本文字处理、高级语言编程、信息的输入/输出等，见表 1-3。随着计算机的发展和普及，对各国语言的支持变得越来越重要，随之产生了其他各种编码方式。

表 1-3 ASCII 字符表

	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYM	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	)	8	H	X	h	x
1001	HT	EM		9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

对字符最早的编码是 ASCII，是 1963 年由美国定义的常用字符编码方式，其全称为 American Standard Code for Information Interchange（美国信息交换标准码）。它采用 7 位二进制对 128 个字符进行了编码，可以用于字符的传输、存储和比较，主要包括：

1）26 个小写英文字母，a(97)，b，c…。

2）26 个大写英文字母，A(65)，B，C…。

3）10 个数字码，0(48)，1，2…。

4）各类数学符号，+(43)，*(42)，/(47)，<(60)，!(33)，"(34)，

%(37) …。

5）显示打印通信控制命令，BS(8)，LF(10)，CR(13) …。

在采用 ASCII 的系统中，所有字符的保存、传输均采用 ASCII，如把字符 A 保存到硬盘中，实际上保存的是 65 这个数。即使是后来提出的其他字符编码，为了保持兼容，对表中涉及的字符仍然采用 ASCII 编码。

ASCII 表前 32 个字符主要用于屏幕显示、打印和串行通信的控制命令，表中的字符串表示其含义，而不是表示几个字符。

ASCII 编码 27（0101011）的字符，表中用 ESC 表示，这个字符是过去针式打印机的控制命令起始符，当希望打印机打印标准的 16 点阵汉字时，先送给打印机控制命令序列 ESC I A，用 C 语言表示如下：

```
fputc(27,stdprn);fputc(73,stdprn);fputc(65,stdprn);
```

按下键盘左上角的 Esc 键，送给计算机的实际上就是 27 这个 ASCII 编码的值。

ASCII 编码为 10 和 13 的两个字符分别为换行和回车控制字符，换行表示光标移动到下一行，回车表示光标移动到第一列，当把这两个字符送给屏幕显示或打印机打印时，都会把当前光标移动到下一行开始。键盘的回车键产生这两个字符。

当用文本格式把三行字符保存到文件中时，实际保存内容如下所示：

```
30 31 32 33 34 35 36 37 38 39 0D 0A 41 42 43 44 ;0123456789..ABCD
45 46 47 48 49 4A 0D 0A 61 62 63 64 65 66 67 68;EFGHIJ..abcdefgh
69 6A                                           ;ij
```

右侧为三行字符，左侧为其保存的 ASCII 码值（十六进制），每行之间插入了 0A 和 0D 两个数，表示回车换行。某些仪表或控制器传输数据时往往采用回车换行作为数据帧的结束标志。

如果把这些控制字符送到屏幕显示，DOS 操作系统会自动将其转换成对应的 CRT 显示输出码，如 ASCII 编码 01 显示成笑脸。

ASCII 的编码采用 7 位，而计算机系统存储信息时按 8 位（1 个字节）保存，所以 ASCII 的保存也采用 8 位。C 语言中的 char 类型变量可以用于保存 ASCII 字符，对于这样的变量，可以给它赋值字符，也可以给它赋值 ASCII 码值，例如：

```
char c; c='A'; c=65;
```

由于 ASCII 的应用，字符可以比较大小，实际就比较其 ASCII 码值，如 a 比 A 大。英文字母的大小写转换实际上也就是 ASCII 码值加减 32。大写转小写，ASCII 码值加 32；小写转大写，ASCII 码值减 32。

```
char c; c='A'; c=c+32; //大写转小写
```

字符 0～9 的 ASCII 实际上是 BCD 码的一种，由于采用 8 位，属于非压缩 BCD 码，其与 8421 码的转换就在于高 4 位是否变成 0011。

许多系统中采用 ANSI 编码，实际上是与 ASCII 编码完全兼容的。如 Windows 系统中的记事本程序保存文件时，可以选择保存的字符编码格式，其中的 ANSI 编码就是以 ASCII 编码保存字符。

由于 ASCII 编码保存时也采用 8 位，可以把所有 8 位都对字符进行编码，就形成了扩展 ASCII 编码，这样共可以对 256 个字符进行编码，从 128～255 这 128 个编码主要用于制表符号和特殊符号，如 171 表示 1/4，191 表示┐，227 表示 π。由于这些字符应用较少，最初处理汉字时就采用了这部分编码。

