

E&C

1993

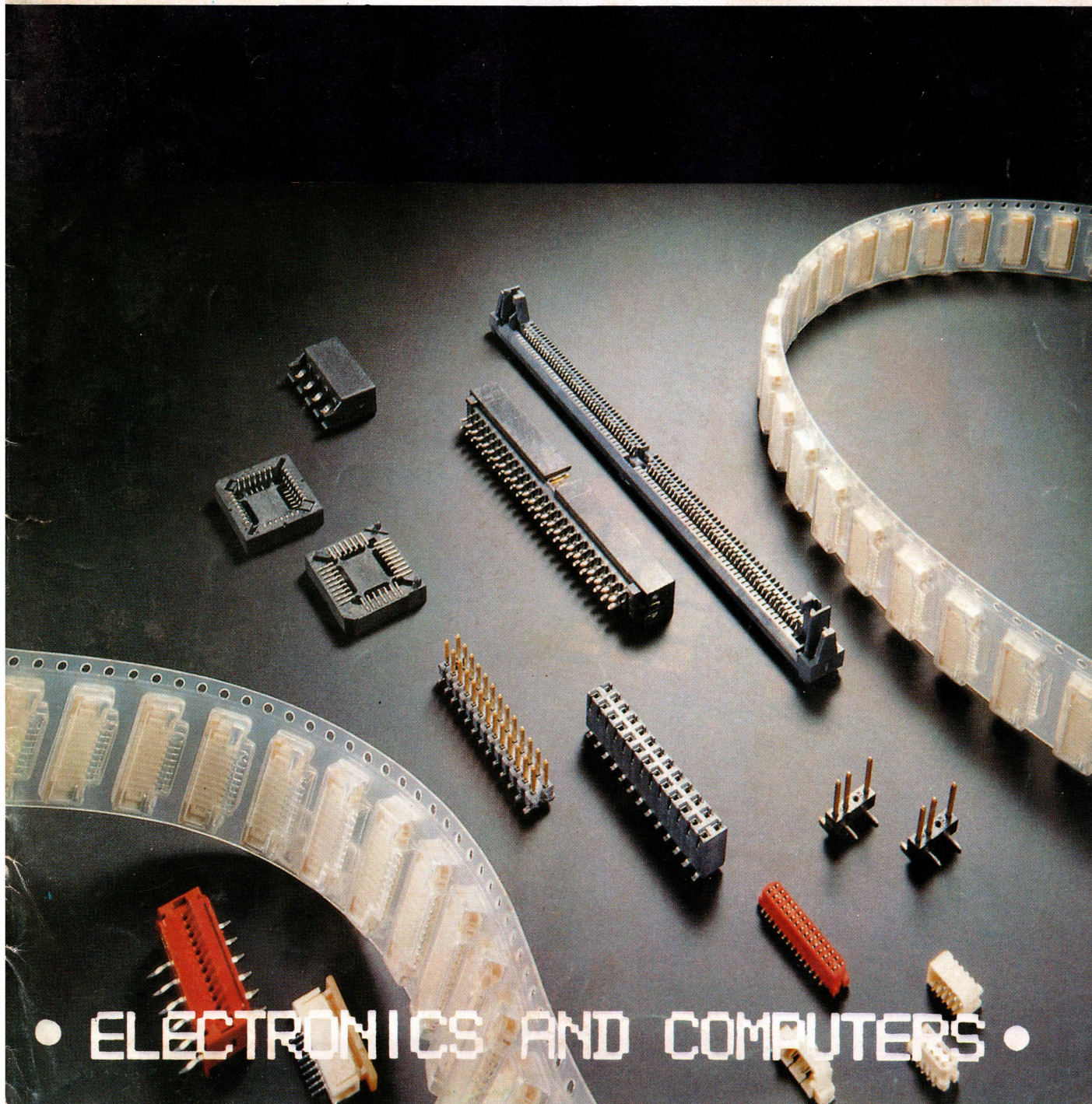
●一九九三年 ●总期第97期

4

電子

ISSN 1000-1077

與電腦



• ELECTRONICS AND COMPUTERS •

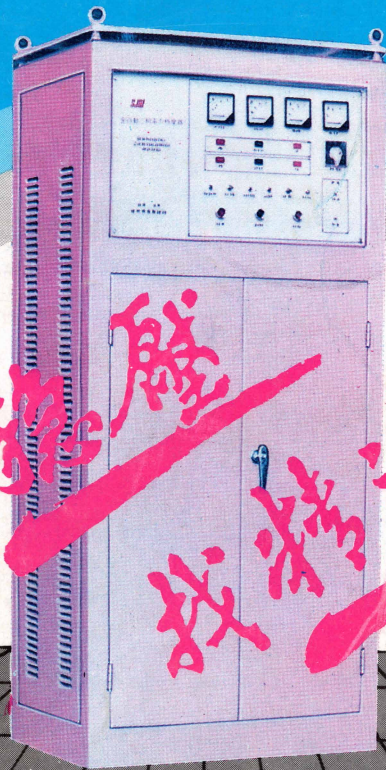
SJW系列 三相大功率稳压器



获国家电力电子产品合格证书 ● 获91年北京国际博览会银奖
邮电部电信总局指定必配电源 ● 稳压器标准由我厂起草制订

- 高效率 ● 低损耗 ● 波形畸变小 ● 稳压精度高
- 稳压范围宽 ● 运行可靠安全 ● 应变时间短
- 适用各用电单位 规格：20~1000kVA

银奖产品 ● 配套出口



要 您 騰 送！
我 滿 意！

经销单位
山东省邮电器材公司
湖北省邮电器材公司
四川省邮电器材公司
辽宁省邮电器材公司

经销单位
安徽省邮电器材公司
福建省邮电器材公司
陕西省邮电器材公司
浙江省邮电器材公司



上海市精达电子仪器厂

地址：上海市新闻路579号 邮编：200041

电话：2563294 2170089 电挂：6910

质优价廉 ● 现货供应 ● 资料备索 ● 服务完善 ● 国内代号：2-888 定价：1.60元

微机性能比较一览表

性能	AST PII 386SX/20	Compaq Prolinea 3/25s	AST PP3/25s	附注
处理器	386SX/20	386SX/25	386SX/25	
协处理器	80387 SX	80387 SX	80387 SX/25	
内存	2MB可扩至16MB	2 MB可扩至16 MB	2 MB可扩至16MB	AST支持1 MB或4MB Write Per Bit-70ns内存插条
快速缓冲	16KB SmartCache	无	16 KB SmartCache	
总线	ISA	ISA	EISA	
显示	内置VGA接口 800x600x16色可升级 1024x768x256色	COMPAQ Qvision 1024x768x16色 (用户可选)	内置Premium-VGA接口 1024x768x256或 1280x1024x16色	AST集成在母板上, 速度快, 可靠性强, 不占用扩展槽
扩展槽	6个ISA(直插)	3个ISA(横插)	6个EISA(直插)	AST具有优秀的扩展能力, 采用传统直插方式。
驱动器托架	5个(3外2内)	3个(2外1内)	5个(3外2内)	AST扩展能力高, 可配合磁带机
硬盘	80MB	84MB	80MB	AST可内置双硬盘
输入/输出	2串, 1并, 1鼠标	2串, 1并, 1鼠标	2串, 1并, 1鼠标	
升级途径	CUPID-32	无	CUPID-32	AST操作简单, 一次升级可将CPU、CACHE、MEMORY协处理器性能全部升级。
保密	开机密码, 外壳锁定	开机密码, 外壳锁定	开机密码, 外壳锁定	
电流	145W	145W	145W	AST采用ASIC技术、SMT工艺、耗电量低
外形尺寸	15.5"x6.25"x16.5"	15.9"x4.0"x15.1"	15.5"x6.25"x16.5"	AST外壳适中, 扩展能力高
保修	2年	1年	2年	AST提供2年保养, 是信心的保证
性能指标	4.4 MIPs	4.1 MIPs	5.5 MIPs	AST提供最佳性价比产品。
Power Meter V1.7	29.7 MHz	31.26 MHz	35.96 MHz	
Landmark V3.33	17	18.2	21.2	
Norton SI V6.0				

CR3240 打印机备件

维修手册	彩色色带盒	彩色色带芯	黑色色带盒	黑色色带芯
40 点阵字库	M51977P	MC34063AP1	NJM78L18A	电源小变压器
2SA1441	大上盖	控制面板	大底板	主板
字库板	链轮上盖	复位 1CM51953BL	UPC1093J	128KPSRAM
16 兆宋体字库	接口 LC92018B—ZB1	2 安保险	5 安保险	3 安保险
电磁组件	字车电机	字车	头缆	色带电机
打印头盖板	前面板	STA404A	STA434A	2SC1740SE
针 驱 动 门 阵 D65012CW—ZB2	2SC3331U	AR3240CPU	色带轮托	字车前挡片
链轮右轴套	字车轴左固件	彩色选择杆	压纸右抬杆	压纸左抬杆
释放调节杆	标志杆	手扭	色带轮	610 齿轮组 A
620 齿轮组 B	打印头	送纸器 SF—15DM	头 驱 动 芯 片 SLA7026M	链轮送纸器 PT—15XM

AR3240 打印机备件

用户手册	24、32、40 点阵字库	色带盒	针驱动门阵 D65006—15	步进皮带
RD20F	RD24F	RD3.9E—71	2SD1273	2SD1725ST
STA403A	RD39FB1—T	UPC339C	32KPSRAM	MSM82C53
桥 S2VB10、S4VB10	2SD1593	STA405A	色带轮	字库选择开关
字车后托	打印头前塑片	进纸电机	AR3240 字车	ASF 开关
感应器组	AR3240 字车电机	电源板	送纸器 EB—2	手扭
胶辊	字车架	接口 D65006CW—LC	CPU TMP90C041	变压器
2SB825	2SB1100	STA404A	右滚套	左滚套

AR2463 打印机备件

AR2463 字库	头驱动 PMM8713	2SD1308	UPC393C	接口 54610	CPU UPD7810HG—36
-----------	-------------	---------	---------	----------	------------------

CR3200 打印机备件

技术手册	40 点阵字库	CR3200 头缆	三极管 MP4104	4.7K 调节器	
------	---------	-----------	------------	----------	--

VS—16 打印机备件

左固纸板	右固纸板				
------	------	--	--	--	--

电子与电脑

一九九三年 总期第 97 期

目 录

• 综述 •

又闻又见——电视电话…………… 晓 亮(2)

• PC 用户 •

IBM PC 微型机 FORTRAN 语言绘图子程序设计
…………… 张 辉(4)
扩展 DOS 内部命令 DIR …………… 戴福华(7)
C 语言汉字显示技术 …………… 颜 飞(12)
灵活实用的打印程序 …………… 郝显峰(14)
巧用数组和 GATHER FROM 命令 …… 何钧军(16)

• 学习机之友 •

CEC-I 中华学习机系统子程序…………… 王志超(17)
ProDOS 系统内部结构剖析(续) …… 廖 凯(22)

• 语言讲座 •

FORTH 语言讲座 …………… 丁志伟(24)

• 初、中级程序员软件水平考试辅导 •

C 语言试题与分析(二) …………… 赵国瑞(28)

• 新书与软件 •

邮购消息 …………… (27)

• 学用单片机 •

BJS-98 硬件、软件典型实验
…………… 袁 涛 魏 峰(31)
MCS-51 单片单板机监控程序分析(下)
…………… 江琪 刘葳(34)

• 电脑巧开发 •

IBM-PC 与彩色电视机接口 …… 蒋海明 王 辉(37)

• 维修经验谈 •

彩色显示器工作原理简述及故障诊断(中)
…………… 胡野红(40)
电源引起的特殊故障一例 …………… 王 飙(41)
《对单片机旅馆客房门卫系统中 P212 使用的异议》
…………… 秦化渤(42)

• 电脑游戏机 •

第五讲 游戏程序的编程技巧(下) …… 于 春(43)
实用程序——彩色电视信号发生器 …… 王长宾(46)

• 电脑通信 •

• 传真机讲座 •
(二) 传真机原理(上)——传真图象数据的编码和解码
…………… 张建军 后俊堂 张景生(47)
• 电脑通信大家谈 •
PC 间能传送文件的程序…………… (51)

• IC 电路应用 •

开关电源用集成控制器
…………… 李秀华(53)

• 读者联谊 •

回音壁 …………… (52)
咨询 …………… (55)

机械电子工业部电子工业出版社主办

编辑、出版:《电子与电脑》编辑部
(北京 173 信箱 邮政编码:100036)

印刷:北京三二〇九厂

国内总发行:北京报刊发行局

国内统一刊号:CN11-2199

邮发代号:2-888

国外代号:M924

出版日期:每月 23 日

主编:王惠民 特约编审:苏子栋

责任编辑:杨逢仪

订购处:全国各地邮电局

国外总发行:中国国际图书贸易总公司

(北京 399 信箱 邮政编码 100044)

广告经营许可证:京海工商广字 147 号

定价:1.60 元

74
对
胶

又闻又见——电视电话

晓 亮

1876年,随着贝尔的一声呼喊,人类开创了使用电话的历史。到如今,电话已经发展成为一个庞大的家族,新产品层出不穷,电脑电话,BP机,大哥大……在人们的社会生活中发挥了极为重要的作用。在它们中间,一支新军正悄然崛起,这就是电视电话。其实,电视电话早在50多年前就已问世。它经历了艰难的发展历程,直到最近几年,随着集成电路技术突飞猛进,成本不断下降,它才逐渐为人们所接受。现在,国外的电视电话已不再为少数富翁所独有,一般消费者都能买得起,而且,可以很方便地建立一个电视电话系统。

让我们把镜头推回到1939年。在纽约世界商品博览会的AT&T展厅里,迪克·特雷西始终在他的手腕上带有一个双向电视电话,这便是它的首次亮相。在那些年代里,电视电话存在一个主要问题是需要使用庞大的设备。用于电视电话的机箱特别大,例如,罗伯特研究所1978年研制的电视电话盒。而现在,电视电话带有很小的且容易瞄准的摄像机,所有元件(除话筒外)都可装在一个约为标准电话台空间一半的容器中。

在50年代末期,业余无线电爱好者研制了慢速扫描电视(SSTV),以便使它们能在联邦通信委员会(美国)允许的低于450MHz的狭窄频宽(3500KHz)范围内发送图像。在旧式的慢速扫描形式中,一幅画面包括128行(正规的电视有525行,而Colby电视电话有260行)。此形式具有一个1/15秒的垂直扫描率(而正规的电视是1·5734秒或63微秒)。

那时候没有高速A/D转换器之类的器件,故必须要静候3~10秒等待一个整帧,扫描才完毕。如果你移动,就会使图像模糊,就象在摄影技术发展初期,透镜需打开10秒以得到合适的曝光量。因为SSTV系统需花10秒钟时间才在屏幕上描绘一幅图像,所以早期的该系统使用一个长余辉荧光阴极射线管,这样在接收端便能逐次地看到逐行检索的扫描束产生的光点。当操作到屏幕最下面一行时,图像的顶部已经减弱了。

以后,出现了模拟扫描转换器,可使你输入慢速扫描信号并输出1/60秒的静止电视图像,不等图像顶部减弱就能在一个正规的快速扫描电视上看到整个图像。但是这些模拟设备既庞大又昂贵。一些部门常年使用慢速扫描电视来进行远距离监视;许多航天飞行器也使用SSTV从空间传输图像,“海盗号”和“旅行者号”太空探测器就是很好的例子,美国加州Pasadena的喷气推进实验室一直是使用SSTV的支持者,因为它有非常高的效率,可用在长距离的航天飞行器上。

在70年代初期有了突破性进展,开发出了使用数

字技术存储和恢复图像的方法。先使用多固态扫描转换器中的许多移位寄存器来存储数据,再将其转换成视频数据,存入动态随机存取存储器集成电路块中。

大约在两年前,一些日本公司推出了家庭电视电话设备,其售价约为500~600美元。这些设备只有一个鞋盒那么大,并有一个内装的摄像机和监视器。摄像机的位置是固定的,必须把目标移到摄像机的视野内。该设备使用一个对电话线上的噪音特别敏感的已调幅的协议。存在的问题也许是96×962像素的低分辨率的监视器。

本文所介绍的电视电话已克服了许多问题。它具有一个较小体积的随意移动和取向的摄像机,一个高分辨率、50级灰度的监视器,一个免除噪声的脉宽调制传输系统,以及一个高技术的外观。看上去它似乎应该是办公桌上需要的一个办公设备。该设备可通过电话线传输和接收一个静止的具有64级灰度的高分辨率(200×24像素)的电视画面。

此系统的组成包括电视电话基本设备、小型CCD(电荷耦合器件)摄像机组件和电视监视器。为了发送和接收一个视频图像,当然需要两个一模一样的这样的装置。整个系统框图如图所示。它使用标准单色NTSC制视频信号用于输入和输出。任何标准的电视摄像机、摄像记录仪、录像机、激光视盘等设备的视频输出被送入J2,然后,对该输出进行放大、数字化,并把它转变成在1700~3500Hz范围内的一系列音频音调。该频率范围较低,足以能通过标准电话线发送出去,记录在一个低成本的音频盒式磁带上,或在双向接收机之间传输。两个电话都处于静默状态时,发送或接收一个静止的电视图像要用9—12秒。被发送的数据是以约为32Kbit/秒的数据速率进行脉宽调制的,这个速率约为新的ISDN系统(国际标准数字网)的一半。

电视电话基本设备包括用于电视电话上的电子设备,以及一个带有八个集成电路(IC)和其它相应元件的双面印刷线路板。此板还包括用于电源和电话线的输入输出插口(J)、四个按钮开关和发光二极管(LED)指示器。由于基本设备使用专用大规模集成电路,特别是电视控制器集成电路(IC7),大大减少了所需的元器件,因为该集成电路一身兼职多种功能,这一点在后面的分析中便可看到。

与原型机(样机)一起使用的电视摄像机是一个ChinanCCD组件。它包括两个很小的集成电路板,在板的两面都有表面安装元件。它的尺寸为2.5×2×

7cm,重 60 克。该摄像机包括一个固定焦点的 4.5mm 的 F1.8 透镜,它处于通常和较宽的角度之间。它距离摄像机 45~60cm 时,一个人的面孔约占据屏幕的 2/3。视野的深度从 15cm 到无穷远,这意味着距离摄像机 15cm 之外的任何东西都将在焦点上。

这个组件有一个手动电子快门,其速度范围在 1/60 秒~1/1500 秒。它提供一个自始至终有 2 勒克斯照度的可用的图像。它在 NTSC 制式中使用一个有 80000 个像素的 8.5 毫米的 MOS 电视摄像管器件,可提供 240 条垂直线×230 条水平线的分辨率,行频是 15.734KHz,场频是 59.94Hz(对于用于欧洲的 PAL 制式摄像机组件也是适用的)。该输出是一个标准的复合视频信号(1VP-P,75 欧姆)。

摄像机在 7~12 伏直流输入电压的条件下进行操作。电视电话基本设备经 J3 为摄像机提供电源。该设备中有一个跳接器(J9),使 J3 获得 12V 或 5V 直流电压。如果你使用同样的摄像机,就必须确保把 J9 跳接器置于 12V 位置。

摄像机组件被置于一个圆形罩壳中,用尼龙线捆在监视器的天线上。因为视频输入直接进入监视器而无需途经天线,所以天线转动不影响信号的接收,这就能很轻易地使摄像机指向任何方向。这比只能把主要目标置于摄像机视野中而无法顾及周围目标的固定摄像机系统具有更大的灵活性。

电视监视器安装在电视电话基本设备的顶部。这

里所用的是一个未调制的、旧器新用的索尼黑白电视机(FD-250 型)。该设备有一个复合视频输入插口(J6),因此,它能从电视电话基本设备中直接地接收电视图像。该图像特别鲜明和清晰,但这在一定程度上取决于你所使用的摄像机的位置。

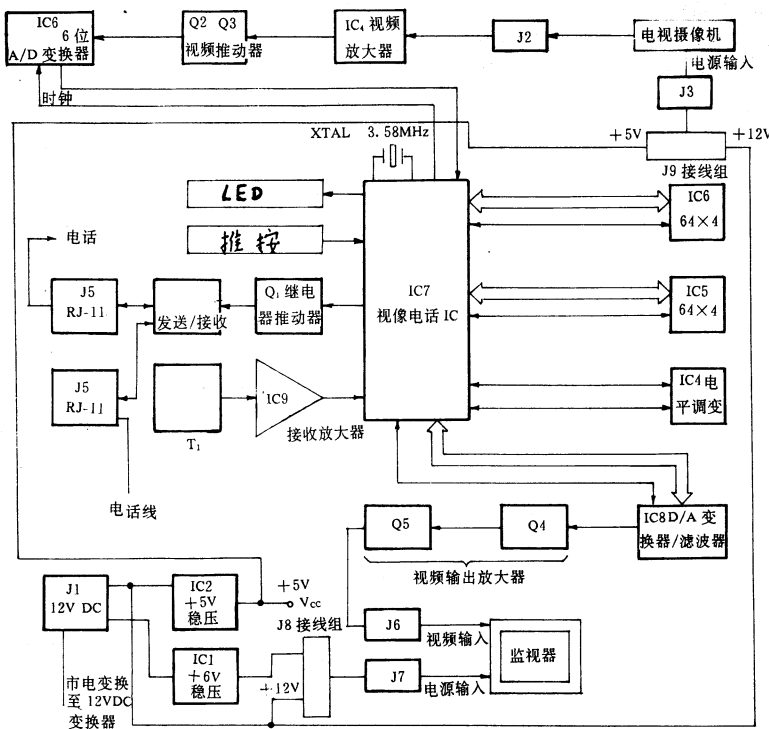
我们现在来讨论一下,一个电视图像是怎样从一个电视电话发送到另一个电话的。来自摄像机的电视信号进入 J2,至高速 A/D 转换器 IC3 输入端。在高速 A/D 转换器中,输入信号被转换成一组 64 个的通一断状态,再译码成一个 6 位字节。一个 0V 模拟信号输入产生一个为“000000”的输出代码,一个 1.5V 的模拟信号输入产生一个为“111111”的输出代码。然后,将 000000 和 111111 之间的一个数的 6 位数字输出码(表示一个在 0V 和 1.5V 之间的电视输入电平)送入电视电话控制器集成电路 IC7。在这里,数字输入被转变成一系列的在 1200—3450Hz 范围内的音频音调,并输出至变压器线圈 T1,再通过继电器 RY1 和 RJ-11 插口 J5 将信号隔离,并耦合至电话线。在传输开始时,一个短暂的 3442Hz 的音调猝发信号触发正在接收的电视电话进入接收方式。

当一个 3442Hz 的音调出现在 IC7 上时,它自动地使系统转换至接收方式。在 J5 上的来自电话线的音调(表示电视信息)耦合到 T1 上,由 T1 的次级线圈送至运算放大器 IC9 放大音频音调,并使它们出现在 IC7 上。然后,电视控制器集成电路(IC7)把音频音调转变成 6 位数字代码,并把电视画面以数字形式储存在动态随机存取存储器 IC5 和 IC6 中。接着,数字信息被输出到一个专用的 Colby 程序逻辑阵列 IC8 中,IC8 起到一个 D/A 转换器和滤波器的作用。它把数字信号转变回复合视频信号,该信号进一步被 Q4 和 Q5 放大,再送到电视输出插口 J6 中。电视形式是标准的 NTSC 制(或者在特别情况下是 PAL 制),因此能把它输入到任何录像机或电视监视器中。

12 伏直流电源由 J2 输入至电压调整器 IC1 和 IC2 上。IC2 供应 +5 伏电压给大多数电路。IC1 产生 6 伏电压。跨接器盒 J8 和 J9 用来选择输出电压以便向摄像机和监视器供电。J8 可以在 J7 上提供 6 伏或 12 伏的直流电压分别供给索尼监视器或使用 12 伏的监视器。J9 则提供 12 伏、5 伏电压给 J3 以用于本文中提到的 CCD 摄像机组件或其它摄像机。整机功率约为 5W。

电视电话的操作很简单。“冻结”按钮捕捉并保持来自摄像机的电视画面,而“发送”按钮则通过电话线传输图像。按下“自动”按钮则每隔 38 秒自动地冻

(下转第 55 页)



图

IBM PC 微型机 FORTRAN 语言绘图子程序设计

沈阳东北工学院机械二系(110006) 张 辉

编者按 虽然 MS-FORTRAN V5.0 已支持绘图功能,但对手头只有较低版本的用户,或者对绘图功能扩充原理感兴趣的读者,本文提供的设计方法和程序还是有参考价值的,原文还附有画直线的 Bresenham 法框图,限于篇幅只好略去。

FORTRAN 语言具有极强的工程计算能力,在 IBM PC 机上使用的为 Microsoft FORTRAN。但是与 BASIC 语言相比较,它的一个致命弱点就是不能直接进行屏幕绘图,因而限制了它的广泛应用。为弥补这一缺点,笔者开发了一系列独具特色的 FORTRAN 语言绘图子程序。这些子程序可支持多种 BASIC 语言不支持的显示卡,这些程序的开发增加了 FORTRAN 语言的使用范围,方便了具有图形功能要求的 FORTRAN 程序设计。目前,这些绘图子程序已成功的应用于频谱分析、数据采集和处理等软件中,收到了满意的效果。

在 IBM PC 微型机上进行屏幕绘图至少应具有以下三种基本绘图子程序:设置屏幕状态、在屏幕上任一点开始写字符串和画直线。下面介绍用 8088 汇编语言编写的可供 FORTRAN 语言直接调用的三个基本绘图子程序的实现及使用方法,给出这些子程序的源程序清单供读者使用、参考和开发。

一、设置屏幕方式

IBM PC 系列微机的彩色/图形监视适配器有多种工作方式,首先必须将其设置为图形方式,方可以在其上绘制图形。完成该功能可采用固化在 ROM 中的 BIOS 功能 INT 10H 来实现,子程序清单见附录。调用该子程序的 FORTRAN 语句为

```
CALL CRT(K)
```

语句中 K 为屏幕工作方式代号。表 1 中列出了一些常用的屏幕方式及代号。该子程序也可用来清屏,与 BASIC 语言中的 CLS 语句类似。

表 1 常用屏幕方式代号

代号	方式	显示卡
0	40×25 黑白文本	CGA, EGA, Color 400
1	40×25 彩色文本	CGA, EGA, Color 400
2	80×25 黑白文本	CGA, EGA, Color 400
3	80×25 彩色文本	CGA, EGA, Color 400
4	320×200 彩色图形	CGA, EGA, Color 400
5	320×200 黑白图形	CGA, EGA, Color 400
6	640×200 黑白图形	CGA, EGA, Color 400
13	320×200 16 色图形	EGA
14	640×200 16 色图形	EGA
15	640×350 黑白图形	EGA
16	640×350 16 色图形	EGA
66	640×400 色图形	Color 400

二、屏幕上任一点开始写字符串操作

在许多要求有绘图功能的软件中,都要求有能在

任一点开始写字符串的功能,如画坐标轴时标注坐标值。BASIC 语言中虽有定位光标和写字符串的功能,但只能以行和列为单位开始写字符串,不能满足上述要求,因此,我们采用 BIOS 功能的 INT 10H 的写点功能来实现。

采用写点功能进行写字符串操作,首先要解决的问题是如何获得各字符的点阵数据。完成该任务可有如下两种方法:一是可直接访问 BIOS 的图形方式字符发生器的数据区,获取字符点阵信息;另一种方法是将这些点阵信息取出,作为数据加入到字符串操作子程序中。前一种方法可节省一些内存,但使用灵活性差,只能显示 ASCII 字符。后者虽然浪费一些内存,但增加了使用灵活性,可以自己造一些特殊字符和图案代替 ASCII 字符点阵信息。在本文中我们采取了第二种方法,收到了良好的效果。在该子程序中我们采用了与 BIOS 数据区相同的格式(即用 8×8 点阵显示一字符),可显示 ASCII 码值为 32~127 的所有字符。

调用该子程序的 FORTRAN 语句为:

```
CALL WRA (A, IX1, IY1, ICOLOR);
```

```
CALL WRA ('How are you ! $', IX1, IY1, ICOLOR)
```

其中, A 为字符型变量,必须以“\$”结尾,以表示字符串末尾; IX1, IY1 为字符串的第一个字符的左下角坐标值(对于 640×200 方式,屏幕左上角坐标为(0, 0),右下角坐标为(639, 199)); ICOLOR 为字符色彩代号,对于黑白图形方式,0 为黑色,1 为白色;对于具有 16 种色彩的显示卡,其值如表 2 所示。

表 2 屏幕彩色代号表

代 号	彩 色	代 号	彩 色
0	黑 色	8	深 灰
1	蓝 色	9	浅 蓝
2	绿 色	10	浅 绿
3	青 色	11	浅 青
4	红 色	12	淡 红
5	洋 红	13	品 红
6	棕 色	14	黄 色
7	淡 灰	15	白 色

三、画线子程序

在屏幕绘图中,经常需要绘制曲线。曲线可用一系列直线段来代替。在画线中采用 Bresenham 的算法调用该子程序的 FORTRAN 语句为:

```
CALL LINE (IX1, IY1, IX2, IY2, ICOLOR)
```

语句中 IX1, IY1, IX2, IY2 分别为直线的起点和终点的屏幕坐标值, ICOLOR 为直线的色彩代号,同写字符串子程序 WRA。

四、使用方法

为能使用上述子程序,必须用 MASM·EXE 将其汇编成目标文件,然后用 LINK·EXE 文件将它们

与FORTRAN程序形成的目标文件连接在一起。也可以用库管理程序LIB·EXE将这些目标文件加入到FORTRAN·LIB文件中,像FORTRAN的内部过程一样直接调用,LINK程序自动在FORTRAN·LIB

中寻找这些子程序。为了能将屏幕图形硬拷贝,我们还编制了各种屏幕状态的M2024/M1724打印机驱动程序。图形完成后,只要简单地按下拷屏键,即可在打印机上将屏幕图形拷贝出来。

设置屏幕方式子程序:

```

FRAME STRUC
SAVEDS DW ?
SAVEBP DW ?
RETADD DD ?
K DD ?
FRAME ENDS
PUBLIC CRT
CODE SEGMENT 'CODE'
ASSUME CS, CODE
CRT PROC FAR
PUSH BP
PUSH DS
MOV BP, SP
LES SI, [BP], K
MOV AX, ES, [SI]
MOV AH, 0
MOV BX, 0
INT 10H
POP DS
POP BP
RET 4
CRT ENDP
CODE ENDS
END

```

```

INC BX
DEC SI
JNZ CH1
POP DX
POP CX
INC DI
MOV BL, ES, [DI]
CMP BL, 24H
JZ CCCC
ADD CX, 8
JMP CC2

```

```

DB 00H, 00H, 00H, 00H, 00H, 030H, 030H, 000H,
00H,
DB 006H, 00CH, 018H, 030H, 060H, 0C0H, 080H,
00H, /
DB 07CH, 0C6H, 0CEH, 0DEH, 0F6H, 0E6H, 07CH,
00H, 0
DB 030H, 070H, 030H, 030H, 030H, 030H, 0FCH,
00H, 1
DB 078H, 0CCH, 00CH, 038H, 060H, 0CCH, 0FCH,
00H, 2
DB 078H, 0CCH, 00CH, 038H, 00CH, 0CCH, 078H,
00H, 3
DB 01CH, 03CH, 06CH, 0CCH, 0FEH, 00CH, 01EH,
00H, 4
DB 0FCH, 0C0H, 0F8H, 00CH, 00CH, 00CH, 078H,
00H, 5
DB 038H, 060H, 0C0H, 0F8H, 0CCH, 0CCH, 078H,
00H, 6
DB 0FCH, 0CCH, 00CH, 018H, 030H, 030H, 030H,
00H, 7
DB 078H, 0CCH, 0CCH, 078H, 0CCH, 0CCH, 078H,
00H, 8
DB 078H, 0CCH, 0CCH, 07CH, 00CH, 018H, 070H,
00H, 9
DB 000H, 030H, 030H, 000H, 000H, 030H, 030H,
00H, :
DB 000H, 030H, 030H, 000H, 000H, 030H, 030H,
006H, :
DB 018H, 030H, 060H, 0C0H, 060H, 030H, 018H,
000H, <
DB 000H, 000H, 0FCH, 000H, 000H, 0FCH, 000H,
000H, =
DB 060H, 030H, 018H, 00CH, 018H, 030H, 060H,
000H, >
DB 078H, 0CCH, 00CH, 018H, 030H, 000H, 030H,
000H, ?
DB 07CH, 0C6H, 0DEH, 0DEH, 0DEH, 0C0H, 078H,
000H, @
DB 030H, 078H, 0CCH, 0CCH, 0FCH, 0CCH, 0CCH,
000H, A
DB 0FCH, 066H, 066H, 07CH, 066H, 066H, 0FCH,
000H, B
DB 03CH, 066H, 0C0H, 0C0H, 0C0H, 066H, 03CH,
000H, C
DB 0F8H, 06CH, 066H, 066H, 066H, 06CH, 0F8H,
000H, D
DB 0FEH, 062H, 068H, 078H, 068H, 062H, 0FEH,
000H, E
DB 0FEH, 062H, 068H, 078H, 068H, 060H, 0F0H,
000H, F
DB 03CH, 066H, 0C0H, 0C0H, 0CEH, 066H, 03EH,
000H, G
DB 0CCH, 0CCH, 0CCH, 0FCH, 0CCH, 0CCH,
0CCH, 000H, H
DB 078H, 030H, 030H, 030H, 030H, 030H, 078H,
000H, I
DB 01EH, 00CH, 00CH, 00CH, 0CCH, 0CCH, 078H,
000H, J
DB 0E6H, 066H, 06CH, 078H, 06CH, 066H, 0E6H,
000H, K
DB 0F0H, 060H, 060H, 060H, 062H, 066H, 0FEH,
000H, L
DB 0C6H, 0EEH, 0FEH, 0FEH, 0D6H, 0C6H, 0C6H,
000H, M
DB 0C6H, 0E6H, 0F6H, 0DEH, 0CEH, 0C6H, 0C6H,
000H, N
DB 038H, 06CH, 0C6H, 0C6H, 0C6H, 06CH, 038H,
000H, O
DB 0FCH, 066H, 066H, 07CH, 060H, 060H, 0F0H,
000H, P
DB 078H, 0CCH, 0CCH, 0CCH, 0DCH, 078H, 01CH,

```

在屏幕任一点开始写字符串子程序

```

PUBLIC WRA
FRAME STRUC
SAVEDS DW ?
SAVEBP DW ?
RETAD DD ?
COLOR DD ?
Y DD ?
X DD ?
CHA DD ?
FRAME ENDS
CODE SEGMENT 'CODE'
WRA PROC FAR
ASSUME CS, COD
PUSH BP
PUSH DS
PUSH CS
POP DS
MOV BP, SP
LES DI, [BP], COLOR
MOV AX, ES, [DI]
LES DI, [BP], X
MOV CX, ES, [DI]
LES DI, [BP], Y
LES DI, [BP], CHA
MOV BL, ES, [D1]
CMP BL, 24H
JZ CCCC
PUSH CX
PUSH DX
XOR BH, BH
SUB BL, 20H
SHL BX, 1
SHL BX, 1
SHL BX, 1
PUSH AX
MOV AX, 8
MOV SI, AX
SUB DX, 8
MOV AX, OFFSET CHR
ADD BX, AX
POP AX
MOV AH, [BX]
CALL PUTBYTES
INC DX

```

```

CCCC, POP DS
POP BP
RET 16
WRA ENDP
PUTBYTES PROC NEAR
PUSH AX
PUSH BX
PUSH DX
PUSH CX
PUSH SI
MOV BX, 8
MOV SI, BX
MOV BX, AX
MOV AH, 12
MOV AL, BL
CMP BH, 7FH
JNC P1
XOR AL, AL
INT 10H
INC CX
SHL BH, 1
DEC SI
JNZ P2
POP SI
POP CX
POP DX
POP AX
RET
PUTBYTES ENDP

```

```

DB 000H, 000H, 000H, 000H, 000H, 000H, 000H,
000H,
DB 030H, 078H, 078H, 030H, 030H, 000H, 030H,
000H, !
DB 06CH, 06CH, 06CH, 000H, 000H, 000H, 000H,
000H, "
DB 06CH, 06CH, 0FEH, 06CH, 0FEH, 06CH, 06CH,
000H, #
DB 030H, 07CH, 0C0H, 078H, 00CH, 0F8H, 030H,
000H, $
DB 000H, 0C6H, 0CCH, 018H, 030H, 066H, 0C6H,
000H, %
DB 038H, 06CH, 038H, 076H, 0DCH, 0CCH, 076H,
000H, &
DB 060H, 060H, 0C0H, 000H, 000H, 000H, 000H,
000H, '
DB 018H, 030H, 060H, 060H, 060H, 030H, 018H,
000H, (
DB 060H, 030H, 018H, 018H, 018H, 030H, 060H,
000H, )
DB 000H, 066H, 03CH, 0FFH, 03CH, 066H, 000H,
000H, *
DB 000H, 030H, 030H, 0FCH, 030H, 030H, 000H,
000H, +
DB 000H, 000H, 000H, 000H, 000H, 030H, 030H,
060H, ,
DB 000H, 000H, 000H, 0FCH, 000H, 000H, 000H,
000H, -

```



```

000H,Q
DB 0FCH,066H,066H,07CH,06CH,066H,0E6H,
000H,R
DB 078H,0CCH,0E0H,070H,01CH,0CCH,078H,
000H,S
DB 0FCH,0B4H,030H,030H,030H,030H,078H,
000H,T
DB 0CCH,0CCH,0CCH,0CCH,0CCH,0CCH,
0FCH,000H,U
DB 0CCH,0CCH,0CCH,0CCH,0CCH,078H,030H,
000H,V
DB 0C6H,0C6H,0C6H,0D6H,0FEH,0EEH,0C6H,
000H,W
DB 0C6H,0C6H,06CH,038H,038H,06CH,0C6H,
000H,X
DB 0CCH,0CCH,0CCH,078H,030H,030H,078H,
000H,Y
DB 0FEH,0C6H,08CH,018H,032H,066H,0FEH,
000H,Z
DB 078H,060H,060H,060H,060H,060H,078H,
000H,[
DB 0C0H,060H,030H,018H,00CH,006H,002H,
000H,\
DB 078H,018H,018H,018H,018H,018H,078H,
000H,]
DB 010H,038H,06CH,0C6H,000H,000H,000H,
000H,^
DB 000H,000H,000H,000H,000H,000H,000H,
0FFH,-
DB 030H,030H,018H,000H,000H,000H,000H,
000H,
DB 000H,000H,078H,00CH,07CH,0CCH,076H,
000H,a
DB 0E0H,060H,060H,07CH,066H,066H,0DCH,
000H,b
DB 000H,000H,078H,0CCH,0C0H,0CCH,078H,
000H,c
DB 01CH,00CH,00CH,07CH,0CCH,0CCH,076H,
000H,d
DB 000H,000H,078H,0CCH,0FCH,0C0H,078H,
000H,e
DB 038H,06CH,060H,0F0H,060H,060H,0F0H,
000H,f
DB 000H,000H,076H,0CCH,0CCH,07CH,00CH,
0F8H,g
DB 0E0H,060H,06CH,076H,066H,066H,0E6H,
000H,h
DB 030H,000H,070H,030H,030H,030H,078H,
000H,i
DB 00CH,000H,00CH,00CH,00CH,0CCH,0CCH,
078H,j
DB 0E0H,060H,066H,06CH,078H,06CH,0E6H,
000H,k
DB 070H,030H,030H,030H,030H,030H,078H,
000H,l
DB 000H,000H,0CCH,0FEH,0FEH,0D6H,0C6H,
000H,m
DB 000H,000H,0F8H,0CCH,0CCH,0CCH,0CCH,
000H,n
DB 000H,000H,078H,0CCH,0CCH,0CCH,078H,
000H,o
DB 000H,000H,0DCH,066H,066H,07CH,060H,
0F0H,p
DB 000H,000H,076H,0CCH,0CCH,07CH,00CH,
01EH,q
DB 000H,000H,0DCH,076H,066H,060H,0F0H,
000H,r
DB 000H,000H,07CH,0C0H,078H,00CH,0F8H,
000H,s
DB 010H,030H,07CH,030H,030H,034H,018H,
000H,t
DB 000H,000H,0CCH,0CCH,0CCH,0CCH,076H,
000H,u
DB 000H,000H,0CCH,0CCH,0CCH,078H,030H,
000H,v
DB 000H,000H,0C6H,0D6H,0FEH,0FEH,06CH,

```

```

000H,w
DB 000H,000H,0C6H,06CH,038H,06CH,0C6H,
000H,x
DB 000H,000H,0CCH,0CCH,0CCH,07CH,00CH,
0F8H,y
DB 000H,000H,0FCH,098H,030H,064H,0FCH,
000H,z
DB 01CH,030H,030H,0E0H,030H,030H,01CH,
000H,{
DB 018H,018H,018H,000H,018H,018H,018H,
000H,|
DB 0E0H,030H,030H,01CH,030H,030H,0E0H,
000H,}
DB 076H,0DCH,000H,000H,000H,000H,000H,
000H,~
DB 000H,010H,038H,06CH,0C6H,0C6H,0FEH,
000H,Δ
CODE ENDS
END
画直线子程序
FRAME STRUC
SADS DW ?
SABP DW ?
READ DD ?
C1 DD ?
AY2 DD ?
AX2 DD ?
AY1 DD ?
AX1 DD ?
FRAME ENDS
DATA SEGMENT FARA PUBLIC 'DATA'
X1 DW ?
X2 DW ?
Y1 DW ?
Y2 DW ?
COLOR DW ?
DELTAX DW ?
DELTAY DW ?
HALFX DW ?
HALFY DW ?
COUNT DW ?
DATA ENDS
DGROUP GROUP DATA
PUBLIC LINE
CODE SEGMENT 'CODE'
LINE PROC FAR
ASSUME CS, CODE
ASSUME DS, DGROUP
PUSH BP
PUSH DS
MOV BP, SP
LES SI, AX1. [BP]
MOV AX, [SI]
MOV X1, AX
LES SI, AY1. [BP]
MOV AX, [SI]
MOV Y1, AX
LES SI, AX2. [BP]
MOV AX, [SI]
MOV X2, AX
LES SI, AY2. [BP]
MOV AX, [SI]
MOV Y2, AX,
LES SI, C1. [BP]
MOV AX, [SI]
MOV COLOR, AX,
MOV AX, Y2, AX, Y2
SUB AX, Y1
MOV SI, 1
JGE STOY
MOV SI, -1
NGE AX
STOY: MOV DELTAY, AX
MOV AX, X2
SUB AX, X1
MOV DI, 1

```

```

JGE STOX
MOV DI, -1
NEG AX
STOX: MOV DELTAX, AX
MOV AX, DELTAX
CMP AX, DELTAY
JL STEEP
CALL EASY
JMP FINISH
STEEL: CALL STP
FINH: POP DS
POP BP
RET 20
EASY: LINE ENDP
PROC NEAR
MOV AX, DELTAX
SHR AX, 1
MOV HALFX, AX
MOV CX, X1
MOV DX, Y1
MOV BX, 0
MOV AX, DELTAX
MOV COUNT, AX
CALL DOTPLOT
ADD CX, DI
ADD BX, DELTAY
CMP BX, HALFX
JLE DCNT
SUB BX, DELTAX
ADD DX, SI
DEC COUNT
JGE NDOT
RET
EASY: ENDP
STP: PROC NEAR
MOV AX, DELTAY
SHR AX, 1
MOV HALFY, AX
MOV CX, X1
MOV DX, Y1
MOV BX, 0
MOV AX, DELTAY
MOV COUNT, AX
CALL DOTPLOT
ADD DX, SI
ADD BX, DELTAX
CMP BX, HALFY
JLE DCN2
SUB BX, DELTAY
ADD CX, DI
DUN2: DEC COUNT
JGE NEWDO2
RET
STEEL: ENDP
DOTPLOT: PROC NEAR
PUSH BX
PUSH CX
PUSH DX
PUSH AX
PUSH SI
PUSH DI
MOV AX, COLOR
MOV AH, 12
MOV BX, 0FH
INT 10H
POP DI
POP SI
POP AX
POP DX
POP CX
POP BX
RET
DOTPLOT: ENDP
CODE ENDS
END

```


扩展 DOS 内部命令 DIR

广东汕头龙眼北路 2 号(515041) 戴福华

用法: XDIR [+|-ADHRSV][filespec][D][/
S][P][W][?][nn][R[N|E|D
|S...]]