ASCII 编码加上扩展 ASCII 编码最多编码 256 个字符，只能解决英文输入。随着世界各个国家计算机的普及，对本国语言的支持（显示、打印）变得非常重要，为了解决这个问题，各个国家采取了不同的措施。对于中日韩采用象形文字的国家，涉及的字符比较多，8 位编码远远不能满足需要，只能采用 16 位进行编码。这些国家和地区各自对本国语言制订字符集标准，相互之间不同，比如对汉字的编码，中国大陆采用 GB 编码，而中国台湾地区采用大五码（Big 5）。这就给信息交换带来了问题，比如中国大陆的用户登录中国台湾地区的网页，看到的就是乱码。

1.1.5 汉字编码

1. GB2312 编码

我们国家为了适应计算机的迅速普及，解决汉字的显示、输入、打印问题，于 1981 年公布了一套汉字编码标准，称为《信息交换用汉字编码字符集基本集》，即 GB 2312—1980，把高频字、常用字归结为汉字基本字符集（共 6763 个汉字），其中一级汉字 3755 个（按拼音排序），二级汉字 3008 个（按部首排序），再加上西文字符（希腊文、俄罗斯文、日文等）、数字、图形符号等 700 多字符。整个字符集编码分成 94 个区，每个区 94 位，如第一个汉字“啊”处于 16 区 1 位。区号采用 1 个字节编码，位号采用 1 个字节编码，每个字符共采用两个字节编码（16 位），对于该字符集中的每个汉字和符号均有唯一的编码，称为区位码。

由于 1～94 的数用于 ASCII，将汉字实际保存时，为了便于区分汉字和 ASCII 字符，对汉字的编码做了适当的处理，形成了汉字的机内码。机内码是在区位码基础上产生的，它对区和位信息采用扩展 ASCII 范围内的数进行编码，当汉字系统显示字符时，连续两个扩展 ASCII 就认为是一个汉字。

由区位码转换成机内码，是在区码位码基础上各加 160（A0H），如汉字“啊”，区码和位码分别为 16、01，其机内码则为 176（B0H）、161（A1H），显

示为：

B0　A1　B1　B1　BE　A9　BF　C6　BC　BC　B4　F3　D1　A7　；啊北京科技大学

机内码解决了汉字的存储和传输问题，而汉字的显示和打印则是通过绘图的方法实现的，具体方法是按照汉字的形状和轮廓在屏幕或打印纸上绘制点或线。为此，首先要确定每个汉字的形状和轮廓，处理的方法有两种：点阵和矢量。

(1) 点阵汉字

最早的汉字系统采用点阵方法显示汉字，常用的是 16×16 的点阵。处理方法是在一个 16×16 的网格中书写汉字，使得其每个笔划都填入网格的某个格子中。图 1-1 即为汉字“北”的点阵。

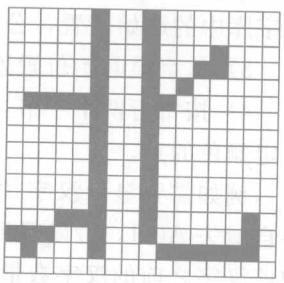

图 1-1　汉字“北”的点阵

对于图 1-1 的汉字“北”，其网格中填充了汉字笔划的地方用 1 表示，没有填充的地方用 0 表示，可以表示成如下二进制形式：

```
0 0 0 0 0 1 0 0   1 0 0 0 0 0 0 0      04H 80H
0 0 0 0 0 1 0 0   1 0 0 0 0 0 0 0      04H 80H
0 0 0 0 0 1 0 0   1 0 0 0 1 0 0 0      04H 88H
0 0 0 0 0 1 0 0   1 0 0 1 1 0 0 0      04H 98H
0 0 0 0 0 1 0 0   1 0 1 0 0 0 0 0      04H A0H
0 1 1 1 1 1 0 0   1 1 0 0 0 0 0 0      7CH C0H
0 0 0 0 0 1 0 0   1 0 0 0 0 0 0 0      04H 80H
0 0 0 0 0 1 0 0   1 0 0 0 0 0 0 0      04H 80H
0 0 0 0 0 1 0 0   1 0 0 0 0 0 0 0      04H 80H
0 0 0 0 0 1 0 0   1 0 0 0 0 0 0 0      04H 80H
0 0 0 0 0 1 0 0   1 0 0 0 0 0 0 0      04H 80H
0 0 0 0 0 1 0 0   1 0 0 0 0 0 0 0      04H 80H
0 0 0 1 1 1 0 0   1 0 0 0 0 0 1 0      1CH 82H
1 1 1 0 0 1 0 0   1 0 0 0 0 0 1 0      E4H 82H
0 1 0 0 0 1 0 0   0 1 1 1 1 1 1 0      44H 7EH
0 0 0 0 0 0 0 0   0 0 0 0 0 0 0 0      00H 00H
```