说明:

(一)本程序需用 Turbo C++ 编译,以便在汉字系统下正常运行。

(二)选项含义:

a)+|ADHRSV

根据目录项属性列出匹配目录项。“+”/“-”表明只具有/不具有该属性的目录项。缺省为所有目录项属性。属性选项含义为:

A 档案位 D 目录位 H 隐藏属性 R 只读属性
S 系统属性 V 卷标位

b)filespec

文件说明,可含“*”、“?”通配符。缺省为当前目录下的所有文件。若只指定目录,则为该目录下的所有文件。

c)/D 列出指定目录或缺省目录的目录树。

d)/S 搜索对指定目录或缺省目录的所有子目录进行。

e)/P 满屏后不暂停。缺省为显示 23 行后暂停,暂停行数可用/nn 参数加以改变。

f)/W 以每行 5 个目录项的宽行格式显示。

g)/? 显示帮助信息。

h)/nn nn 为 1 或 2 位十进制数,决定显示暂停行数,缺省为 23。

i)[R[N|E|D|S...]]

按给定的先后次序进行排序。多个排序码,则按先后次序决定优先权。缺省为不排序。排序码选项说明:

N-文件名 E-扩展名 D-日期 S-长度

例子: XDIR C:\213 /S /REN+HRDS /8

XDIR D:\FOXBASE /D /RS /13

(三)选择项的缺省值可由环境变量 XDIRCMD 加以改变,用 DOS 内部命令 SET 完成。

格式为 SET XDIRCMD=[/R[N|E|D|S...]]
[/S][D][P][W][Lnn]。

其中除 Lnn 外均同上。而 Lnn 则为显示暂停行数。可放在 AUTOEXEC.BAT 中。

例子: SET XDIRCMD=/S L13 /REN

SET XDIRCMD=/RS L8 /S

/* TURBO C++ 源程序 XDIR.C */

```
#include <dir.h>
#include <dos.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <alloc.h>
#include <process.h>
#define AND &&
#define OR ||
#define BACKLASH '\\\'
#define noSort 0
#define byName 1
#define byExt 2
#define byDate 3
#define bySize 4
struct FAMILY_DIR
{ int subdir_num;
  char * father_dir;
  char * sub_dir[100];
};
struct FILE_SORT
{ char name[9],ext[5];
  int date,time,attrib;
  long size;
  struct FILE_SORT * next_pr;
};
int subdir_count=0,disp_line_num=1;
int screen_line=23,flag_pause=1;
/* 函数 born(struct FAMILY_DIR * d)从 d->father_dir 中得到子目录表 */
void born(struct FAMILY_DIR * d)
{ int i,j;
  int s_attrib=FA_RDONLY | FA_HIDDEN | FA_SYSTEM | FA_DIREC | FA_ARCH;
  unsigned lenth;
  struct fblk ff;
  char sub_dir[81]=" ";
  d->subdir_num=0;
  strcpy(ff.ff_name," ");
  strcpy(sub_dir,d->father_dir);
  strcat(sub_dir,"\\*.*");
  j=findfirst(sub_dir,&ff,s_attrib);
  While(! j)
  { if (((int)ff.ff_attrib & FA_DIREC) AND ff.ff_name[0] != '.')
    { i=d->subdir_num++;
```



```

NULL) ? 0;1;
point=NULL;
point=strchr(xdircmd-env, 'L');
if (point !=NULL) screen-line=atoi(point+
1);
point=strchr(xdircmd-env, 'R');
if (point !=NULL)
{ env=point+1;
while((*(env+sortnum)!='\0') AND (*(
env+sortnum)!=' ')AND(sortnum<4))
{ ++sortnum;
sortBy[0]=1;
switch(* (point+sortnum))
{ case 'N': sortBy [ sortnum ] byName;
break;
case 'E': sortBy [ sortnum ] = byExt;
break;
case 'D': sortBy [ sortnum ] = byDate;
break;
case 'S': sortBy [ sortnum ] = bySize;
break;
}
}
}
}

/* * * * * *
/* 分析命令行参数 */
d2=1;
for (d1=1;d1<argc;d1++)
{ strcpy(str1,argv[d1]);
strupr(str1);
switch(str1[0])
{ case '+':
attrib_r=(strchr(str1, 'R'))?FA_RDONLY;0;
attrib_h=(strchr(str1, 'H'))?FA_HIDDEN;0;
attrib_s=(strchr(str1, 'S'))?FA_SYSTEM;0;
attrib_d=(strchr(str1, 'D'))?FA_DIREC;0;
attrib_a=(strchr(str1, 'A'))?FA_ARCH;0;
attrib_v=(strchr(str1, 'V'))?FA_LABEL;0;
break;
case '-':
attrib_r=(strchr(str1, 'R'))?0;FA_RDONLY;
attrib_h=(strchr(str1, 'H'))?0;FA_HIDDEN;
attrib_s=(strchr(str1, 'S'))?0;FA_SYSTEM;
attrib_d=(strchr(str1, 'D'))?0;FA_DIREC;
attrib_a=(strchr(str1, 'A'))?0;FA_ARCH;
attrib_v=(strchr(str1, 'V'))?0;FA_LABEL;
break;
case '/':
switch (str1[1])
{ case 'D': flag_dir_tree=1; break;
case 'R':
sortnum=0;
while((str1[2+sortnum]! = '\0') AND (sortnum
<4))
{ ++sortnum;

```

```

sortBy[0]=1;
switch(str1[1+sortnum])
{ case 'N':sortBy[sortnum]=byName; break;
case 'E': sortBy[sortnum]=byExt; break;
case 'D':sortBy[sortnum]=byDate; break;
case 'S':sortBy[sortnum]=bySize; break;
}
}
break;
case 'S':flag_subdir=1; break;
case 'P': flag_pause=0; break;
case 'W':flag_wide=1; break;
case '?':flag_help=1; break;
default:
if (str1[1]>= '0' AND str1[1]<= '9')
screen-line=atoi(argv[d1]+1);
break;
}
break;
default:
if (d2==1)
{ strcpy(path,argv[d1]);
d2=(strchr(path, '*'))? 0;1;
d4=(strchr(path, '?'))? 0;1;
if (d3==1 AND d4==1)
{ d7=findfirst(path,&find_file_message,FA_
DIREC);
if ((d7==0) AND (find_file_message.ff_
attrib&FA_DIREC)! =0)
{ lenth=strlen(path);
if (path[lenth-1]! = '\\') strcat (path, "\\
");
}
}
d7=fnsplit(path,drive,dir,file,ext);
if (d7 & DRIVE) strcpy(path,drive);
else getcwd(path,81);
if (d7 & DIRECTORY) strcat(path,dir);
if (d7 & FILENAME) strcpy(search_name,
file);
else strcpy(search_name, " * ");
if (d7 & EXTENSION) strcat (search.name,
ext);
else if ((d7&FILENAME) == 0) strcat
(search_name, " * . * ");
}
else display_help_message();
++d2;
}
}
search_attrib=attrib_r|attrib_h|attrib_s|attrib_d|at_
trib_a|attrib_v;
if (argc==1 OR path[0]== ' ')
{ getcwd(path,81);
strcpy(search_name, " * . * ");
}
}

```



```

d7=tmp1->time-tmp2->time;
if (d6>0) cmp_value[3]=1;
else if (d6<0) cmp_value[3]=-1;
if (d6==0)
{ if (d7>0) cmp_value[3]=1;
  else if (d7<0) cmp_value[3]=-1;
  else cmp_value[3]=0;
}
d6=tmp1->size-tmp2->size;
if(d6>0) cmp_value[4]=1;
else if(d6<0) cmp_value[4]=-1;
else cmp_value[4]=0;
j=1;
if (sortBy[0]! =noSort)
{ d5=cmp_value[sortBy[j]];
  while ((d5==0) AND (j<sortnum))
  d5=cmp_value[sortBy[+j]];
  d5=(d5==0) ? -1:d5;
}
if (d5>0) tmp=tmp2;
}
while((d5>0) AND (tmp->next_pr !=
NULL));
tmp1->next_pr=tmp->next_pr;
tmp->next_pr=tmp1;
}
}
d1=findnext(&fine_fine_message);
/* * * * * * * * * * * * * * * * */
/* 显示得到的目录项 */
tmp=head;
while(tmp->next_pr !=NULL)
{ tmp=tmp->next_pr;
  d3=tmp->attrib;
  d5=tmp->date;
  d6=tmp->time;
  if(tmp==head->next_pr)
{ wait();
  wait();
  printf("Dir\t%s",b[d4]->father_dir);
  wait();
  d2=0;
}
++find_files_num;
find_files_size +=tmp->size;
if(flag_wide)
{ if (d2 % 5==0) wait();
  if (d3 & FA_LABEL)
  { wait();
    printf("DISK LABEL is\t%9s%5s",tmp->name,
    tmp->ext);
  }
}

```

```

else
{ if (d3 & FA_DIREC) printf("[");
  else printf(" ");
  printf("%-8s%-4s", tmp->name, tmp->
  ext);
  if (d3 & FA_DIREC) printf("]");
  else printf(" ");
  ++d2;
}
}
else
{ wait();
  printf("%-8s%-4s",tmp->name,tmp->ext);
  file_attr[0]=(d3 & FA_RDONLY)? 'R':',';
  file_attr[1]=(d3 & FA_HIDDEN)? 'H':',';
  file_attr[2]=(d3 & FA_SYSTEM)? 'S':',';
  file_attr[3]=(d3 & FA_ARCH)? 'A':',';
  printf("%5%",file_attr);
  if (d3 & FA_DIREC) printf("< DIREC> ");
  else if (d3 & FA_LABEL) printf("< LABEL> ");
  else printf("%10ld",tmp->size);
  d7=(d5>>5 & 0x0f)-1;
  printf(" %02u %3s %04u",d5 & 0x1f,month[d7],
  (d5>>9 & 0x7f)+1980);
  printf(" %02u:%02u:%02u",d6>>11 & 0x1f,d6>
  >5 & 0x3f,(d6 & 0x1f) * 2);
}
}
tmpfree=head;
tmp=head->next_pr;
while(tmpfree !=NULL)
{ free(tmpfree);
  tmpfree=tmp;
  if(tmpfree !=NULL) tmp=tmpfree->next_pr;
}
++d4;
}
while(flag_subdir==1 AND d4 <= subdir_count);
wait();
wait();
printf("DIR\t%s",b[0]->father_dir);
if (flag_subdir==1) printf("\t(INCLUDE ALL SUB-
DIR)");
wait();
wait();
if(find_files_num==0)
printf("N0 files that meet the condition:%14s",search-
name);
else
printf("Include %u files,%lu bytes",find_files_num,
find_files_size);
return(0);
}
}

```

C 语言汉字显示技术

西安公路学院 387 信箱(710064) 颜 飞

C 语言的 printf() 函数具备汉字输出功能,但不能实现定位和色彩变换。本文提出几种解决办法,给出实证程序。这些程序通过 Microsoft C 编译,能在多数汉字操作系统下运行。

1. 不使用汉字操作系统

汉字显示实质是将字符点阵映射到屏幕上,因此可以根据汉字字模画出汉字。在点阵字库中,每字节代表 8 个点,汉字在字库中占用的字节数为其点数除以 8。

运行在 MS-DOS 下的汉字操作系统大多采用与 CCBIOS 兼容的双字节汉字代码,第一字节是 0Ah 加汉字区码,第二字节是 A0h 加汉字位码。汉字在字库中按区存放,若某汉字其第一字节为 h1,第二字节为 h2,区码为 q,位码为 w,在字库中占据 x 个字节,到文件开始处的偏移量为 s 字节,则:

$$q = h1 - A0h;$$

$$w = h2 - A0h;$$

$$s = [(q-1) * 94 + w - 1] * x;$$

根据 s 的值可以将该汉字的字模从字库中调出。程序一给出了 24 点阵汉字的实例,它清屏后显示“华夏”两字。函数 write() 拆分字节,若某一位为“1”则在相应位置显示一点。改进程序一,能实现字型变换,放大、倾斜等许多显示效果。

为提高显示速度,可将大量汉字的字模从字库中一次调入内存,程序二可将文本文件中汉字的字模数据从 24 点阵字库中(CLIB,588816 字节)提取出来,生成标准 C 语言数组定义文件。程序二需要两个参数:汉字文本文件名和数组文件名。程序一中的数组便是由程序二生成的。

2. 使用汉字操作系统

2.1 使用 ANSI.SYS 逸出序列

MS-DOS 提供的设备驱动程序 ANSI.SYS 支持一套逸出码协议,输出到显示器的逸出码序列将由 ANSI.SYS 处理。常用 ANSI.SYS 逸出码协议列于表一。表中“ESC”为逸出标志,其 ASCII 码值为 27。

程序三是使用 ANSI.SYS 逸出协议的例证。它清屏后在第一行第一列反相显示一行汉字。因为 ANSI.SYS 是一个被广泛支持的终端驱动协议,所以这种方法具有良好的移植性。有的汉字操作系统对 ANSI.SYS 进行了修改,请使用汉字操作系统原配的 ANSI.SYS 程序。

2.2 调用 CCBIOS 中断

调用 CCBIOS 的 10h 中断的 09h 功能可以实现汉字的各种显示效果。该中断要求:

$$AH = 09h;$$

AL = 要显示的字符;

BH = 显示页(在汉字状态下为 00h);

BL = 字符显示属性;

CX = 欲写字符次数(在汉字状态下为 01h);

寄存器 BL 的构成如表二;前景和背景的取值见表三。

仅利用 10h 中断的 09h 功能是不够的,还必须调用 10h 中断的 02h 功能设置光标位置。此中断要求:

AH = 02h;

DH = 行;

DL = 列;

BH = 显示页(在汉字状态下为 00h);

程序四调用 10h 中断显示汉字,其中的 ctext() 函数用于显示汉字字符串。

表一:

逸出序列	简称	功能
ESC[pl;pcH ESC[p;pcf	CUP HVP	将光标移动到由参数指定的位置,如果没有规定参数,光标移动到原点。pl,pc 分别为行、列号
ESC[pnA	CUU	光标向上移动 p _n 行,列数不变,如果光标已在最上面一行,则忽视 CUU 序列。
ESC[pnB	CUD	光标向下移动 p _n 行,列数不变,如果光标已在最下面一行,则忽视 CUD 序列。
ESC[pnC	CUF	光标向前移动 p _n 列,行数不变。如果光标已在最右列,则忽视 CUF 序列。
ESC[pnD	CUB	光标向后移动 p _n 列,行数不变。如果光标已在最左列,则忽视 CUB 序列。
ESC[s	SCP	保存当前光标位置。此位置能被 RCP 恢复。
ESC[u	RCP	恢复光标位置为被 SCP 序列保存的值。
ESC[J	ED	清除屏幕,并将光标放回原点。
ESC[K	EL	删除从光标位置到本行末尾内容
ESC [Ps;...;Psm	SGR	通过指定下列数字参数,调用图形功能。这些功能一直保持到下一个 SGR 序列的出现。 0 还原 前景: 背景: 1 醒目 30 黑色 40 黑色 2 暗淡 31 红色 41 红色 3 斜体 32 绿色 42 绿色 4 底线 33 黄色 43 黄色 5 闪耀 34 蓝色 44 蓝色 6 高速闪耀 35 绛红 45 绛红 7 反向 36 青色 46 青色 8 隐藏 37 白色 47 白色

表二:

7	6	5	4	3	2	1	0
闪烁	背景色 八种			加亮	前景色 八种		

表三:

数值	0	1	2	3	4	5	6	7
颜色	黑 色	兰 色	绿 色	青 色	红 色	绛 红	棕 色	灰 色

```

/* 程序一: 显示 24 点阵汉字 */
#include <graph. h>
char ch [ ] = {
0, 0, 0, 0, 32, 128, 0, 64, 128, 0, 192, 128, 1, 128, 128, 3, 0,
128, 15, 252, 128, 255, 248, 128, 116, 0, 128, 64, 0, 128, 0, 64,
128, 0, 71, 255, 0, 131, 254, 255, 242, 128, 127, 248, 128, 66,
24, 128, 6, 24, 128, 12, 24, 128, 28, 24, 128, 8, 24, 128, 8, 25,
128, 1, 249, 128, 0, 16, 128, 0, 0, 0, 0, 0, 0, 0, 1, 32, 0, 17, 32,
0, 33, 32, 0, 66, 47, 252, 194, 39, 253, 130, 37, 75, 4, 37, 79,
132, 37, 77, 76, 61, 77, 40, 53, 73, 24, 37, 73, 48, 37, 73, 120,
37, 73, 200, 37, 73, 140, 47, 253, 12, 47, 252, 12, 36, 0, 6, 100,
0, 6, 96, 0, 6, 32, 0, 4, 0, 0, 4, 0, 0, 4};
void write(int ih, int x, int y, int color)
/* 显示 24 点阵汉字 */
{
char * p;
int yt=y;
register int i;
_setcolor(color);
for(p=&ch[ih]; p-&ch[ih]<72;){
for(i=0; i<3; i++){
if(* p&0x0080)-setpixel(x,y); y++;
if(* p&0x0040)-setpixel(x,y); y++;
if(* p&0x0020)-setpixel(x,y); y++;
if(* p&0x0010)-setpixel(x,y); y++;
if(* p&0x0008)-setpixel(x,y); y++;
if(* p&0x0004)-setpixel(x,y); y++;
if(* p&0x0002)-setpixel(x,y); y++;
if(* p&0x0001)-setpixel(x,y); y++;
}
y=yt;
x++;
}
}
main()
{
_setvideomode(_MAXRESMODE);
write(0, 48, 48, 7);
write(72, 72, 48, 7);
}
/* 程序二: 产生 24 点阵字模 */
#include <stdio. h>
#define NULL 0
main(int argc, char * argv)
{
int qm, wm, i, j;
unsigned char ch[72], cch[5];
FILE * zk, * wj, * hz;
if(argc!=3){
puts("正确的调用格式: hzarray 汉字文件 数组文件");
}

```

```

exit(0);
}
if((zk=fopen("clib", "rb"))==NULL){
puts("不能打开汉字库\n");
exit(0);
}
if((wj=fopen(argv[1], "r"))==NULL){
puts("不能打开汉字文件\n");
exit(0);
}
if((hz=fopen(argv[2], "w"))==NULL){
puts("不能打开数组文件\n");
exit(0);
}
fputs("char ch[]={\n", hz);
while(feof(wj)==0){
if((qm=getc(wj))>0xa0){
qm=qm-0xa0;
wm=getc(wj)-0xa0;
}
else
continue;
fseek(zk, (long)((qm-1)*94+(wm-1)*72, SEEK-
SET);
fread(ch, sizeof(char), 72, zk);
if(feof(wj)==0){
for(i=0; i<72;){
for(j=0; j<18; j++){
sprintf(cch, "%3d", ch[i]);
fputs(cch, hz); i++;
}
putc('\n', hz);
}
}
fseek(hz, -3L, SEEK_CUR);
fputs("};\n", hz);
fclose(zk); fclose(wj); fclose(hz);
puts("数组文件已经生成\n");
}
/* 程序三: 调用 ansi. sys 改善汉字显示效果 */
#define EL "\033[J\n"
#define CUP "\033[%d;%dH"
#define SGP "\033[%d;%dm"
main()
{
printf(EL);
printf(CUP, 1, 1);
printf(SGP, 30, 47);
printf("调用 ansi. sys 改善汉字显示效果");
printf(SGP, 37, 40);
}
/* 程序四: 调用 CC-BIOS 实现汉字显示 */
#include <stdio. h>
#include <dos. h>
void goto_xy(int x, int y) /* 置光标位置 */

```

```

{
union REGS r;
r.h.ah=2;
r.h.dl=x;
r.h.dh=y;
r.h.bh=0;
int86(0x10,&r,&r);
}
void ctext(int x,int y,unsigned char *str,unsigned char a)
/* 显示汉字 */
{
union REGS r;
r.h.ah=0x09; r.h.bh=0x00;
r.h.bl=a; r.x.cx=0x01;
while(*str){
goto_xy(x++,y);
r.h.al=*str++;
int86(0x10,&r,&r);
}
}
main()
{
int i;
char str[60];
ctext(10,0,"程序四",(unsigned char)0x70);
for(i=1;i<16;i++){
sprintf(str,"%2d,%s",i,"调用 CC-BIOS 实现汉字显示");
ctext(0,i,str,(unsigned char)i);
}
}

```

灵活实用的打印程序

齐齐哈尔电业局单身宿舍 407 室
(161005)郝显峰

调试程序时,难免要经常打印源程序清单;一个应用系统比较成熟时,还要打印程序清单做为资料保存。然而无论是基于操作系统下的假脱机打印还是 TYPE 命令,或者各种工具软件如 WS、CCED、PC Tools 中的打印功能,都不尽如人意:一般只能单列打印;只能输入一个打印一个;对其它路径中的文件缺乏处理能力;缺少一些重要信息,如文件名称,页码,行号等,不便于调试程序和保存。根据这种情况,本人利用 dBASE III 编制了一个打印程序,较好地解决了这些问题。本程序具有下列特点:

1. 能打印任何路径中的文本文件,并且能自动打印完所有您想打印的文件。
2. 可设定每页打印多少行,每行左边空白,是否加行号。
3. 可设定使用宽纸还是窄纸,单列打印还是双列

打印。

4. 可设定打印份数。

关于程序的几点说明:

1. 由于 dBASE III 缺乏直接处理文本文件的能力,故用 APPE FROM &M-NAME SDF 将文本文件的每一行转化为数据库中的一条记录。这里共建有两个数据库:FILENAME.DBF 和 TEXTBASE.DBF,结构比较简单,均只含有一个字符型字段。前者含有的字段名为 NAME1,长度为 25,用于存放需打印文件名称。后者含有的字段名为 ZL,长度为 200,用于存储具体要打印的文件。

2. 各程序的具体功能。

PRN-FILE.PRG 用于设置打印参数。其中产生的变量 N1 用来调整打印宽度,C1 用来确定是单列打印还是双列打印,M-YN 确定是否要行号,PF 决定打印份数。

过程文件 PRN-PROC 中包含下列程序:

TEXTCHK1.PRG 用来清除需打印程序名输入中混入的空格以及删除不存在的文件。TEXTCHK2.PRG 用于调整打印宽度,即当文件中一行超过一定限制(>N1)时,自动地将一行变为几行。为便于阅读,在超长行末尾加上分号,并将其长出部分移至下一行并在头部空出两格。PRIN-1.PRG 和 PRIN-2.PRG 用来完成具体的打印任务。

3. 由于各种打印机控制命令很不一致,计算机运行环境各不相同,为保证通用性,这里没有过多加入打印机控制命令。

实验证明,此程序能良好地运行于 dBASE 及 FoxBASE+ 各种版本,并支持多种打印机,如 M1724, LQ1600K, AR3240 等。

一、PRN-FILE.PRG

```

1 SET TALK OFF
2 SET SAFE OFF
3 CLEA
4 SET PROC TO PRN-PROC
5 SELE 1
6 USE FILENAME
7 ZAP
8 @7,40 SAY '提示:直接打回车结束'
9 DO WHIL.T.
10 APPE BLAN
11 @3,25 SAY '请输入需打印文件名:' GET NAME1
12 READ
13 IF LEN(TRIM(NAME1))=0
14 EXIT
15 ENDI
16 ENDD
17 DO TEXTCHK1
18 SELE 2
19 USE TEXTBASE
20 STOR 60 TO P1,PP
21 CLEA
22 @3,20 SAY '每页打印多少行? GET PP PICT' 99'

```



```

RANG 1,P1
23 READ
24 PB=5
25 @4,20 SAY '每行左边空白?' GET PB PICT'9'
26 READ
27 @5,20 SAY '1...窄纸 2...宽纸'
28 SET CONS OFF
29 C1=' '
30 DO WHIL .NOT.C1$'12'
31   WAIT '' TO C1
32 ENDD
33 IF C1='1'
34   N1=110-PB
35 ELSE
36   N1=180-PB
37 ENDI
38 N11=N1+PB
39 @6,20 SAY '1...单列 2...双列'
40 C1=' '
41 DO WHIL.NOT.C1$'12'
42   WAIT '' TO C1
43 ENDD
44 IF C1='1'
45   PP0=PP
46 ELSE
47   PP0=PP*2
48   N1=INT(N1/2)-7
49 ENDI
50 @7,20 SAY '是否要行号? (Y/N)'
51 M_YN=' '
52 DO WHIL .NOT. M_YN$'YNyn'
53   WAIT '' TO M_YN
54 ENDD
55 M_YN=UPPE(M_YN)
56 SET CONS ON
57 PF=01
58 @8,20 SAY '打印多少份?' GET PF
59 READ
60 CLEA
61 @4,20 SAY '请联打印机,然后按任一键打印.....'
62 WAIT ''
63 CLEA
64 @4,20 SAY '正在打印,请耐心等待.....'
65 DATE1=DOE(DATE())
66 M_DATE = SUBS (DATE1, 7, 2) + '年' + SUBS
(DATE1,1,2) + '月' + SUBS(DATE1,4,2) + '日'
67 SET DEVI TO PRIN
68 SET PRIN ON
69 N2=0
70 DO WHIL N2<PF
71   SELE 1
72 GO TOP
73 DO WHIL .NOT. EOF()
74   M_NAME=TRIM(NAME1)
75   SELE 2
76   USE TEXTBASE

```

```

77   ZAP
78 APPE FROM &M_NAME SDF
79 DO TEXTCHK2
80 COUN ALL TO N3
81 IF N3/PP0=INT(N3/PP0)
82   N4=N3/PP0
83 ELSE
84   N4=INT(N3/PP0)+1
85 ENDI
86 GO TOP
87 EJEC
88 N5=1
89 M_EOF=. T.
90 DO WHIL .NOT. EOF() .AND. M_EOF
91   @2,N11/2-10 SAY '文件名:' + M_NAME
92   @2,N11-19 SAY '打印日期:' + M_DATE
93   @3,N11-12 SAY 'Page' + STR(N5,3) + ' of' + STR
(N4,3)
94   N6=0
95   IF C1='1'
96     DO PRIN-1
97     ELSE
98       DO PRIN-2
99     ENDI
100 @64,N11/2-10 SAY'——第'+STR(N5,3)+'页 共'
+STR(N4,3)+'页——'
101 @65,0 SAY ' '
102 @66,0 SAY ' '
103 N5=N5+1
104 EJEC
105 ENDD
106 SELE 1
107 SKIP
108 ENDD
109 N2=N2+1
110 ENDD
111 EJEC
112 SET PRIN OFF
113 SET DEVI TO SCRE
114 SELE 1
115 ZAP
116 SELE 2
117 ZAP
118 SET TALK ON
119 SET SAFE ON
120 CLOS DATA
121 CLOS PROC
122 CLEA
123 RETU
二、PRN-PROC.PRG
1 * TEXTCHK1
2 PROCEDURE TEXTCHK1
3 SELE 1
4 GO TOP
5 DO WHIL .NOT. EOF()
6 N10=1

```

```

7 M_NAME=SPAC(0)
8 DO WHILE N10<25
9 IF SUBS(NAME1,N10,1)#' '
10 M_NAME=M_NAME+SUBS(NAME1,N10,1)
11 ENDI
12 N10=N10+1
13 ENDD
14 REPL NAME1 WITH M_NAME
15 M_NAME=""+M_NAME+"'"
16 IF .NOT. FILE(&M_NAME)
17 DELE
18 ENDI
19 SKIP
20 ENDD
21 PACK
22 RETU
23 * TEXTCHK2
24 PROCEDURE TEXTCHK2
25 GO TOP
26 DO WHILE .NOT. EOF()
27 IF LEN(TRIM(ZL))>N1
28 C9=TRIM(ZL)
29 REPL ZL WITH SUBS(C9,1,N1)+';'
30 INSE BLAN
31 REPL ZL WITH ' '+SUBS(C9,N1+1,200-N1)
32 ELSE
33 SKIP
34 ENDI
35 ENDD
36 RETU
37 * PRIN_1
38 PROCEDURE PRIN_1
39 DO WHILE .NOT. _EOF() .AND. N6<PP
40 IF M_YN='Y'
41 @PROW()+1,PB SAY STR(RECN(),4)+''+
TRIM(ZL)
42 ELSE
43 @PROW()+1,PB SAY TRIM(ZL)
44 ENDI
45 SKIP
46 N6=N6+1
47 ENDD
48 RETU
49 * PRIN_2
50 PROCEDURE PRIN_2
51 IF N3-RECN()<PP
52 DO PRIN_1
53 ELSE
54 DO WHILE N6<PP
55 IF M_YN='Y'
56 @PROW()+1,PB SAY STR(RECN(),4)+''+
TRIM(ZL)
57 ELSE
58 @PROW()+1,PB SAY TRIM(ZL)
59 ENDI
60 SKIP PP
61 IF EOF()
62 SKIP 1-PP
63 N6=N6+1
64 DO PRIN_1
65 M_EOF=.F.
66 RETU
67 ELSE
68 IF M_YN='Y'
69 @PROW(),PB+INT((N11-PB)/2)+2 SAY
STR(RECN(),4)+''+TRIM(ZL)
70 ELSE
71 @PROW(),PB+INT((N11-PB)/2)+2 SAY
TRIM(ZL)
72 ENDI
73 ENDI
74 SKIP 1-PP
75 N6=N6+1
76 ENDD
77 SKIP PP
78 ENDI
79 RETU

```

巧用数组和 GATHER FROM 命令

安庆石化总厂教育中心(246001) 何钧军

FoxBASE+ 不仅与 dBASE III plus 完全兼容,而且增加了许多功能,巧用 dBASE 不具备的数组和 GATHER FROM 命令,将对您的编程有所帮助。

1. FoxBASE+ 中使用数组,只须用 DIMENSION 命令定义即可,如 DIMENSION A(10),B(3,4)。

2. GATHER FROM 命令的作用,是将数组内容

取代数据库当前指针所指记录的字段内容,其格式为: GATHER FROM <内存变量数组> [FIELDS <字段名表>]。只要有足够的数组元素,就按次序替换当前记录所选字段的内容,余下的元素被忽略。若数组元素少于所选字段个数,则仅依次替换所选字段内容。

程序 1 是本人在编写学生成绩统计程序中的实

例,该程序根据初一年级各班数据库(C101.DBF——C106.DBF),统计出各班各学科的平均分,并将其合并到一个数据库 C1AV.DBF 中。

```

程序 1:
set talk off
set safe off
use c101
copy to clav fiel xm,zz,yw,sx,yy,ls,dl,sw,zf
use clav
zap
dimension a(9)
input '请输入该年級的班级总数:' to d1
d=1
do while d<=d1
  stor' c10' to y
  y=y+str(d,1)
  use &y
  repl all zf with yw+sx+yy+ls+dl+zz+sw
  aver zz,yw,sx,yy,ls,dl,sw,zf to a(2),a(3),a(4),a(5),a
  (6),a(7),a(8),a(9)
  stor y to a(1)
  use clav
  appe blank

```

```

gath from a(9)
d=d+1
enddo
use
close data
retu

```

C101.DBF 数据库结构:

display structure

数据库结构: C:\C1\C101\DBF
 数据记录数: 40
 最新更改日期 :12/21/92

字段	字段名	类型	宽度	小数
1	XM	字符	6	
2	ZZ	数值	5	1
3	YW	数值	5	1
4	SX	数值	5	1
5	YY	数值	5	1
6	LS	数值	5	1
7	DL	数值	5	1
8	SW	数值	5	1
9	ZF	数值	5	1

CEC—I 中华学习机系统子程序

湖南株洲硬质合金厂财务室(412000)王志超

灵活地运用系统子程序,不但能节约大量的时间和脑力,还能开发出许多新的功能,把程序编得更加简练。但许多资料对这些子程序都只有零星的介绍,初学者往往不知道有哪些子程序可供调用和如何调用,使得它们的功能得不到充分的利用和发挥。笔者目前收集整理了一些常用的系统子程序,现奉献给大家,希望能给各位带来方便。

所收集的系統子程序基本上分成三类:一类是DOS系统子程序;二是CEC—I BASIC系统子程序;三是监控系统子程序。以下的简表按它们的入口地址排列,括号里面是10进制地址。需要说明一点:有的子程序可在BASIC程序中以CALL语句调用,有的则只能在机器语言程序中调用。另外,CEC—I BASIC语言中,有个别入口地址与苹果机浮点BASIC语言不同的语句以及新增语句,也在简表中列出。

CEC—I 中华学习机系统子程序简表

说明:FAC)浮点累加器(\$9D~\$A3);ARG)次浮点累加器(\$A5~\$AB);TXIPTR)代码存储地址指示器(\$B8~\$B9);LINUM)行号指示器(\$50~\$51);KSW)输入向量(\$38,\$39);CSW)输出向量

(\$36,\$37);BAS)光标位置基值(\$28,\$29)

标号	首地址	功能
CHRGET	\$B1(177)	扫描键盘缓冲区
	\$3D0(976)	DOS 暖启动入口
	\$3D3(979)	DOS 冷启动入口
	\$3D6(982)	DOS 文件处理器 FM 入口
	\$3D9(985)	RWTS 入口
	\$3DC(988)	设置 DOS FM 输入参数
	\$3E3(995)	设置 RWTS 输入参数
	\$3F5(1013)	BASIC 语言 & 命令入口
	\$3F8(1016)	监控 CTRL-Y 命令入口
	\$3FB(1019)	不可屏蔽中断服务子程序入口
	\$9D84(40324)	DOS 冷启动(由 \$3D3 跳入)
	\$9DBF(40383)	DOS 暖启动(由 \$3D0 跳入)
	\$9E25(40485)	DOS 设置第 3 页指针
	\$9E81(40577)	DOS 键入截断处理
	\$9EBD(40589)	DOS 输出截断处理

	\$ 9FCD(40909)	DOS 命令扫描和剖析		\$ D0EC(53484)	MUSIC 入口
	\$ A180(41343)	DOS 命令执行入口		\$ D10C(53516)	LG 入口,进入 LOGO
	\$ A316(41750)	释放全部 DOS 文件缓冲区。可 CALL 41750 调用		\$ D26C(53868)	SAVE 入口
	\$ A6D5(42709)	DOS 错误处理		\$ D27E(53886)	=全清屏(若是中文状态还清汉字映射区)。可以 CALL 53886 调用
	\$ A7D4(42964)	重建 DOS 文件缓冲区		\$ D2A3(53923)	GAME 入口
	\$ A851(43089)	更换 DOS 输入/输出向量		\$ D2B4(53940)	=TEXT,进入西文状态
	\$ AAFD(43773)	DOS 文件处理器 FM 进入点(由 \$ 3D6 跳入)		\$ D350(54096)	小汇编入口,可以 CALL 54096 进入
	\$ AB06(43782)	DOS 文件处理器 FM 主程序		\$ D412(54290)	显示 BASIC 出错信息
	\$ AFF7(45047)	读取 VTOC,资料缓冲区在 \$ B3BB~\$ B4BA		\$ D559(54617)	将输入的 BASIC 命令代码化
	\$ AFFB(45051)	写入 VTOC,资料缓冲区在 \$ B3BB~\$ B4BA		\$ D61A(54810)	找出指定行号的地址(行号取自 \$ 50,\$ 51)
	\$ B800(47104)	DOS 写数据预处理(编码)		\$ D6A5(54949)	进入 BASIC 状态
	\$ B82A(47146)	DOS 写数据至磁盘		\$ D828(55336)	BASIC 命令执行入口
	\$ B8C2(47298)	DOS 数据解码		\$ D941(56617)	将 TXTPTR 指向 \$ 50,\$ 51 规定的语句去执行
RWTS	\$ BD00(48384)	磁盘驱动(由 3D9 跳入)		\$ DA0C(55820)	将 TXTPTR 所指的 10 进制数化为 16 进制数送 \$ 50,\$ 51
	\$ C1A0(49568)	自选格式打印。打印卡须插 1 号槽。调用格式 CALL 49568:[串变量;]变量或表达式;格式符(整型为 In,实型为 Fn.m,指数为 En.n 为数字个数,m 为小数位数)	STROUT	\$ DB3A(56112)	将 Y(高),A(低)所指的字符串输出。字符串须由 0 或单引号结束
	\$ C300(49920)	初次进入中文状态的处理		\$ DD67(56679)	读输入参数(常量,变量,表达式)送入 FAC
	\$ C303(49923)	中文状态的字符输入入口		\$ DD7B(56699)	取 TXTPTR 所指的代码,并将表达式的结果送 FAC
GB.CSWA	\$ C322(49954)	中文状态的字符或异形码汉字输出		\$ DEBE(57022)	检查读入的是否逗号,否则显示错误信息并返回 BASIC
CSWA	\$ C32B(49966)	中文状态的字符或机内码汉字输出		\$ DEC9(57033)	BASIC 错误处理
ZT.XS1	\$ C36E(50030)	在第 11 行显示中文状态字		\$ DFE3(57315)	读参数变量,并将变量在内存中的首址存入 A(低),Y(高)
ZT.XS2	\$ C377(50039)	在第 11 行显示键入提示字符		\$ E07D(57469)	判断 A 中键码是否 < \$ 41,是则 C=0,否则 C=1
ZT.XS3	\$ C380(50048)	显示汉字提示,并自动插入序号			
USR.DCOD	\$ C389(50057)	将异形国标码转换为机内码(入,出口参数在 A)	AYZNT	\$ E10C(57612)	将 FAC 中的数转换成 2 字节整数存 \$ A0(高),\$ A1(低)
USR.ECOD	\$ C392(50066)	将机内码转换为异形国标码(入,出口参数在 A)	GIVAYF	\$ E2F2(58098)	将 A(高),Y(低)中的带符号整数转换成浮点数存 FAC 或 USR 函数指定的变量中
BACKSP	\$ C39B(50075)	中文状态下删除光标处字符,并将光标退一格			
SLECTA1	\$ C3A4(50084)	选择辅存 1			
SLECTA2	\$ C3AB(50091)	选择辅存 2	SNGFLT	\$ E301(58113)	将 Y 中的不带符号整数转换成浮点数存 FAC
SLECTM1	\$ C3B2(50098)	选择主存 1			
SLECTM2	\$ C3B9(50105)	选择主存 2	CONINT	\$ E6FB(59131)	将 FAC 中 < 256 的浮点数转换成 1 字节整数送 X 和 \$ A1
	\$ C800(51200)	开机时磁盘假读			
	\$ D0CE(53454)	PLAY 入口			

	\$ E746(59206)	读入二个以逗号分隔的算术表达式(表达式之值均须<256),值1送\$50,值2送\$A1	SIGN	\$ EB82(60290)	根据 FAC 之值设定 A。若 FAC=0,A=0;若 FAC>0,A=1;若 FAC<0,A=\$FF
	\$ E74C(59212)	将 CALL 地址后以逗号分隔的参数存入 X。参数均<256,每个参数调用本程序一次	FLOAT	\$ EB93(60307)	将 A 中的带符号整数转换成浮点数存 FAC
	\$ E752(59218)	将 FAC 中的浮点数转换成 2 字节正数(<65536)置入 \$50(低),\$51(高)	FCOMP	\$ EBB2(60338)	将 Y(高),A(低)为首址的浮点数与 FAC 比较并设定 A。若(Y,A)<FAC,A=1;若(Y,A)=FAC,A=0;若(Y,A)>FAC,A=\$FF
	\$ E7A0(59296)	$1/2 + FAC \rightarrow FAC$			
FSUB	\$ E7A7(59303)	将 Y(高)A(低)为首址的 5 字节浮点数送 ARG 后转 \$E7AA	OUT	\$ ED24(60708)	将 A(高),X(低)中的 4 位 16 进制数化为 10 进制输出
FSUBT	\$ E7AA(59306)	$ARG - FAC \rightarrow FAC$ (减法)	FOUT	\$ ED34(60708)	将 FAC 中的数转换成 10 进制数字串,存入首址为 \$100 的区域,并将 Y(高),A(低)指向 \$100,可调用 \$DB3A 输出
FADD	\$ E7BE(59326)	将 Y(高)A(低)为首址的 5 字节浮点数送 ARG 后转 \$E7C1		\$ F128(61736)	BASIC 系统初始化
FADDT	\$ E7C1(59329)	$ARG + FAC \rightarrow FAC$ (加法)		\$ F26D(62061)	=TRACE(行号跟踪)
FMULT	\$ E97F(59775)	将 Y(高)A(低)为首址的 5 字节浮点数送 ARG 后转 \$E982		\$ F26F(62063)	=NOTRACE(取消行号跟踪)
FMULTT	\$ E982(59778)	$ARG * FAC \rightarrow FAC$ (乘法)		\$ F3F2(62450)	高分率清屏,预置 \$E6 为 \$20(1 页)或 \$40(2 页)
CONUPK	\$ E9E3(59875)	将 ARG 中的浮点数送 Y(高),A(低)为首址的 5 字节区域		\$ F3F6(62454)	将屏幕置成 HCOLOR 设定色,须先执行 HPLOT
	\$ EA39(59961)	$FAC * 10 \rightarrow FAC$		\$ F411(62481)	计算高分率基地址存 \$26(低),\$27(高),预置 A=行号(0~191),\$E6=\$20(1 页)或=\$40(2 页)调用后 X,Y,A 改变(原值存 \$E0,\$E1,\$E2)
	\$ EA55(59989)	$FAC/10 \rightarrow FAC$ (只送正值)			
FDIV	\$ EA66(60006)	将 Y(高)A 低为首址的 5 字节浮点数送 ARG 后转 \$EA69		\$ F457(62551)	高分率绘点,预置 A=纵座标(0~191),X=横座标(0~279)低位,Y=高位,调用后 A,X,Y 改变
FDIVT	\$ EA69(60009)	$ARG/FAC \rightarrow FAC$ (除法)		\$ F53A(62778)	高分作图时,从当前座标绘线至预置座标,预置 Y=纵座标(0~191);A=横座标(0~279)低位,X=横座标高位,调用后 A,X,Y 改变。
MOVFM	\$ EAF9(60153)	将 Y(高)A(低)为首址的 5 字节浮点数送 FAC			
MOV2F	\$ EB1E(60190)	$FAC \rightarrow TEMP2$ (\$98~\$9C)		\$ F6F0(63216)	设置高分作图颜色,预置 X=颜色序号(0-7)\$E4=颜色代码,序号与代码对应关系:0-0,1-2A,2-52,3-7F,4-80,5-AA,6-D5,7-FF
MOV1F	\$ EB21(60193)	$FAC \rightarrow TEMP1$ (\$93~\$97)			
MOVML	\$ EB23(60195)	将 FAC 中的浮点数送 X 为首址的 0 页单元			
MOVMF	\$ EB2B(60203)	将 FAC 中的浮点数送 Y(高),A(低)为首址的 5 字节区域			
MOVFA	\$ EB53(60243)	$ARG \rightarrow FAC$			
MOVAF	\$ EB63(60259)	$FAC \rightarrow ARC$			