将网格每行的 16 位二进制按从左自右的顺序组合成两个字节，这样 16×16 的网格共可以组合成 32 个字节。这 32 个字节表征了一个汉字的形状，称为字模，把字符集中所有汉字的字模按照其区位码顺序保存在一个文件中，就形成了汉字库，每个字模在汉字库中存放时从左上角开始，先行后列，对上面“北”的字模，存放顺序依次为 04 80 04 80 04 88 04 98 04 A0 7C C0 04 80 04 80 04 80 04

80 04 80 04 80 1C 82 E4 82 44 7E 00 00。

需要显示或打印16点阵汉字时，采取如下步骤：

1）将汉字机内码的两个字节各自减去160（A0H），得到其区位码。

2）由区位码计算汉字库中该汉字之前的字符数，并由此确定该汉字字模在汉字库中的起始位置，公式为（（区码－1）×94＋（位码－1））×32。

3）从字模起始位置连续读取32个字节的字模数据。

4）按每行两个字节，根据字模每个字节的各位信息在屏幕或打印纸上绘制指定颜色的点。

例如需要显示汉字“北”，其机内码的两个字节分别为B1H和B1H，它们各自减去A0H，可以得到其区码和位码分别为17和17；由此可以计算出汉字库中该汉字之前的汉字数量为（17－1）×94＋（17－1）＝1520，则“北”的字模在汉字库中的起始位置为1520×32＝48640；打开汉字库文件，从该位置读取32个字节，在屏幕指定坐标位置按点阵的形式绘制点即可。

下面代码即为用C语言实现读取汉字字模的功能：

```
unsigned char buffer[32];
void get_hanzi_data(char * hanzi)
{
    int fl;
    long qu,wei,seekip;
    if((fl=open("c:\\ucdos\\hzk16",O_RDONLY |O_BINARY))==-1){
      printf("cannot open hzk16\n");
      exit(1);
    }
    qu=hanzi[0]-0xa0; /* 取汉字的区码 */
    wei=hanzi[1]-0xa0; /* 取汉字的位码 */
    seekip=((qu-1) * 94+wei-1) * 32;
    lseek(fl,seekip,SEEK_SET);
    _read(fl,buffer,32); /* 取汉字点阵数据,送入缓冲区 buffer
}
```

下面代码采用VB语言实现汉字的显示功能：

```
hanziDBCS=StrConv(hanziStr, vbFromUnicode)'  编码转换
Qu=AscB(LeftB(hanziDBCS, 1)) -160
Wei=AscB(MidB(hanziDBCS, 2, 1))-160
Ip=(Qu-1) * 94+Wei-1
Open "hzk16" For Random As #1 Len=32
Get #1, Ip,hanziData  '读取汉字的字模保存到变量 hanziData
```

```
    Close #1
    Picture1. Cls  '汉字绘制在图片框 Picture1 上
    Picture1. DrawWidth=1
    For i=1 To 14  '在图片框 Picture1 上绘制点阵网格
      Picture1. Line (i, 0)-(i, 15)
      Picture1. Line (0, i)-(15, i)
    Next i
    Picture1. DrawWidth=15  '采用 15 宽的大点绘制汉字
    For i=0 To 15
      For j=0 To 1
        For k=0 To 7
          If (hanziData(i * 2+j) And 2^(7-k))<>0 Then
            Picture1. PSet (j * 8+k,i), RGB(255, 0, 0)
          End If
        Next k
      Next j
    Next i
    Picture1. DrawWidth=1
```