PLOT	\$ F800(63488)	在低分 1 页 A 行 Y 列绘点, 预置 \$ 30=色码	PCADJ	\$ F953(63827)	计算下一条指令的操作地址存入 A(高), Y(低)
	\$ F80E(63502)	低分一页在前一点的旁边绘点, 预置 Y=纵座标(0~39)		\$ FA40(64064)	可屏蔽中断服务子程序
				\$ FA4C(64076)	BRK 中断服务子程序
HLINE	\$ F819(63513)	低分率状态在 A 行从 Y 列至 H2(\$ 2C)列画一条由 COLOR 置色的水平线, 调用后 A, Y 改变	RESET	\$ FA62(64098)	开机或复位时系统初始化
				\$ FAA6(64166)	BASIC 冷启动, 且保留内存程序, 可以 CALL 64166 调用
VLINE	\$ F828(63528)	低分率状态在 Y 列从 A 行至 V2(\$ 2D)行画一条由 COLOR 置色的垂直线, 调用后 A 改变	PREAD	\$ FB1E(64286)	读游戏控制器第 X 号模拟量输入, 结果存 Y
			INIT	\$ FB2F(64303)	置文本第 1 页并将 STATUS\$ 48 清 0
CLRSCR	\$ F832(63538)	低分率第一页清屏. 若是文本第一页, 则全屏形成反相@字符	GR	\$ FB40(64320)	置低分 1 页方式
			TABV	\$ FB5B(64347)	将行位置存入 CV(\$ 25)后转 VTAB 去计算文本页基地址, 并将光标移至该位置
CLRTOP	\$ F836(63542)	把低分率第一页屏幕上 40 行清屏, 若是文本第一页, 则上 20 行形成反相@		\$ FB6F(64367)	将 \$ 3F3 之值与 A5 进行异或运算后存 \$ 3F4
CLRSC2	\$ F838(63544)	把低分屏幕 0-Y 行清屏, 调用后 A, Y 改变	VIDWAIT	\$ FB78(64376)	处理由 RDCHAR 返回的 A 中的字符
	\$ F83A(63546)	把低分屏幕 0-V2 行清屏, 调用后 A, Y 改变	ESCNEW	\$ FBA5(64421)	判断 A 中是否 ESC 键码(编辑键码)
	\$ F847(63559)	计算 A(低分率行号除 2 之值)所对应的内存基地址并存入 GBASL, H(\$ 26, \$ 27)		\$ FBC1(64449)	计算文本第一页基地址, 并存入 BASL, H
NXTCOL	\$ F85F(63583)	现色码+3 设定为低分新色码, 调用后 A 改变	BELL1	\$ FBD9(64473)	若 A = \$ 87, 响铃, 频率 1KHZ, 延续 0.1 秒, 调用后 A, Y 改变; 否则退回
SETCOL	\$ F846(63588)	根据 A 中右半字节的颜色码设定低分颜色, 调用后 A 改变		\$ FBE2(64482)	响铃. 频率 1KHZ, 延续 0.1 秒
SCRN	\$ F871(63601)	将 A(行), Y(列)点的低分色码取至 A 的右半字节	BELL2	\$ FBE4(64484)	= 功能同上, 但延续由 Y 定, 调用后 A, Y 改变
			ADVANCE	\$ FBF4(64500)	=>, 可以 CALL -1036 调用
INSTDSP	\$ F8D0(63696)	操作码分析, 并依次输出指针(\$ 3A, \$ 3B)所指的地址, -, 3 个空格, 操作码, 指令助记符	VIDOUT	\$ FBFD(64509)	将 A 中可显示字符, 送入光标所对应的显示缓冲区
PRNTYX	\$ F940(63808)	将 Y(高位), X(低位)用 4 位 16 进制数输出, 调用后 A 改变	BS	\$ FC10(64528)	=←, 可以 CALL -1008 调用
PRN-	\$ F941(63809)	将 A(高位), X(低位)用 4 位 16 进制数输出, 调用后 A 改变	UP	\$ FC1A(64538)	=↑, 可以 CALL -998 调用
TAX			VTAB	\$ FC22(64546)	在 \$ 25 单元中取行位置 CV 后转 \$ FBC1
PRNTX	\$ F944(63812)	将 X 用 2 位 16 进制数输出, 调用后 A 改变	VTABZ	\$ FC24(64548)	预置 A = 光标行位置 CV 后转 \$ FBC1
PRBLNK	\$ F948(63816)	输出 3 个空格, 调用后 A, X 改变		\$ FC2C(64556)	ESC 编辑入口
PRBL2	\$ F94A(63818)	输出 X 个空格, 预置 A = \$ A0, X = 空格数	CLREOP	\$ FC42(64578)	西文状态从光标处清屏至页末, 可以 CALL -958 调用
			CLREOP1	\$ FC46(64582)	西文状态从 A 行 Y 列清屏至页末

HOME	\$ FC58(64600)	西文状态全屏,光标回左上角. 可以 CALL - 936 调用	VFY	\$ FE36(65078)	监控 V 命令入口,将两个指定地址的数据逐个比较,若有不等则显示
CR	\$ FC62(64610)	送 \$ 8D(回车)至显示器,调用后 A, Y 改变	LIST	\$ FE5E(65118)	监控 L 命令入口. 显示 20 条指定地址的指令反汇编
LF	\$ FC66(64614)	= ↓, 换行, 可以 CALL - 922 调用		\$ FE75(65141)	取操作指令地址
SCROLL	\$ FC70(64624)	上滚一行, 可以 CALL - 912 调用	SETINV	\$ FE80(65152)	监控 I 命令入口,置反相显示 (INVERSE)
SCRL3	\$ FC95(64661)	西文状态下从光标处清屏至下一行末,光标移至下一行首, 可以 CALL - 875 调用	SETNORM	\$ FE84(65156)	监控 N 命令入口,置正常显示 (NORMAL)
CLREOL	\$ FC9C(64668)	ESC - Em 从光标处清至行末, 可 CALL - 868 调用		\$ FE8B(65163)	输入设备初始化, 预置 A = 设备槽口号
CLREOLZ	\$ FC9E(64670)	从 BASL, BASH + Y 处清屏至行末	XBASIC	\$ FEB0(65200)	输出设备初始化, 预置 A = 设备槽口号
WAIT	\$ FCA8(64680)	延时, 时值预置 A 中, 返回时 A = 0; X, Y 不变	BASCONT	\$ FEB3(65203)	监控 CRTL - B 命令入口, 返回 BASIC 并进行系统初始化
RDKEY	\$ FD0C(64780) \$ FD18(64792)	出闪烁光标后转 \$ FD18 转入 KSWL, KSWH 所指向的输入子程序	GO	\$ FEB6(65206)	监控 CRTL - C 命令入口, 返回 BASIC 不进行系统初始化
KEYIN	\$ FD1B(64795)	由键盘接收一个字符至 A	WRITE	\$ FECD(65229)	监控 G 命令入口, 转指定地址执行指令
RDCHAR	\$ FD35(64821)	调用 RDKEY 后再调用 ESCNEW, 判断 A 中是否编辑键字符, 是再调用 RDKEY 后转入 \$ FD2F; 否则返回	TRACE	\$ FED0(65232)	监控 W 命令入口, 将指定地址数据写至磁带
GETLNZ	\$ FD67(64871)	回车换行后转 GETLN	STEP2	\$ FED2(65234)	监控 T 命令入口, 从指定地址开始的程序每执行一条指令, 显示反汇编及各寄存器值
GETLN	\$ FD6A(64874)	显示提示符后转 GETLNI	READ	\$ FEFD(65277)	监控 S 命令入口, 单步执行指定地址的指令, 并显示反汇编及各寄存器之值
	\$ FD6F(64879)	由输入设备接收一行字 (< 256), 调用后 X 含字符个数	PRERR	\$ FF2D(65325)	监控 R 命令入口, 将磁带数据读至指定地址
	\$ FD8B(64907)	从光标清至行末, 并输出一个回车	RELL	\$ FF3A(65338)	送 ERR 至输出设备并响铃, 调用后 A 改变
CROUT	\$ FD8E(64910)	送回车 (\$ 8D) 至输出设备. 调用后 A 改变	RE-	\$ FF3F(65343)	送 CHR \$ (7) 至输出设备
PRBYTE	\$ FDDA(64986)	将 A 以 2 位 16 进制数输出. 调用后 A 改变	STORE		将 \$ 45 - \$ 48 内容送回 A, X, Y, P
PRHEX	\$ FDE3(64995)	将 A 的低 4 位以 1 位 16 进制数输出. 调用后 A 不变	SAVE	\$ FF4A(65354)	将 A, X, Y, P, S 保存到 \$ 45 - \$ 49, 并清除 10 进制工作方式
COUT	\$ FDED(65005)	将 A 中的字符送由 CSWL, CSWH 指定的输出设备, A, X, Y 不变	MONZ	\$ FF69(65385)	监控命令处理入口, 可以 CALL - 151 进入
COUT1	\$ FDF0(65008)	将 A 中字符输出到屏幕, 并将光标前移一格		\$ FF70(65392)	键盘扫描和翻译, 可以 CALL - 144 进入
MOVE	\$ FE2C(65068)	监控 M 命令入口, 将 \$ 3C \$, \$ 3D 至 \$ 3E, \$ 3F 中指定地址的数据送至 \$ 42, \$ 43 指定首地址单元, 调用前置 Y = 0	GETNUM	\$ FFA7(65447)	键缓冲区扫描并换码
			TOSUB	\$ FFBE(65470)	从 SUBTBL (\$ FFE3 监控命令入口地址表) 中, 取操作子程序入口地址

ProDOS 系统内部结构剖析(续)

廖 凯

三、标准文件的格式

在一个目录文件内,每个现有(有效)项目指向一个文件的关键块(第一块)。一个标准文件的关键块可能有几种类型的资料在其内。在那个文件的项目内,存储类型必须被用以确定关键块的内容,下面描述三种标准文件的结构:树苗、幼树和树型文件。

1. 产生一个树型文件

此部分说明在一个卷(含有 280 块的软盘)上生长一个树型文件。大容量卷在卷位图内会有许多块,但处理方法是相同的。

一个格式化后的空磁盘,其块分布情况如下:

块 0—1	: 引导程序
块 2—5	: 卷目录
块 6	: 卷位图
块 7—279	: 未用

如果你打开一个非目录型的新文件,立即给这文件分配一个数据块,在卷目录内放置一个登记项,并且它指向块 7 这个新的数据块,作为这文件的关键块,这关键块在下面用箭头指出,此卷现如下所示:

数据块 0		关键块
块 0—1	: 引导程序	
块 2—5	: 卷目录	
块 6	: 卷位图	
→块 7	: 数据块 0	
块 8—279	: 未用	

这是一个树苗型文件,它的关键块包含最大 512 字节的数据。如果你要写入比 512 字节多的数据给文件,那么这文件生长成为一个幼树型文件。一旦需要分配第二个数据块,就同时分配一个索引块,并且成为文件的关键块。这索引块可以指向最大 256 个数据块(两字节指针)。此时这卷的块分配如下:

索引块 0		关键块
数据块 0		
数据块 1		
块 0—1	: 引导程序	
块 2—5	: 卷目录	
块 6	: 卷位图	
块 7	: 数据块 0	
块 8	: 索引块 0	
块 9	: 数据块 1	
块 10—279	: 未用	

这幼树型文件可以容纳最多 256 个数据块:128K

数据,若此文件再增大,将再成长为一个树型文件。此时将分配一个主索引块,并成为文件的关键块:这主索引块可以指示最大 128 个索引块,而每个索引块可以指示最大 256 个数据块。索引块 0 成为主索引块指向的第一个索引块。另外还分配一个新的索引块并指向一个新的数据块,下面是此卷的新分配图;

主索引块		
索引块 0		
索引块 1		
数据块 0		
数据块 255		
数据块 256		
数据块 0		

块 0—1	: 引导程序
块 2—5	: 卷目录
块 6	: 卷位图
块 7	: 数据块 0
块 8	: 索引块 0
块 9—263	: 数据块 1—255
块 264	: 主索引块
块 265	: 索引块 1
块 266	: 数据块 256
块 267—279	: 未用

关键块

在数据块被写到这文件时,附加的数据块和索引块按需要被分配,一个树型文件最多有 129 个索引块(包含一个主索引块)和 32,768 个数据块,最大容量为 16,777,215 字节。最大文件的最后块的最后字节不能使用,因为 EOF 不能超过 16,777,216。

2. 树苗型文件

树苗型文件是一个包含不超过 512 数据字节($\$0 \leq EOF \leq \200)的标准文件。此文件做为一个块被存储在卷上,并且这数据块是文件的关键块,树苗型文件的结构如下:

关键指针 → 数据块 数据块 512 字节长
 $\$0 \leq EOF \leq \200

指向一个树苗型文件的项目的“存储类型”字段的值为 \$1。

3. 幼树型文件

幼树型文件是一个包含比 512 字节多而不大于 128K 字节($\$200 < EOF \leq \20000)的标准文件。一个幼树型文件由一个索引块和 1 到 256 个数据块组成。索引块包含数据块的块地址。

幼树型文件的关键是它的索引块,ProDOS 首先通过检索索引块内的数据块地址,来检索文件中的数据块,指向幼树型文件的项目的“存储类型”字的值为\$2。

4. 树型文件

一个树型文件包含比 128K 字节多而少于 16M 字节($\$20000 < EOF < \1000000)的数据。一个树型文件由一个主索引块、1 到 128 个索引块和 1 到 32768 个数据块组成。主索引块包含索引块的地址,每个索引块包含最大 256 个数据块的块地址。

树型文件的关键是主索引块。通过搜寻主索引块,ProDOS 可以找到所有索引块的地址,通过查找这些索引块就可以找到所有数据块的地址。

指向树型文件的项目的“存储类型”字段的值为\$3。

5. 使用标准文件

尽管可以使用文件项目内的“存储类型”来区分文件类型,系统程序或应用程序却可以用相同的方法对三类文件进行操作。一个程序很少读取一卷上的索引块或分配块,这由 ProDOS 来管理,程序只关心存在文件内的数据,而不关心它们怎样存储。

每种类型的标准文件是以从 0 到 EOF-1 的顺序来读取数据的。

6. 稀疏文件

稀疏文件是一个可以从文件读取超过实际存储在分配给文件数据块内的字节数的幼树型或树型文件。ProDOS 通过只分配那些已写入数据的数据块,以及需要指出它们的索引块,来实现稀疏文件。

例如,你可以定义一个 EOF 是 16K 的文件,它占用卷上的三个块,并只有四个数据字节写入它。如果你建立一个 EOF 为\$0 的文件,ProDOS 只分配关键块(一个数据块)给一个树苗型文件,并用空字符(ASCII \$00)填满它。

若你当时设置 EOF 和 MARK 为位置\$0565,并写入四个字节,ProDOS 计算位置\$0565 是文件的第三块(块\$2)的\$0165 字节($\$0564 - (\$0200 * 2)$),然后它分配一个索引块,存储当前数据块的地址于索引块的 0 位置,再分配一个数据块,存储那个数据块的

地址于索引块的 2 位置上,并在那个数据块的\$0165 到\$0168 字节存储数据。EOF 是\$0569。

若你现在设置 EOF 为\$4000 并关闭此文件,那么你有一个在卷上占有三个块的 16K 文件:两个数据块和一个索引块,你可以从这文件读取 16384 个字节数据。但在\$0565 之前和\$0568 之后的所有字节是零。

这样 ProDOS 只给文件中实际包含数据的那些块分配卷空间。

对于树型文件情况相似:如果赋予一个索引块的 256 个数据块中,没有一个被分配卷空间,则该索引块也未予分配。

另一方面,若你建立一个 EOF 为\$4000 的文件(使它具有 16K 字节或 32 个块长度),ProDOS 则分配一个索引块和 32 个数据块给一个幼树型文件,并用零填满数据块。

一个标准文件的第一数据块,不论是树苗、幼树或树型文件,总是被分配的。这样,在文件被打开时,总有一个数据块可读取。

7. 在文件内定位一个字节

对于在一个标准文件内寻找一个指定字节的算法在下面给出。

字节号	字节 2	字节 1	字节 0
位号 7	07	07	0

MARK

索引号	数据块号	块的字节
-----	------	------

用于:树型 树和幼树型 三种类型

若文件是一个树型文件,那么 MARK 的高 7 位确定指向此字节的索引块号(0 到 127)。7 个位元的值指出主索引块内索引块地址的低字节所在单元。索引块地址的高字节单元被这些 7 位的值加 256 所指出。

如果文件是一个树型文件或一个幼树型文件,那么 MARK 的次八个位元确定被索引块指示的数据块的块号(0-225)。这八位元值指出在索引块内数据块地址的低字节单元。索引块地址的高字节可从这偏移量加 256 得到。对于树、幼树和种树苗型文件,MARK 的低 9 位是所选择的数据块内该字节的绝对位置。

(未完待续)

(上接第 27 页)

BIOS 的扩充 ROM—BIOS10H 外部显示管理模块的功能,新设计了 16H 中断管理程序,使之可以支持 CEGA 和 CMGA 等显示系统,屏幕提示区扩大,提出格式优化,支持多种汉字输入法且具有联想功能,它有字词两种编码,可重复输入,能自动进行大小转换,自动显示内码、区位码和外码,提供窗口式帮助。

PC13 手写汉字输入系统(HGD-9200-1) 3 片 2000 元

该系统通过接口卡和标准 RS-232 串行接

口与主机连接,启动后即可用写字笔在书写板上手写汉字、绘图和制表等,实时地将汉字和图形输入计算机。凡是会写汉字的人,无需记忆任何编码,均可按手写汉字的习惯进行信息输入。

邮局汇款寄:北京市万寿路 173 信箱电子工业出版社软件部(邮编 100036)。王爱英、王旭

银行汇款:软件订购单附在银行汇款单内一并寄来(请注明软件名称)。

开户银行:北京工商银行翠微路分理处 帐号:891333-59

户名:电子工业出版社杂志编辑部

第五讲 编辑及有关操作

北京 FORTH 应用研究会(100034) 丁志伟

在学习使用一个语言版本时,了解这个版本的具体操作方法,是掌握这个版本的必经之路。在前几讲中,曾介绍过部分程序,用来说明问题,但这些程序如何输入、编辑、编译和调试,读者还不清楚。了解这些,不但对使用一个版本有必要,而且对理解 FORTH 系统是如何工作的,也很有好处。这一讲,就介绍有关编辑、程序格式及注释、磁盘存取、编译运行等操作知识。

前几讲介绍过一些小程序,或者说是某些单词的定义,有些单词较简单,比如问号词:

```
: ? @ U. CR ;
```

定义出这样的词,直接在键盘上输入即可。但这样做有两个问题:一是定义之后,程序已经过编译,源程序已不存在了;一是对于大程序,要有多行源程序才行。这都必须使用编辑器才能有效地进行处理。

笔者曾接触到十余种 FORTH 版本,发现这样一个现象:在众多高级语言中,FORTH 是比较重视标准化工作的,因此,学习过一种版本,就很容易掌握其他版本。然而 FORTH 中的编辑却是例外,它没有标准,各版本编辑部分使用方法各异,有些操作在不同版本中差别很大。这可能是利用 FORTH 设计编辑器比较容易的缘故,不论是哪个版本的编辑器,所占内存都不大,因而可以装入词典,常驻内存。这为 FORTH 的集成编程环境提供了条件,使得程序的调试非常方便。当然,也可以把编辑器从词典中删除,以便腾出更多的空间供其他程序用。FORTH 各版本中的编辑器经常与磁盘打交道,其处理的源程序文件可分为顺序文件方式和屏块方式两大类。

有些 FORTH 版本把源文件做为顺序文件处理。这种顺序文件与操作系统常用的顺序文件是一样的,只不过在这里,文件中存放的是 FORTH 程序。FORTH 环境以外的文本文件处理工具,也可以用来编辑和处理 FORTH 程序。当然,如果不在 FORTH 环境下,就只能进行编辑,而无法编译、调试。因此也就无法获得人机交互的编程效果。

对于大多数 FORTH 版本,其源程序使用的是屏块方式。这是 FORTH 中特有的文件处理方式,被编辑的内容以多个同样大小的块为单位,一块中所有内容同时显示在屏幕上进行处理,因此一块经常称为一屏,一屏通常占 1K(1024)字节,对于 PC 机,按 16 行、每行 64 个字符显示;对于 APPLE 机,则有 23×40、16×32 等不同显示格式。

无论是顺序文件或屏块方式,都分别有行编辑和全屏幕编辑方式,因版本不同而不同。编辑器的使用方法,无论简单或复杂,要想详细介绍,一讲的篇幅都是不够的,这里只能是简介。

先介绍 D1.0 中的编辑器和有关单词的操作方法。编辑器是词典中一部分,和系统其他部分一样,也是由一个个词构成的,在词典结构中与其他部分并无明显区别。

下面是编辑命令和有关单词。

BUF 变量。存放编辑缓冲区起址指针。

BTOP 变量。存放编辑缓冲区终址指针。

ADD 向编辑缓冲区加入新的内容。

操作: ADD↙

L 显示源程序。

操作: L n↙

或 L n1 n2↙

INS 在源程序中某行开始插入源程序。

操作: INS n↙

DEL 删除若干行源程序。

操作: DEL n↙

或 DEL n1 n2↙

FIND 查找字符串。

操作: FIND n↙

或 FIND n1 n2↙

显示出问号,再输入:

要查找的字符串↙

CH 替换字符串。

操作 CH n↙

或 CH n1 n2↙

再输入: 被替换的字符串↙

再输入: 新字符串↙

CLR 清除编辑缓冲区

SAVE 将缓冲区中源程序存盘。

操作: SAVE filename↙

LOAD 将编辑缓冲区内容编译。

操作: LOAD↙

D1.0 系统在内存中开辟了一块空间作为编辑缓冲区,被编辑的内容都存放在其中。BUF 和 BTOP 是两个变量,其中存放着缓冲区起址和终址指针,将其内容改变,就可以改变缓冲区位置和大小。

刚进入系统时,BUF 中内容是 6000H,即缓冲区

首址为 6000H,一般情况下,它是不变的。BTOP 是缓冲区终址指针,进入系统时,它也指向 6000H,即缓冲区是空的。BTOP 的内容随着编辑操作,因文件长度改变而随时变化。

在编辑新内容之前,一般应该先清除缓冲区,这要使用 CLR 这个词,相当于 BASIC 中的 NEW 命令。请看其定义:

```
: CLR BUF @ BTOP ! ;
```

它是把终址指针指向首址,缓冲区长度为 0,缓冲区就是空的了。

开始编辑时,要用

```
ADD
```

屏幕上会显示出

```
1
```

这个 1 是行号,由编辑器自动给出的,其后出现光标,随后可以输入程序。如

```
1 (LOOP TEST )
```

```
2 DEC
```

```
3 : TEST1 5 0
```

```
4 DO I . LOOP CR ;
```

```
5 :TEST2 5 0
```

```
6 DO I . CR LOOP ;
```

每输入完一行,打入回车,屏幕上会自动给出下一个行号,等待继续输入。如果想结束输入过程,继续打入两次回车即可。

如果想看一下缓冲区中的程序,要用 L 命令。比如想看第 3 行,就打入:

```
L 3
```

这时会显示出

```
3 : TEST1 5 0
```

左边的 3 是编辑器给出的行号

请注意:这里的 L 是一个单词,与其后的行号之间要加空格。其他编辑命令也是如此。

想看第 4 行到第 5 行,就用:

```
L 4 5
```

显示出

```
4 DO I . LOOP CR ;
```

```
5 : TEST2 5 0
```

有时需要修改某些程序,这时要用 ED。比如要修改第 4 行,就打入:

```
ED 4
```

屏幕上会先显示出第 4 行,然后把光标停在第 1 个字符,实际上是要求重新输入这一行。将程序修改之后,回车即可。

如果需要在程序中插入若干行,比如从第 5 行前边插入,就用:

```
INS 5
```

之后的操作与使用 ADD 命令相同。

有时 would 想删除某些行,则要用 DEL 命令,行号的用法与其他编辑命令相似。

编辑命令中还有 FIND(查找)、CH(替换)等,就不

介绍了。

被编辑的内容,始终是在内存中,一关机就消失了,因此需要将其存盘。这时需要使用 SAVE 命令。要给文件起个名。比如名为 TEST,就用

```
SAVE TEST
```

这样就缓冲区内容以文件形式存放在磁盘上,它是 T 类文件,名为 TEST。

反过来,把文件从磁盘中取出装入磁盘缓冲区,则要用

```
DAPP TEST
```

这样就能把 TEST 这个文件添加到 BTOP 指针之后,缓冲区原有内容依然保留,其效果与 ADD 输入相同。如果不想保留缓冲区内容,就先用 CLR 清除。也可以定义一个词:

```
: DREAD CLR DAPP ;
```

之后可以直接使用

```
DREAD TEST
```

先把缓冲区清除,然后将 TEST 文件取出存入缓冲区。

上边介绍了对源文件的编辑和存取。那么文件中的程序如何编译进入词典呢?假设文件已在编辑缓冲区,就用:

```
LOAD
```

如果源文件在磁盘中,要先用 DAPP 命令把文件调入内存,再执行 LOAD。

LOAD 这个词的功能是编译,把源文件编译成词典中的词。在其他语言中,编译是非常复杂的功能。现在看看 FORTH 中的 LOAD 是什么样子的。D1.0 版本中有反编译,执行:

```
FBY LOAD
```

会把这个词反编译出来。显示结果如下:

```
: LOAD BTOP @ 1 - INTOP !
```

```
BUF @ INPUT ! ;
```

对于 BUF 和 BTOP,读者已经知道,是编辑缓冲区的指针。INPUT 和 INTOP 是两个变量,其中存放着指针,指向正在被解释执行的输入缓冲区指针。它们平常指向键盘缓冲区,而 LOAD 这个词的作用是指向编辑缓冲区,因此,能从键盘输入的内容都可以放在编辑缓冲区执行。源程序中存放的不一定是定义,也可以有其他内容。比如改变进位制、删除词典一部分、执行某些单词、加注释等。这样的文件,有些地方类似于 DOS 中的批命令文件。

其他版本中的编译词(LOAD)一般要比 D1.0 版本中的复杂,但总的来说,FORTH 中的编译比其他语言的编译简单得多。

D1.0 版本中还有一个编译词 DLOAD,它可以把一个磁盘文件直接编译进词典而不影响编辑缓冲区,这为较复杂程序的编译提供了方便。比如,某些情况下要把汇编器中途调入内存,又不想将其加入缓冲区,这时就可以使用 DLOAD。

上边介绍了 D1.0 版本中的编辑器和与之有关的

操作。这个编辑器功能简单,但也还好使,能满足一般需要。编辑、磁盘存取、调试和运行等各种操作,基本上是在同一个界面上进行,或者说,这是一个集成调试环境,便于人机交互,使得编程效率很高,其方便程度与 BASIC 相近。当然,其他版本的 FORTH 语言,大多也是集成环境。

下面再介绍一下 PC/FORTH 2.0 的编辑器。

PC/FORTH 的编辑对象,是磁盘中的文件,其后缀是 SCR,称为屏幕(SCREEN)文件。如果要编辑的文件不存在,就先建立。键入:

SCOPY ✓

屏幕上会显示出一系列提示,按提示输入之后,即可建立一个新的屏幕文件。

系统把屏幕文件划分为若干块,每块为 1K(1024)字节,一块内容同时显示在屏幕上,按 16×64 个字符显示出来,又称为一屏。对一屏做为一个整体进行全屏编辑。编辑器把文件按随机文件(或者说以虚拟存储)的方式进行处理。

使用编辑器,要输入

EDIT ✓

对于 PC/FORTH,刚进入系统时,编辑器并不在内存,此时打入 EDIT,会将编辑器取出,存入词典,然后进入编辑状态,下一次再打入 EDIT 时,由于编辑器已调入内存,就会直接进入编辑状态。此时屏幕上会显示出标题和一些提示信息,然后在屏幕上显示出:

Command:

这是要求你输入编辑命令。如果不知道有哪些命令及这些命令的功能,可以按一下 F1 键(这是帮助键),会显示出这些命令及说明。它们是:

- C 拷贝一屏
- D 显示磁盘目录
- E 开始编辑
- F 显示磁盘目录开关
- H 高亮度显示
- I 屏面索引
- M 移动一组屏面
- O 关闭高亮度显示
- T 设置表格间隔
- U 编辑新的屏幕文件
- ESC 键 退出编辑

这些命令中,最主要的也最常用的,是 E 命令,按一下 E 键,屏幕上会接着显示:

Enter screen number:

意思是要求你输入一个屏块号。一个文件可以有多个屏块,每一个屏块有一个编号,从 0 数起。比如编辑第 5 屏,就打入 5,再回车,屏幕上会显示出第 5 块的内容,随后可以对这屏做各种输入、修改等编辑工作。这是一个全屏编辑,功能较多,大多数操作与 WORDSTAR 等常见的编辑器相同或相似。屏幕下方还有一些功能键的提示。

编辑命令摘要如下:

光标命令:

上下左右方向键。

TAB 键 将光标移动到下一个表格位置。

Home 键 光标回左上角

End 键 光标移到行尾

词命令:

^ L 光标移动到前一个词开始

^ R 光标移动到下一个词开始

^ T 删除光标右边一个词

行命令:

^ Y 删除当前行

^ N 插入一个空行

Alt-N 从光标位置插入一个空行

行命令中有一组行操作值得重点介绍。PC/FORTH 将堆栈概念作用于编辑,可以把一行源程序做为一个单位,压入编辑器的行堆栈。这个行栈是独立的,与参数栈或返回栈都不是一回事,最多可以放 16 行。行栈命令有 4 种:

F3 键把当前行压入行栈中,光标下移一行,屏面内容不变。

F4 键把当前行压入行栈,并在屏幕上删除此行。

F5 键 从行栈中弹出一行,屏幕上当前行被栈中内容复盖。

F6 键 从行栈中弹出一行,插入光标所在位置。

字符命令:

^ G 删除光标所在字符

Del 键 删除光标左边字符

←(BS 键) 退格

Ins 键 插入/复盖。这是一个开关,能改变当前输入状态

字符串命令:

F7 键 查找字符串

F8 键 替换字符串

屏面命令:

F9 键 删除整个屏幕

F10 键 恢复编辑前的屏面内容

Alt-F9 删除光标之后的屏面内容

PsUp 键 把当前屏面内容存盘,而将上一屏取出进行编辑

PgDn 键 把当前屏面内容存盘,而将下一屏取出进行编辑

^ PgUp 转到 0 号屏

^ PgDn 转到最后一屏

其它命令:

F1 键 帮助键。显示出各种操作功能

F2 键 把当前屏面第 1 行做为注释行,加上日期

ESC 键 退出编辑

PC/FORTH 因为使用全屏屏块方式的编辑,操作就与 D1.0 版本的编辑差别很大。

不进入编辑,又想查看一个屏块内容时,可以用 LIST。

n LIST

可以把第 n 屏显示出来。用法与 D1.0 版本不一样。

编译与 D1.0 版本一样,也是用 LOAD,但使用方法与概念不一样。编译第 3 屏,要用:

3 LOAD

这样只能编译出一个屏幕内容。多个屏幕的编译有如下几种处理方法:

如果连续编译几屏,比如第 4~7 屏,可以在第 4~6 屏的最后,各加一个“-->”,这是一个词,表示继续编译下一屏。这样,执行 4 LOAD,就可以把第 4~7 屏连续编译。

也可再用一屏放入几个编译命令。还是用第 4~7 屏做例子,可以在第 8 屏中装入 4 LOAD 5 LOAD 6 LOAD 7 LOAD,然后执行 8 LOAD,系统就会自动执行第 8 屏的几个 LOAD 命令。这个第 8 屏,相当于 DOS 里的批命令。用这个方法,被编

译的几个屏块不一定连续,也不一定按从小到大的顺序。当然,定义出的各单词的逻辑关系和调用关系还是要注意的;

还可以用 THRU 命令。执行

4 7 THRU

就会把这几屏编译出来。请看其定义:

: THRU 1+ SWAP DO I LOAD

LOOP :

这是一个循环,把 n₁~n₂ 各屏连续编译。

后两种方法,被编译的屏块不应有“-->”。

PC/FORTH 中,被编辑的内容是自动存盘的。在退出系统时使用 BYE 就能将文件安全存盘。

这一讲中介绍了 D1.0 版本和 PC/FORTH 2.0 版本编辑器的主要功能,若要全面掌握,这里的篇幅是不够的,要对照说明书实际上机才行。这两个版本都有说明书,不妨在需要时找来参考。

邮购消息

由本期起本刊将陆续登电子工业出版社软件出版的软件目录及简介,欢迎选购。邮购办法:按软件定价通过邮局或银行汇款我部,不收邮费。收款后一周内发货。确保售前咨询、售后服务。

电子工业出版社软件目录

一、PC 机软件

- PC01 报表程序自动生成工具 GT5.1 1片 260元
经选择不必编程序即可生成 PRG 文件。特点:可任选择打印字段及打印字段顺序,可进行多库连接打印,多层实线表头打印,超长字段打印,可自动增加小计,计算字段,可自动分页,标题自动居中;可将英文字段名替换中文输出。
- PC02 任意条件查询程序自动生成工具 1片 180元
经选择可生成在 FOXBASE 下运行的任意查询程序,在生成的条件查询中进行挑选,可按关键字挑选,可浏览输出查询结果,可调接口程序处理结果(如屏幕输出、打印输出、调 AT、GT 及用户自定义扩展接口)。
- PC03 彩色屏幕界面程序自动生成工具 1片 260元
本软件通过选择菜单,利用上下左右键即可确定其在屏幕上的位置,自动生成用户设计的在 dBASE 或 FOXBAS 环境下运行的程序,该程序能并入用户开发的系统。
- PC04 通用数据管理系统 1片 450元
适用于企事业单位的产品管理,人事管理,事务管理等,特别适合于非计算机专业又从事计算机管理工作的人员使用。可任意更改库结构,输

- 出报表格式任选、可同时管理多个库。
- PC05 CTE 通用汉字表格编辑软件 H1片 200元
本软件具有全屏幕编辑软件所有功能,还能在所编辑的文章任意位置制表或画框,可将某块文字或表格满屏移动,可输出超宽表格。控制键只有 F1~F8,易学易懂。
- PC06 LD 磁盘文件查询 1片 50元
本软件扩大了 DOS 命令,相当于将 DIR 命令和 TYPE 命令合起来,特别适合于文件的查找与管理,可拼接在 DOS 下使用。
- PC07 通用四笔两码汉字输入法(LMDOS 5.0) 1片 60元
利用汉字的声和形来拆分汉字,重码率低,初学者易于掌握。有引导编辑、驱动程序。8000 条词组。
- PC08 表形码汉字录入系统 2片 260元
利用汉字的象形来拆分汉字,易于学习,提高录入速度。
- PC09 凸轮廓线的计算程序 1片 150元
适用于机械厂、研究所等机械加工方面的轮廓线设计与计算。
- PC10 汉字复杂报表自动生成系统 2片 150元
本软件采用菜单方式,能自动生成多层表头汉字报表,表格修改方便,还能被其他管理系统调用,使已用 dBASE 进行事物管理的系统得到扩充。本软件在 DOS 系统下运行,操作简单,兼容性好。
- PC11 模拟电子线路辅助教学软件 3片 460元
本软件包括本门学科的所有内容,适用于大、专、电大、夜大等电子线路课堂教学。提供了所有源代码,供分析数据结构及编程用。
- PC12 多方案联想长城机 DOS 3片 200元
本系统是在长城系列机及其兼容机上运行的新一代汉字磁盘操作系统,它沿用了原 GW

(下转第 23 页)

C 语言试题与分析

天津大学(300072) 赵国瑞

随着 UNIX 操作系统的广泛使用和标准化, C 语言也得到了非常广泛地应用。自从 87 年美国标准 C 语言出台以后, 它更成了微小型机和工作站的“标准”程序设计语言。C 语言具有高级程序语言的结构和编程环境, 同时提供了类似汇编语言那样的系统资源管理能力和程序的执行效率。近年来, 愈来愈多的应用软件设计人员选择 C 语言作为主要编程工具。我国计算机软件专业技术水平考试自 1990 年起把 C 语言规定为五种程序设计语言之一。在初级程序员级资格考试中, 它与 BASIC 语言是两种供选择的编程语言。

下面重点介绍在初级程序员级考试中对于 C 语言语法的要求、C 语言试题和例题以及分析和解答的方法。

一、C 语言语法要点

C 语言的语法, 大致可分为以下几个方面:

1. 数据类型

C 中的数据类型分为两大类:

1) 简单类型

字符类: char(字符型)、unsigned char(无符号字符型)、signed char(有符号字符型)

整数类: int(整型)、unsigned int(无符号整型)、signed int(有符号整型)、short int(短整型)、unsigned short int(无符号短整型)、signed short int(有符号短整型)、Long int(长整型)、unsigned long int(无符号长整型)、signed long int(有符号长整型)

实数类: float(浮点型)、double(双精度型)、long double(长双精度型)

指针: 可指向任何类型的变量或函数。

2) 复合类型

数组: 由相同类型数据元素构成的序列。

结构(struct): 可由不同类型的成员构成的序列。

联合(union): 对同一存储区可由多个不同类型的成员共用的一种数据类型。

枚举(enum): 是一个有名字的整型常量的集合, 这些整型常量是该类型变量可取的所有合法值。

2. 运算符

C 语言除了具有一般高级程序语言的算术运算符、关系与逻辑运算符之外, 还含有位运算符、指针运算符、条件运算符、赋值运算符等。各运算符所用符号按优先级由高到低编排如下表:

3. 函数与函数库

一个 C 程序是由多个预处理命令、说明、定义构成的。

优先级	运算符	名称	结合特性
15	() [] .、→	函数参数表 数组下标 结构取成员	从左至右
14	~ ++、-- - * & sizeof	逻辑非 位串补 增量与减量 负号 取内容运算 取地址运算 判断类型大小	从右至左
13	*/、%	乘、除与取模	从左至右
12	+、-	加与减	
11	<<、>>	算术左移与右移	从右至左
10	<、<=、 >、>=	小于、小于等于、大于、大于等于	从左至右
9	==、!=	等于与不等于	
8	&	按位与	
7	^	按位异或	
6		按位或	
5	&&	逻辑与	
4		逻辑或	
3	?:	条件运算	从右至左
2	+ =、- =、 * =、/ =、 < =、> =、& =、^ =、	(简单)赋值 复合赋值	
1	=、,	顺序计算	从左至右

“预处理命令”告诉 C 编译在第一遍扫描时, 对源程序正文所作的处理。最常用的预处理命令有两个:

#include <文件名>或“文件名”

它指明一个标识符和一个字符串的关系。表示在源程序中每次遇到该标识符时均以指明的串去替换它。通常标识符中仅使用大写英文字母。

#include <文件名>或“文件名”

这个命令将使得指明的文件内容嵌入到该命令所出现的位置。新嵌入的文件中还可包含子处理命令并同样被识别和处理。

“说明”将为变量、函数或类型的名字与属性之间建立联系, 但并未分配存储单元。如以下是两个“说明”

的例子:

```
struct tm {
    int hours;
    int minutes;
    int seconds;
}; /* 结构类型说明 */
extern int max(int,int); /* 函数说明 */
```

“定义”将使变量名字与它们的初值相联系,使函数名字与它们所实现的操作(即函数体)相联系,并实际分配存储单元。如对于上面两个说明的“定义”可为:

```
Struct tm time={0,0,0};
int max(a,b)
int a,b;
{ if (a>b) return (a)
  else return (b);
}
```

C 语言中,允许函数递归调用,即一个函数的函数体中可以直接或间接调用它本身。

在美国标准 C 中,没有规定执行输入输出的任何语句,但却定义了完成输入、输出及其它许多任务的函数库。这些函数及其所访问的数据和变量都通过头文件给予说明。标准 C 所规定的头文件以及作用如下:

assert. h	定义 assert() 宏命令
ctype. h	字符操作函数
float. h	定义从属于环境工具的浮点值
limits. h	定义从属于环境工具的各种限制
locale. h	支持函数 setlocale()
math. h	数学函数所使用的定义
setjmp. h	支持非局部跳转
signal. h	定义信号值
stdarg. h	可变长变量参数表
stddef. h	定义某些常数
stdio. h	流文件 I/O 函数
stdlib. h	其它说明
string. h	字符串函数
time. h	系统时间函数

除了上面的函数外,大多数 C 编译程序还提供了另外一些其它功能的函数,如图形函数、进程管理函数、DOS 接口函数、目录操作函数、屏幕操作函数等。这些函数库的使用,大大增强了 C 语言的功能。

4. 语句

跟其它高级程序设计语言一样,C 语言中也有多种描述表达式计算、选择、循环和转移控制的语句:

表达式计算语句中包括赋值语句、函数调用语句等。

选择语句中 if-else 语句可以实现二向分支,switch-case 语句可以实现多向分支。

循环语句中 for 语句用于计数型循环,While 语句用于实现当型循环,do-while 语句用于实现至少执行一次循环体的循环。

转移控制语句中 goto 语句实现无条件转移,break 语句实现终止直接包含它的 do、for、while 循环语句和 switch 语句,continue 语句实现跳过循环语句中该语句之后的全部语句而进入下一次循环;return 语句实现终止其所在函数的执行,将控制返回到函数调用语句处继续执行。

通过以上介绍可以看到,与其高级程序语言比较,C 语句中的语句种类是很少的(包括空语句和复合语句共计 12 种)。但是它有较强的数据描述能力和操作能力。同时,丰富、灵活性质各异的函数库为程序设计人员提供了施展才华的良好环境。

对于资格考试初级程序员级来说,考生应该比较全面掌握 C 语言各方面的内容。特别是整数、浮点、字符、数组、指针等数据类型及其有关运算符的功能和优先顺序,选择和循环语句的使用、基本输入、输出函数及格式描述等应该达到相当熟练的程度。而对于某些较难掌握的内容,如结构、联合和它们的指针的使用,函数的递归调用以及外部文件的随机访问等内容,前两年虽没涉及,但随着应用水平的提高,对考生的要求势必越来越高,至少也应熟练阅读。

二、C 语言试题类型及例题

从已有的初级程序员试题来看,其形式有两种:一种是给定程序写出运行结果,一种是给定问题和算法,给出不完整程序,把正确的答案填入程序的空格栏中。而对程序员和高级程序员的 C 语言试题则仅有后一种形式。

给定程序写出运行结果,解答这类试题的关键是从主函数第一条可执行语句开始,逐句跟踪各个变量内容的动态变化,模拟计算机执行该程序的过程,最后按照题意和格式要求写出所输出的结果。C 语言的这类试题比 BASIC 的稍难:一是只给一个 C 程序,写出它的全部输出,若前面输出结果回答得不对,往往导致后面回答错误,而 BASIC 是给出 4 个独立的短程序,分别填写运行结果;二是 C 语言的数据类型和运算符以及描述选择、循环的语句比 BASIC 丰富得多,各种变量名有它们各自的作用范围,不象 BASIC 那样所有变量名都在整个源程序内有效(新一代的 BASIC 已改变了这种情况),这样就给阅读理解 C 程序增大了难度。但是只要熟记 C 的语法,认真阅读程序,细心记载变量值的变化,就能正确地解答这类试题。

例题 1[1991. 下午、四]阅读下列 C 程序,并写出运行结果。

```
#include <stdio. h>
#define LEN 8
main( )
{int j,c;
 static char num[2][LEN+1]={"17208980",
 "28219198"};
 c=0;
 for (j=LEN-1;j>=0;j--)
 {c+=num[0][j]+num[1][j]-2* '0';
```



```

printf("% d\n",c);
num[0][j]=c%10+'0';
c=c/10;
}
printf("%s\n",&num[0][0]);
}

```

[分析与解答]在阅读本题时应注意以下问题:1)C语言中字符与整数可互相转换,例如 '8'-'0' 结果为 8,8+'0';2)两个整数相除;结果为整数,这与 BASIC 不同,而与 FORTRAN 相同,如 3/5 结果为 0,19/10 结果为 1;3)字符串输出格式说明 %s 要求对应输出项目为被输出字符串指针,即起始地址,且输出时直到遇空字符(即 NUL)结束;4)C 语言中未被初始化的变量的初值均为 0。

注意到以上问题,仔细追踪各变量在程序执行中的变化,不难得出其正确答案:

```

8
17
11
18
2
4
15
4

```

45428178

例题 2[1992、下午、四]阅读下列 C 程序,并写出其运行结果。

```

#include <stdio.h>
#define MN 7
int num_list[ ]={3,4,5,6,7,8,9};
main( )
{int k,j,b,u,w;
u=0;
w=MN-1;
while (u<=w)
{j=num_list[u];
k=2;
b=1;
while (k<=j/2 && b)
b=j%k++ ;
if(b) printf("% d\n",num_list[u++]);
else
{ num_list[ ]=num_list[w];
num_list[w--]=j;
}
}
for(k=0;k<u;K++)
print("% 5d",num_list[k]);
printf("\n");
}

```

[分析与解答]此题的特点是表达式中含增量、减量运

算符较多,增量/减量运算符的位置决定操作顺序,如语句“b=j%k++,”等价于“b=j%k;k=k+1;”,而语句“b=j%++k”则等价于“k=k+1;b=j%k;”。另外本题变量值的变化较紊乱,因此,须格外仔细掌握它们各自的当前值。为了记忆各变量的当前值,一般在跟踪中采用表格方式。对于本题可列出如下表格:

num_list	0	1	2	3	4	5	6	←下标值	
u=0 时	3	4	5	6	7	8	9	←数组元素值	
u=1 时		9				4			
u=2 时		8				9			
u=3 时		7				8			
u	w	j	k	b	打印结果				
0	6	3	2	1	首次循环 3				
		1	4	2	1	} 2 次循环			
		5	3	0					
			9	2	1	} 3 次循环			
		4	3	0					
			8	2	1	} 4 次循环			
		3	3	0					
			7	2	1	} 5 次循环			7
			3	1					
			4	1		} 6 次循环			5
2		5	2	1					
			3	1		} 末次循环			3 7 5
3		6	2	1					
		2	3	0					