其显示效果如图 1-2 所示。

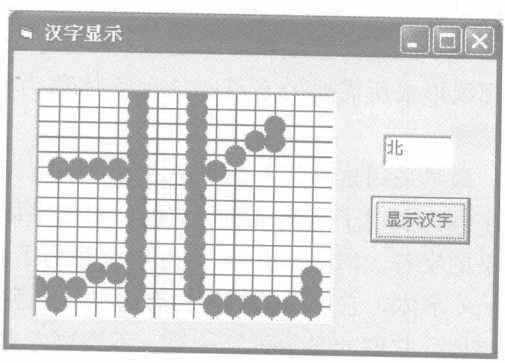

图 1-2　采用 VB 编程显示汉字

由于汉字的显示字体（字形）有许多种，如“宋体”、“黑体”、“仿宋体”等，其字模不同，由此需要对应各种字体的汉字库。

点阵汉字基本解决了汉字显示问题，但存在一个较大的缺点，即不能实现缩放，对于 16 点阵的汉字，希望把它显示成 32×32 的效果，则会出现马赛克效果，所有汉字处理系统都需要提供不同字体大小的汉字库，如 16 点阵汉字库、24 点阵汉字库、32 点阵汉字库、48 点阵汉字库等。

需要注意的是，每个 16 点阵汉字的字模为 32 个字节，每个 24 点阵汉字字模为 72 个字节。处理不同字体大小时，计算字模位置和读取字模应有所区别。

汉字的输入可以采用区位码法，或者与区位码一一对应的其他输入法。

国内最早采用 GB 码实现汉字处理的系统是原电子部六所开发的 CCDOS 汉字操作系统，它解决了汉字的存储、显示、打印与输入问题，为计算机在我国的

普及做出了巨大贡献。其他有代表性的汉字操作系统包括：希望公司的 UCDOS 系统、金山公司的金山 DOS 等。

随着中文 Windows 操作系统的应用，汉字的显示处理与英文字符的处理统一起来，再也不需要进行如上面所述的复杂步骤。但在一些特殊场合，依然要采用上面的方法处理汉字，最典型的应用是单片机系统的汉字显示。如银行的排队系统、各种 LED 信息显示等，都需要直接对汉字库进行操作，这些系统中汉字的滚动显示效果，实际上是按一定时间间隔，改变绘制汉字的起始坐标位置重新绘制汉字而形成的。

(2) 矢量汉字

为了克服点阵汉字不能缩放的缺点，引入了矢量汉字的方法。把一个汉字看作是由多个带有方向和长度的矢量组成，把这些矢量信息保存起来形成矢量字库。显示矢量汉字时，按照保存的汉字矢量信息，绘制直线即可。只要同时改变所有矢量的长度比例，就可以实现汉字的任意缩放。

(3) 轮廓汉字

矢量字体虽然可以实现任意缩放，但对于黑体不太合适，形成一定宽度的笔划需要许多矢量，而且，矢量字体采用直线描述字体特征，涉及到曲线时效果不太理想。

为了克服矢量字体的缺陷，采用数学公式描述字符的外部轮廓线，然后填充轮廓线形成所需要的字符。这种字体称为轮廓字体。一般，采用 Bezier 函数描述轮廓。

最早采用这种技术的是 Adobe 公司，它采用这种方法显示了称为 PostScript 的字体，实现了高质量的所见即所得的字体，可以在显示器和打印机上显示同样效果的字符。Microsoft 与 Apple 公司为了打破 Adobe 的垄断地位，开发了 TrueType 字体，它可以始终保持合适的比例和任意弯曲度。打印机和屏幕使用同样的字体，它们的显示机制一致，不需要考虑文档类型、打印机配置等，屏幕字体甚至可以旋转。

现在的 Windows 操作系统中普遍采用 TrueType 字体。Windows 文件夹下面的 Fonts 文件夹中保存系统安装的字体，其中以 ttf 和 ttc 为后缀的字体文件都是 TrueType 字体。图 1-3 即为某计算机安装的字体文件。

2. GBK 编码与 GB18030 编码

1995 年，为了适应新的 Windows 操作系统，我国在 GB 2312—1980 的基础上对汉字字符集进行了扩充，共包括了 21003 个汉字。所有字符均采用 2 字节 (16 位) 编码，并兼容原有 GB 2312—1980 字符集的机内码编码，称为 GBK 编码。这套字符集被 Windows98 系统采用。