有了这样一个表格,就很容易检查校对和写出答案:

```

3
7
5
3 7 5

```

这个试题充分体现了前后结果的连贯性,无论在 哪一步上疏忽出错,都会影响最终答案的正确性。

要解答初级程序员的填空完善程序类的试题,不仅要求应试者熟练掌握 C 语言本身,而且要求考生有较广泛的常用算法知识,如各种查找、更新、排序、字符处理的算法,简单图形(线段和圆)的计算机图示算法,初等统计算法、初等数论算法、初等代数算法、基本数据结构操作算法等。对于程序员和高级程序员的此类试题,还要求考生熟悉指针、结构、联合外部文件等的处理,同时,对于迭代、递归、数值积分、插值、矩阵计算、方程求解等算法知识也要有深入的了解。因此,解答这类试题的关键是认真阅读程序说明,找出其中所涉及的算法和实现方法。一般来说,每道这类试题中常常涉及 2—3 个算法。

(未完待续)

BJS-98 硬件、软件典型实验

清华大学 袁涛 魏峰

BJS-98 单片机多功能实验系统既适用于进行单片机基础实验,同样也适用于单片机高级实验,如智能仪器实验。BJS-98 既能直接面向 PL/M-96 结构化高级语言编写和调试程序,也能面向宏汇编语言编写和调试程序。BJS-98 不用扩展就能完成一百余个实验,其中包括:程序设计实验、浮点数据处理实验、程序库建立和维护实验、I/O 接口实验、LED 显示实验、时钟实验、音乐与声响实验、顺序控制实验、交通灯管理实验、人—机接口实验(键盘与数码显示)、串行通信实验、A/D 实验、D/A 实验、高速输入(HSI)实验、高速输出(HSO)实验、中断实验等。以上是 BJS-98 的独到之处。

早期在微型计算机上曾强调要掌握汇编语言编程,但后来几乎已经没人这样做了。大部分进行程序设计的人员甚至不曾接触过汇编语言。单片机程序设计也正在进行类似的过渡。由于 PL/M 语言的推广使用,这不仅成为可能,而且正成为现实。因此,BJS-98 的实验都使用 PL/M-96 高级语言(也允许使用汇编语言)。由于篇幅所限,下面仅介绍其中的几个实验。

一、浮点数据处理实验

(一)实验内容:

一组测量数据分别为:6.108V,6.103V,6.116V,6.097V,6.121V,6.119V,6.120V,6.111V,6.105V,6.116V,6.101V,6.107V,6.126V,6.111V,用 WZ(I)来记录上述数据,然后计算其算术平均值 WCP,残差 WV(I)=WZ(I)-WCP,

$$\text{方差 } WD = \frac{1}{14} \sum_{i=1}^{14} WV^2(I),$$

$$\text{标准差 } WS = \sqrt{\frac{1}{14} \sum_{i=1}^{14} WV^2(I)}$$

(二)程序流程图

(三)程序清单

```
START;DO;
$NOLI
$IC(MATH96.PLM)
$LI
DECLARE WZ(14) REAL DATA(6.108,6.103,6.
116,6.097,6.121,6.119,6.120,6.111,6.105,
6.116,6.101,6.107,6.126,6.111);
DECLARE WV(14)REAL;
DECLARE(WCP,WD,WS,SUM,SUMW)REAL;
DECLARE I SHORTINT;
CALL INIT $ REAL $ MATH $ UNIT; /* 初始化
实数数学
单元 */

STA;SUM=0.0; /* 清累加寄存器 */
DO I=0 TO 13; /* 计算和 */;
SUM=SUM+WZ(I);
END;
WCP=SUM/14.0; /* 计算平均值 */
SUMW=0.0; /* 清累加寄存器 */
DO I=0 TO 13;
WV(I)=WZ(I)-WCP; /* 计算残差 */
SUMW=SUMW+WV(I)*WV(I); /* 计算方差 */
END;
WD=SUMW/14.0
WS=SQRT(WD); /* 计算标准差 */
GOTO STA; /* WS=8.183148E-003 */
END START;
EOF;
```

(四)实验操作步骤:假定源文件名为 SA.PLM

1. 输入源程序。可使用 PE、WS、EDLIN 等编辑软件。

2. 使用 PSA.BAT 对程序进行预处理,编译、连接定位等。

C>PSA SA

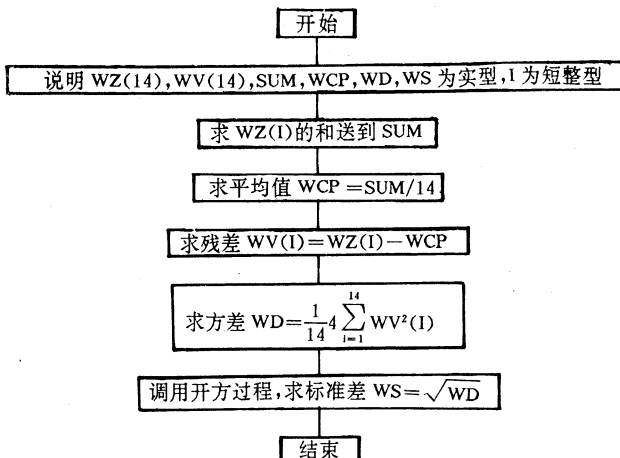
也可不使用批文件 PSA.BAT,而逐项单独操作,如:

C>TP SA.PLM

C>PTM96 SA.PLM DB SB

.....其它操作详见实验手册。

3. 在 BJS-98 上使用 CDW 窗口调试器运行调试程序:



C>CDW SA

此后则可使用单步、断点等运行方式进行运行调试。

说明:批文件 PSA.BAT 的使用主要是为了简化操作,其中每行都是一条 DOS 操作命令,批文件可用 PE、WS、EDLIN 等编辑软件输入或修改。为节约篇幅,下面几个实验将不再叙述详细操作步骤。

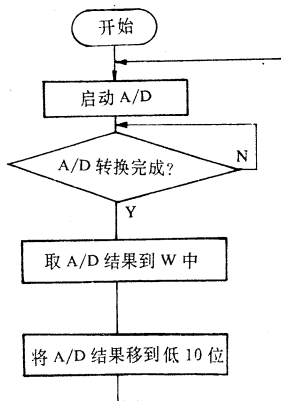
二、模数转换(A/D)实验

(一)实验内容

使用 8098 片内的 10 位 A/D 转换器,其转换速度可高达 22 μ s,可工作在查询方式,也可工作中断方式下,即可立即转换,也可定时转换。本实验使用查询方式,使用 4 个输入通道中的 ACH5 通道,采用立即转换方式。A/D 转换结果取出放在字型变量 W 中,然后将 A/D 值由 W 的高 10 位移到低 10 位。

8098 特殊功能寄存器 AD-COMMAND 的 bit0 和 bit2 置 1,表示选择 ACH5 通道,bit3 置 1,表示立即启动 A/D。A/D 结果寄存器 AD-RESULT 的 bit3,表示 A/D 目前的工作状态。

(二)程序流程图



(三)程序清单

```

M:DO;
$NOLI
$IC(8096.PLM)
$LI
  DECLARE W WORD;
  DECLARE B BYTE AT(.AD-RESULT+1);
  DECLARE WH BYTE AT(.W+1);
LOOP:AD-COMMAND=0DH; /* 也可写 0000 $1101B */
      * /
      R0=0; /* 等待(延时) */
      R0=0;
      DO WHILE BITTST(.AD-RESULT,3); /* 判断 A/D 是否完成 */
      END;
  
```

```

W=AD-RESULT; /* 取 A/D 结果低字节 */
WH=B; /* 取 A/D 结果高字节 */
W=SHR(W,6); /* 将 A/D 结果移到低 10 位 */
GOTO LOOP;
  
```

```

END M;
EOF;
  
```

(四)实验操作步骤:

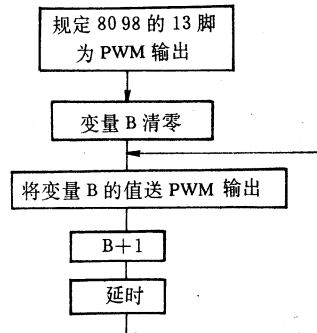
同实验一。进入 CDW 窗口调试器后,可在 GOTO LOOP 语句设置断点(用 F9),用 F5 运行后,在断点处查看 A/D 结果,可将 W 放在观察窗口中,也可用会话方式查看,如操作 EW,W 则可看到。改变电位器 W1 的值,则 A/D 结果相应改变。

三、D/A 输出后锯齿波

(一)实验内容:

使用 8098 的 PWM 与 BJS-98 板上的积分电路(电阻、电容和运放组成,见本刊前一期电路原理图)组成 D/A 电路。此实验也可将 PWM 改为 HSO,但程序也要相应变化。用万用表电压挡在运放输出端测量,或用示波器观察。

(二)程序流程图



(三)程序清单

```

M:DO;
$NOLI
$IC(8096.PLM)
$LI
  DECLARE B BYTE;
  IOC1=1; /* 选择 PWM 输出 */
  B=0;
LOOP:PWM-CONTROL=B; /* D/A 输出 */
      B=B+1;
      CALL TIME(10000); /* 延时 1 秒 */
      GOTO LOOP;
END M;
EOF;
  
```

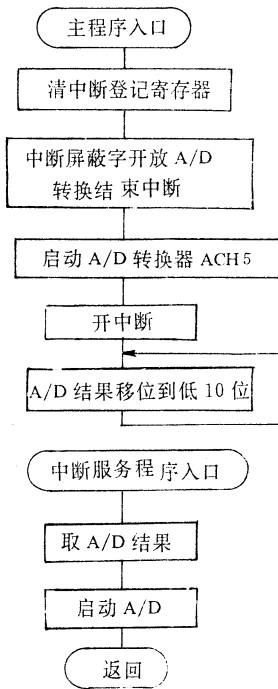
四、中断方式模数转换(A/D)实验

(一)实验内容

本实验与实验二类似,区别在于这里使用了中断方式,A/D 完毕后单片机内部自动发出中断请求信号,然后程序自动进入相应中断服务程序,A/D 转换

结束中断的级别为 1。

(二)程序流程图



(三)程序清单

```

M:DO;
$NOLI
$IC(8096.PLM)
$LI
  DECLARE W WORD;
  DECLARE B BYTE AT(.AD_RESULT+1);
  DECLARE WH BYTE AT(.W+1);

```

```

ADINT:PROCEDURE INTERRUPT 1;
  W=AD_RESULT;
  WH=B;
  AD_COMMAND=0DH;
  END ADINT;
START:INT_PENDING=0;
  INT_MASK=02H;
  AD_COMMAND=0DH;
  ENABLE;
LOOP:W=SHR(W,6);
  GOTO LOOP;
END M;
EOF;

```

(四)有关事项

具体实验步骤与实验一相似。运行调试时,可在中断服务程序 ADINT 中,用 F9 设置一个断点,这样就可以判断是否产生了 A/D 中断并进入相应中断服务程序,如果中断工作正常,程序必然很快会执行到这个断点。

五、三角函数运算实验

本实验内容比较简单,下面仅给出程序清单,其中以正弦和正切为例。程序清单如下:

```

M:DO;
$NOLI
$IC(MATH96.PLM)
$LI
  DECLARE(R1,R2) REAL;
  CALL INT $ REAL $ MATH $ UNIT;
  R1=3.1415926/4.0;
STA:R2=SIN(R1); /* R2=0.707 */
  R2=TAN(R1);
  GOTO STA;
END M;
EOF;

```

裕兴中英文游戏机电脑键盘

质量高、功能强、同类产品价最优

带有中文功能的、裕兴中英文游戏机电脑键盘问世半年多来,以其鲜明的特色、低廉的价格、完善的售后服务,受到广大用户的欢迎,成为同类产品中的名牌。它独有掉电保持、录音机测试功能,带有音乐、英语、数学计算等多种演示程序,同类产品性能价格比最高,是中、小学学生

学习计算机知识的最佳选则。配中英文 2 合 1 卡,可学习中英文、练打字、作曲演奏。利用 F BASIC 语言编制学习和游戏程序,手册详尽。邮购每套 315 元,免邮费并保修一年。索取资料附回信封,北京西城西教场胡同 35 号,裕兴机械电子研究所,电话 2252309,邮编 100035。

MCS-51 单片单板机监控程序分析(下)

北京市电信学校 江琪 刘葳

三、单步执行命令

单步执行命令,即 CPU 从用户指定的地址处开始执行程序,但一次只执行一条指令,然后在 LED 上显示出下一条指令的地址和累加器 A 的内容。以后可重复单步执行,直到程序结束。很显然,这是调试程序的重要手段。那么如何使 CPU 在执行完一条指令后停下来呢?以后又怎样接着运行呢?这正是本节叙述的内容。

1. 保护现场问题

单步执行过程也是一个资源竞争过程。当单步执行用户程序时,8031 的全部资源(片内 RAM)应交给用户使用。执行这一条指令以后,8031 的全部资源又要交给监控程序。以后如再次单步执行用户程序时,还应恢复上次执行的结果。显然这里有一个保护现场和恢复现场的问题。我们使用片外 RAM 1700H 开始的一片 RAM 区来作为内部 RAM 映像区保护用户程序的片内 RAM 资源,即用片外 RAM 的 1700H 单元保护片内 RAM 的 00H 单元内容,用 1701H 单元保护 01H 单元,……,用 17F0H 单元保护 0F0H 单元。恢复现场的过程正好和保护现场相反。那么什么时候需要保护现场,什么时候需要恢复现场呢?

我们说,在执行单步命令之前,内部 RAM 一直是由监控程序来使用的,但是在执行单步命令时,内部 RAM 资源全部交给用户程序了,执行了一条指令以后,就应该及时进行现场保护,将用户使用的内部 RAM 内容及时保存到外部 RAM 中去。以后监控程序完成将下一条指令地址和用户程序中的累加器 A 的内容送 LED 显示,此时如用户使用其它监控命令检查内部 RAM 时,实际上都是在检查相应的外部 RAM 的内容。在执行了一次单步命令而再次运行单步命令时,我们说首先应该进行恢复现场工作,即将上次单步执行的结果(原内部 RAM 内容保存在外部 RAM 映像区处)恢复到内部 RAM 中去,使用户程序的执行具有连续性。这段程序如下:

```
STEP_KEY:MOV A,7EH ;检查是否存在四位地址
          JB ACC.4,STEP0 ;不存在转走,退出不执行单步命令
          ACALL DZ1 ;DZ1 子程序完成将四位地址压入堆栈操作
          MOV A,7AH ;是否第一次执行 STEP 命令
          JB ACC.4,STEP;DG5 灭;表明是第一次,转走;无需恢复现场
          ACALL USE0 ;调恢复现场子程序,此处略
```

STEP:↓;进一步进行处理

2. 单步执行命令的技术关键

8031 能够保证在执行了一条用户指令以后,可靠地停下来而去执行监控程序是靠付出了屏蔽中断 0 这一资源而换来的。

在这里,首先要明确一些概念。如果外部引脚 $\overline{INT0}$ 有一个中断请求信号,8031 只能在软件执行了 STEP: SETB EA;允许中断,此处的 STEP 标号接上节

```
CLR IT0;电平触发方式
SETB PX0;置 $\overline{INT0}$ 为高优先级中断
SETB EX0;允许 $\overline{INT0}$ 中断
```

这些指令以后,才能响应中断。另外,8031 还有一条规定,即如果在上面的程序段后面跟着 RETI 指令,则 CPU 需在执行 RETI 指令后再执行一条指令才允许进入中断应答状态。这样如果在上面程序段后面再加上一条 RETI 指令,就能保证单步执行命令。具体过程是:

①选用一开关来作为单步操作选择开关。当开关拨到单步操作端时,则 8031 的 $\overline{INT0}$ 引脚接至地;当开关拨到另一端时, $\overline{INT0}$ 悬空。这样,在执行单步命令前,先把开关拨到单步操作端,使 $\overline{INT0}$ 发出中断请求信号。

②置单板机为下档状态,键入四位地址(单步运行的程序首地址),键入 STEP 命令键,则如前所述监控程序首先将四位地址压入堆栈,然后执行上述的 STEP 程序段(最后一条是 RETI 指令),在执行 RETI 指令以后,8031 将上面压入堆栈的四位地址弹出到 PC 中,并执行这条指令,然后响应中断转到 0003H 处去执行。在地址 0003H 处我们安排了如下三条指令:

```
CLR EX0 ;屏蔽 $\overline{INT0}$ 中断
ACALL USE0 ;调用保护现场子程序
LJMP DISP
```

③CPU 在执行了一条指令并响应中断后,自动将用户程序的下一条指令地址压入了堆栈。单步执行的最后一步是将下一条指令地址及累加器 A 的内容送 LED 显示。程序段如下:

```
DISP:POP DPH ;将下条指令地址弹出至 DPTR 中
      POP DPL
      ACALL DSD1;DSD1 子程序将 DPTR 内容送显示缓冲区
      MOV DPTR,#17E0H ;从 RAM 映像区中取出累加器 A 的内容
      MOVX A,@DPTR
      MOV R0,#79H ;DSD0 子程序将累加器 A 内容送
```

;显示缓冲区后两位(R0=79H)

ACALL DSD0

AJMP K-P0;转到主程序段

四、断点的设置与处理

断点(Break Point)是调试程序的重要手段。本监控程序设计最多允许建立四个断点,若强行设置第五个断点,监控程序认为非法而不予理会,但不影响已设置的四个断点。断点只能在每条指令的第一字节处设置,否则将造成程序执行错误。在ROM区中设置断点也被认为是非法。

设置断点的方法是:在下档状态下,从键盘输入断点的四位地址,然后按一下EXAM/BP键。如还要设置下一个断点则可再送入下一个断点地址,再按一下BP键。断点设置后,即可通过EXE键来运行用户程序了。用户程序运行过程中若遇到断点,则中断程序运行,在LED上显示出断点地址和累加器A中的内容,并保护CPU内部RAM的内容到外部RAM映象区中。用户此时可通过EXE键继续程序的运行,也可以停下来检查有关寄存器或RAM的内容,然后继续运行用户程序。

监控程序断点处理分为设置断点和处理断点两部分。内部RAM 3EH单元为断点数目标识单元,初始化时清“0”,以后每设置一个断点,则同时要在此单元加1。另外,监控程序还使用了两个队列(先进先出表)。其中一个为断点表,首地址是16ECH,用来存放断点地址。另一个为原指令保护表,首地址为16F4H,用来存放用户程序中每一个断点处的原来三字节内容。这两个表的结构如图5所示。

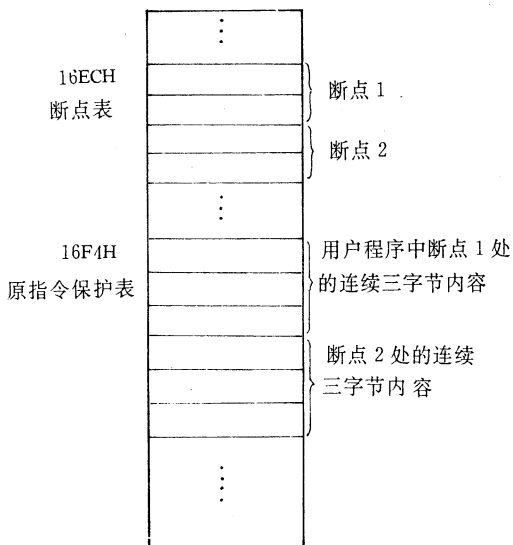


图 5

显然在设置断点时,是断点 1 先填的表,其次是断点 2,……。同样在原指令保护表中,最先保护的为用户程序中断点 1 处的连续三字节内容,其次是断点 2 处的三字节,……。而在处理断点时,是先处理断点 1,然后是

断点 2,……。这样对于断点表来说是先取出断点 1 进行处理,同时断点 2、断点 3、断点 4 依次上移两字节。原指令保护表的处理程序也是这样。可见,这两个表是典型的“队列”结构。具体过程如下:

1. 设置断点

①判断从键盘输入的是否四位地址。如果没有输满四位就按了BP键,则认为是非法,且清除原来已设置的全部断点。

②是否已建立了四个断点,如已有了四个断点则退出。

③建立断点:

• 3EH 单元增值。

• 将四位地址(断点)保存到断点表的相应位置。如原来断点表中无内容,则放在 16ECH 开始的两字节中;如原来断点表中有一个断点,则此次放在 16EEH 开始的两字节中,……

• 将断点处开始的连续三字节内容保存到断点原指令保护区中的相应位置。

• 将 LJMP BBPP(BBPP 为处理断点程序入口地址)写入到用户程序的断点处。

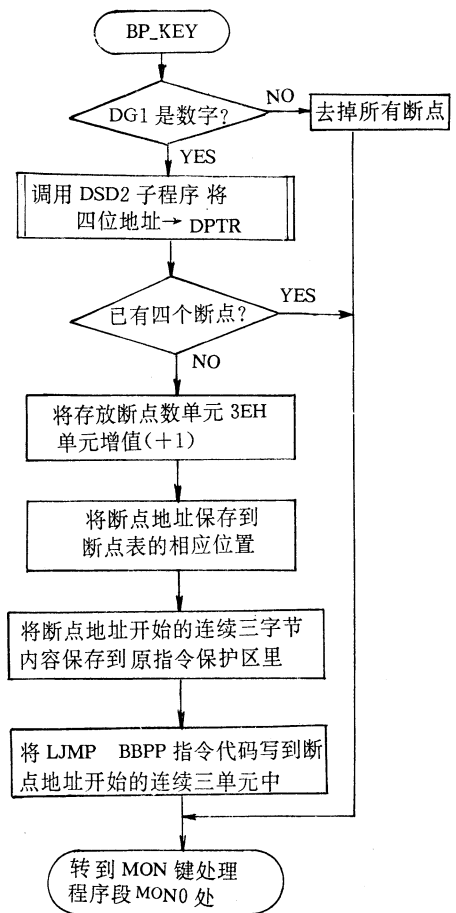


图 6

这样就能够保证多用户程序执行到此断点处时，固定地转到 BBPP 处去执行了。设置断点的程序框图如图 6 所示。

程序清单如下：

```

BP_KEY:MOV A,7EH
        JB ACC.4,BP1 ;DG1 不是数字,转走
        ACALL DSD2 ;断点地址→DPTR
        MOV A,3EH
        JB ACC.2,BP2 ;已有四个断点,转走退出
        INC A ;断点数+1
        MOV 3EH,A
        MOV R2,DPH;断点地址→R2R3
        MOV R3,DPL
        MOV R4,#0ECH ;断点表首地址低位→R4,高位为 16H
        MOV R5,#0F4H ;原指令保护表首地址低位→R5,高位为 16H

BP3:DEC A ;根据断点数来填断点表和原指令保护表
     JNZ BP5
     MOV DPH,#16H;完成填断点表操作
     MOV DPL,R4
     MOV A,R3
     MOVX @DPTR,A
     INC DPTR
     MOV A,R2
     MOVX @DPTR,A
     MOV R6,#03H ;完成填原指令保护表操作
     MOV A,R3 ;断点暂存于 R0,R1 中
     MOV R1,A
     MOV A,R2
     MOV R0,A
BP4:MOV DPH,R2
     MOV DPL,R3
     MOVX A,@DPTR ;从用户程序的断点处取出一字节
     INC DPTR
     MOV R2,DPH
     MOV R3,DPL
     MOV DPH,#16H
     MOV DPL,R5
     MOVX @DPTR,A ;将取出的一字节送入原指令保护表中
     DEC R6 ;计数器减 1
     INC R5 ;准备下次填表
     MOV A,R6 ;三字节都填表了吗?
     JNZ BP4 ;没有完成填表,转回继续
     MOV DPH,R0 ;将 LJMP BBPP 指令放入断点处
     MOV DPL,R1 ;断点地址→DPTR
     MOV A,#02H ;LJMP BBPP 的三字节操作码是:02
     ;05 44
     MOVX @DPTR,A
     INC DPTR

```

```

MOV A,#05H ;05 44 是 BBPP 地址
MOVX @DPTR,A
INC DPTR
MOV A,#44H
MOVX @DPTR,A
AJMP BP2

```

```

BP5:INC R4
     INC R4
     INC R5
     INC R5
     INC R5
     AJMP BP3
BP1:MOV 3EH,#00 ;去掉所有断点
BP2:LJMP MONO

```

2. 处理断点

如果理解了设置断点程序，则处理断点程序也就不难编制了。下面仅给出处理断点程序框图(图 7)，读者可试着自己编写。

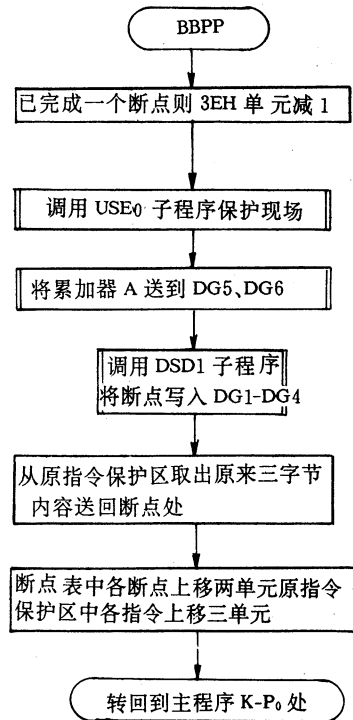


图 7

五、结束语

彻底理解监控程序是一个艰难而又有趣的工作。而修改或独立编写监控程序对自己更是一个很好的锻炼。由于篇幅所限，本文不能给出所有键的处理程序和进行更详细的说明，一些必要的子程序也没有给出，但读者如感兴趣，可与作者联系索取详细的监控程序清单和框图。

IBM-PC 与彩色电视机接口

广州三元里京粤电脑中心家用 PC 部 蒋海明 王辉

以电视机作为计算机的显示器,能大大降低微型计算机的硬件成本,为计算机进入家庭创造条件。本文介绍了一种将 IBM-PC 及其兼容机彩色图形适配卡 (CGA 卡) 输出的三基色信号和同步信号调制成我国彩色电视机标准制式 (PAL 制) 全电视信号的原理和接口电路。

一、PAL 制编码原理

为了实现彩色与黑白电视的兼容,从摄像机出来的三基色 R、G、B 信号被合成为一个亮度信号 Y 和两个色差信号 R-Y、B-Y。PAL 制采用了逐行倒相正交平衡调幅的色彩调制方法以弥补 NTSC 制色调失真的缺点。图 1 为 PAL 制彩色编码框图。

R、G、B 三基色信号首先送

到亮度色差编码矩阵电路,编成 Y、R-Y、B-Y 三个信号。Y 信号经过 0~6MHz 低通滤波器后,与复合消隐脉冲混合。R-Y、B-Y 分别经过 0~1.3MHz 低通滤波器,将频带限制在 1.3MHz 以内。由于压缩频带,引起色差信号在时间上的延时。因此,在 Y 通道中接一个 0.7μs 延时器,使亮度信号与色度信号在时间上对齐。在平衡调制前,色差信号的幅度分别压缩为

$$V=0.877(R-Y)$$

$$U=0.493(B-Y)$$

送入 V 平衡调制器的副载波需要逐行倒相,编码时除了要为接收机的解调器传送一个基准相位的信息外,还需传送一个识别倒相和不倒相的信息。所以,在 U 和 V 信号中分别送入一个 +K 和 -K 色同步信号的选通脉冲(K 脉冲),然后将混有 K 脉冲的 U 和 V 信号对副载波进行平衡调制。调制后 U 信号色同步副载波的相位为 180°,而 V 信号色同步副载波群的相位为 ±90°,实行了逐行倒相。

半行频方波 (f_H/2 方波) 控制 ±90°移相器电子开关逐行倒相,使一路 (V 路) 副载波的相位一行为 +90°,一行 (隔行) 为 -90°,即 V 信号调制在平衡副载波 ±cosωt 上。U 信号平衡调制由载波 sinωt 直接调制,这正是正交平衡调制所需求的。

两色差信号经 U、V 平衡调制后得到不倒相的 U sinωt 色度信号和逐行倒相的 ±V cosωt 色度信号,它

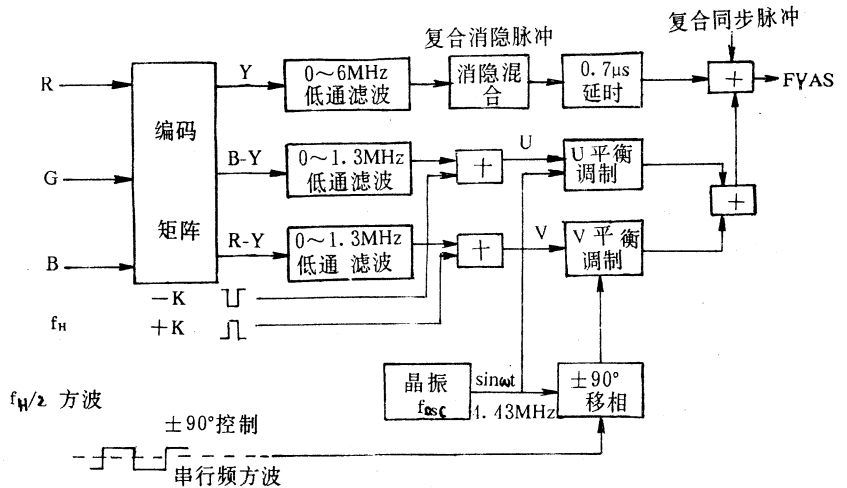


图 1 PAL 制编码方框图

们和 Y 信号在混合放大器与复合同步脉冲混合后,就形成了彩色全电视信号 FYAS。

二、RGB 转变为 PAL 编码器 MC1377

MC1377 是一种将三基色 R、G、B 和同步信号转换为全电视信号的专用集成电路。它包含了彩色副载频振荡缓冲器、电压 90°相移控制器、两个色彩载波双边频抑制调制器、RGB 输入矩阵以及消隐电平箝位 (clamp)。MC1377 只需很少的外接元件就可工作,同时又有外接引脚保证产生高质量的全电视信号,是一个三基色 RGB 转变为 PAL 全电视信号的理想编码器。图 2 即示出了 MC1377 的内部框图。

由图 2 可以看出,MC1377 的内部结构与 PAL 编码十分相似。实际上,MC1377 集成了 PAL 编码的全部功能,请读者对照图 1 自行分析它的内部结构。顺便指出,MC1377 也可以用来调制 NTSC 制电视信号。作者在这里简单介绍一下它的几个主要引脚功能:

- 1 为内部斜波发生器的外接元件端,外接电阻电容产生三角波。
- 2 为组合同步输入端,低电平有效。
- 3、4、5 分别为 R、G、B 三基色输入端。
- 6、8 分别为 -Y 的输入、输出端,在此接一个 700ns 的延时电感以保证亮度和色彩的同步。
- 9 为全电视信号输出端。
- 10、13 分别为色度信号的输入、输出端。

14 为供电电源 +12V。

15 接地。

16 为内部 8.2V 稳压输出, 供应斜波发生器的振荡元件电源。

17, 18 为 4.43MHz 晶体振荡器外接端。彩色副载波也可由 17 脚直接引入。

19 为 R-Y 和 B-Y 相角调节端。减少 19 与地的电容值(外接)将增大 R-Y 和 B-Y 的相角差。

20 为 PAL/NTSC 选择端, 接地为 NTSC 制, 悬空为 PAL 制。

表 1 列出了该集成电路的主要电气参数。

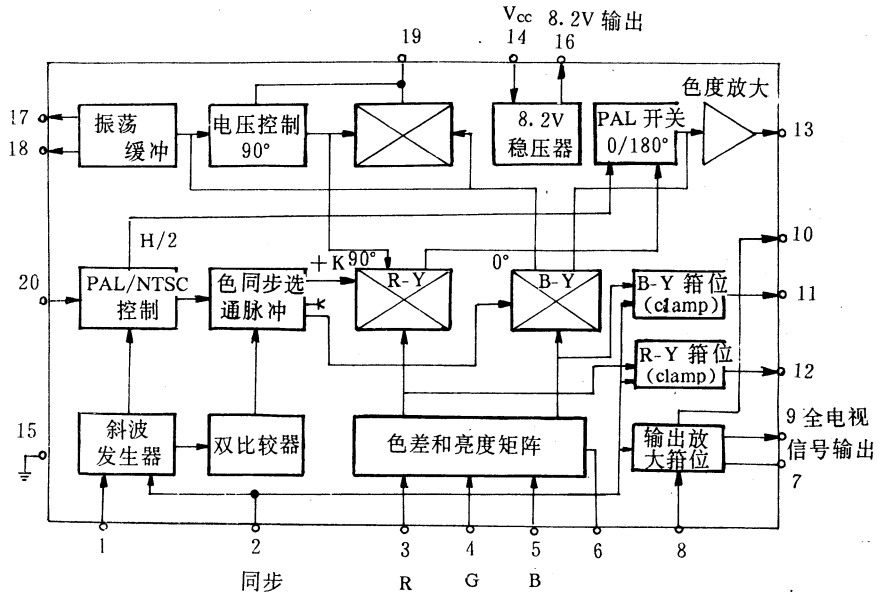


图 2 MC1377 内部结构框图

表 1 主要电气参数

性能	管脚	数值	单位	性能	管脚	数值	单位
电源电压	14	12±2	V _{dc}	同步门檻	2	1.7	V
振荡幅度	18	0.5	V _(p-p)	同步输入电阻	2	10	kΩ
电源电流	14	32	mA _{dc}	色度输出(饱和)	13	1.0	V (p-p)
外接载波输入	17	0.25	V _{RMS}	色度输出电阻	13	80	Ω
(R-Y)相角调整	19	0.25	Deg/μA	色度输入(饱和)	10	0.7	V (p-p)
RGB输入(饱和)	3,4,5	0.95~1.05	V _{p-p}	色度输入电阻	10	10	kΩ
同步峰谷消除	2	-0.5~+1.0	V _{dc}	色度输入电容	10	2.0	pF
		+1.7~+8.2				副载波输入 { 电阻 电容	17
全电视信号输出	9	0.6	V _{p-p}	RGB输入 { 电阻 电容	3,4,5		
		1.4				2.0	pF
		1.7					

三、IBM-PC 与 TV 的接口电路

图 3 示出了计算机 CGA 卡输出信号转变为全电视信号的实际应用电路。

MC1377 的 RGB 输入应设在 1.0V 峰峰值(色饱和时), 这个值是不能任意变化。CGA 卡过来的 5.0V p-p RGB 信号经过 R₁、R₃、R₅ 与 R₂、R₄、R₆ 电阻分压, 再经电容 C₁、C₂、C₃ 的耦合加到 MC1377 的 RGB 输入端。CGA 卡的行场同步信号经 74LS266 异或非门组合成复合同步信号, 低电平有效。由于 266 是集电极开路输出, 接提升电阻 R₇ 以保证 +5V 的高电平输出。在场

同步输入端接了一个电容 C₄, 为的是防止外界信号对同步输入的干扰。

为了实现亮度和色度同步, 在 6、8 之间加了一个 700ns 的亮度延时线圈, 选用彩电的亮度延时线圈基本上能满足同步要求。10、13 之间的色度耦合采用了简单的阻容耦合。若能用 TOKO166NNF10264AG 滤波器耦合色度信号效果更佳。17、18 端接 4.43MHz 晶振, 微调电容 C₁₁ 用来微调彩色载波频率, 以求得到稳定的彩色图象。9 脚输出的全电视信号可直接接电视机的视频输入, 或经过 RF 调制器产生射频信号接电

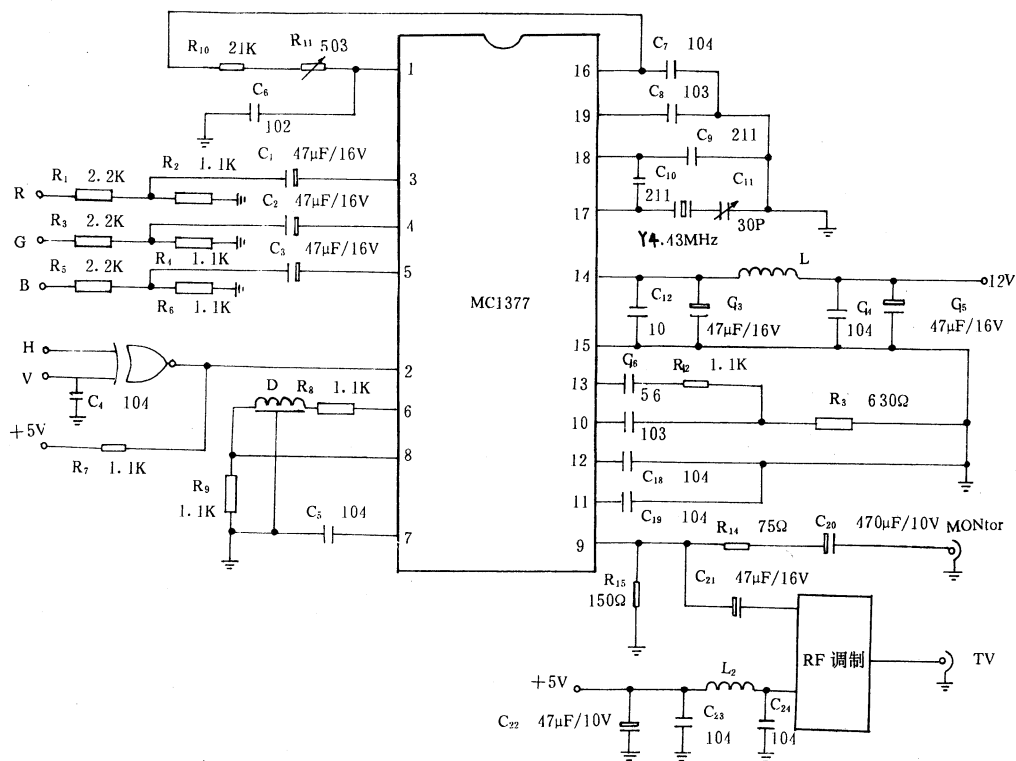


图 3

视机的天线输入。按以上电路参数安装的电路板,不经调试就能得到黑白图象,适当调整 R_{11} 和 C_{11} 值,就能得到彩色图像。

RF 调制器可选用游戏机用 RF 调制器,在电子游

戏机配件商店可以买到。或者也可以选用 Motorola 公司生产的 RF 专用芯片 MC1373,外接几个简单的元件就可实现 RF 调制。应用电路如图(4)所示,感兴趣的读者不妨一试。

四、结论

根据图 3 电路安装调试的 PAL 接口卡,基本上能得到满意的彩色效果。由于在彩色副载频(4.43MHz)附近没有将亮度信号滤去,彩色图象中有微弱的亮度重影。此重影可通过加强电视机对比度,减小亮度和色度来消除。该接口卡已在广州京粤汉字电脑中心最新推出的系列家用电脑产品中得到应用。有兴趣的读者可与我们一起研究、探讨有关这方面的问题。

联系地址:(510400)广州三元里京粤汉字电脑技术研究开发中心家用 PC 部 王辉;

电话:(020)6624960—32。

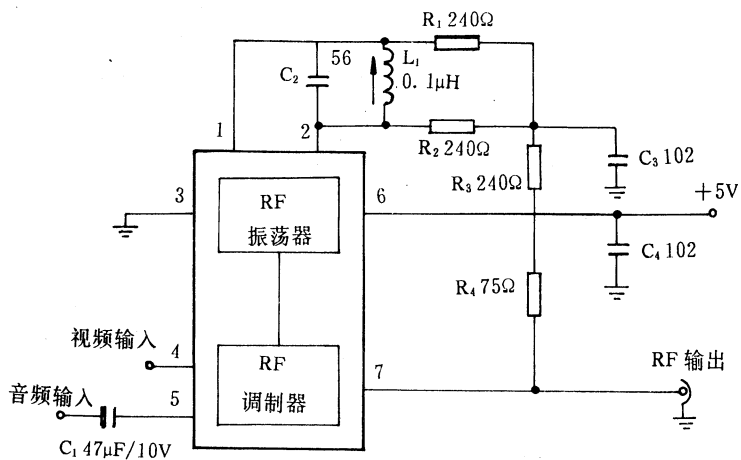


图 4 RF 调制专用 IC MC1373 框图及应用电路

彩色显示器工作原理简述及故障诊断(中)

中国人民大学信息中心(100872) 胡野红

三、行扫描电路出现的故障

由于行扫描电路长期工作在高电压、高频率、大电流的环境中,所以行扫描电路的故障率在显示器故障中占有最大比例。

行扫描电路出现的典型故障现象为垂直一条亮线。由行扫描电路工作原理可知,当摆放在显像管脖颈处的一对行偏转线圈中有锯齿波电流通过时,据“电工原理”中的左、右手定则可知,将产生和锯齿波电流频率相同的均匀变化的水平磁力(在“普通物理”中也称为洛伦兹力)。电子束在此力的作用下,在屏幕上完成从左到右的扫描。垂直一条亮线故障表明显像管阴极发射的电子束经过偏转点时没有受到水平磁力的作用而仅受到垂直磁力的作用。没有水平洛伦兹力表明行扫描电路有故障。

以下几例是行扫描电路故障的典型表现。

例 1:故障机型号:Samsung 牌 SVGA 彩显

故障现象:面板电源指示灯亮,但无光栅

分析诊断:不同牌号的彩显虽然分辨率、扫描方式各有不同,但工作原理却大同小异。自激式扫描电路即使微机主板不送 TTL 信号到显示器,若显示器正常时,屏幕上也有正常亮度的光栅出现,而它激式则相反。所以搞清该机是自激工作方式还是它激工作方式很重要,否则将误入歧途。本例彩显为自激式。无光栅说明扫描电路没有正常工作,图 4 为该机的行扫描电路图。由图中可以看出,LA7851 输出的行、场同步信号分别送到由 Q503 组成的行输出电路和 LA7830 场输出电路。

首先测该机的电源电压,发现插座 J834 红端为 2.5V,桔红端为 5V,怀疑电源有问题。本电源由两块相同的厚膜集成电路 STK7404 组成有两路输出的开关电源。针对本电路中厚膜集成电路易损的特点,用正品 STK7404 分别对两路输出中的厚膜作替换试验,

但故障现象依旧。仔细检查未发现电源工作异常,电源损坏引起的故障可以排除。拔掉场、行偏转线圈进行模块隔离检查时发现,J834 红端电压由 2.5V 上升到 125V,桔红端仍为 5V。由此证明,肯定在场、行输出电路中有短路存在,致使电源中的一路输出电压急剧下降。场、行扫描电路中偏转线圈发生短路的现象很少发生,而且偏转线圈电阻值正常。偏转线圈短路引起电源电压下降的可能性可以排除。

由于场、行输出管功率大,损坏的可能性也大。所以在排除了电源和偏转线圈损坏的可能性后,应重点检查行管 Q503(2SC3486、1500V、6A)。结果是 Q503 的 ce 结短路,D502(1000V 1A)烧糊,换上相同参数的管子后,显示器正常工作。

从以上过程可以看出,由于行管 Q503 和 D502 短路,使电源执行保护,输出电压大幅度降低,场行扫描电路不能输出锯齿波电流,因此场行偏转线圈就不能产生垂直和水平磁力,屏幕上就出现不了扫描光栅。又由于该机电源有两路输出。一路供场、行扫描,另一路供其它模块。当行、场出现故障电源执行保护时,另一路仍有 5V 输出(正常时为 20V)所以故障机面板电源指示灯仍亮,只不过是亮度有差异。(正常时,J834 红端工作电压为 60V,桔红端为 5V)。

例 2:故障机型号:IBM5550 显示器

故障现象:能够正确地显示字符。但字符上充满毛刺,并且左右摇晃。整个屏幕光栅忽大忽小闪动。

分析诊断:该显示器是单色汉字终端。据故障现象看,肯定是在扫描周期内有瞬间短路存在。可能是:①电源问题;②行管及其周围电路的问题;③显像管电路问题。

本显示器电源应输出 72V 直流,用万用表在线实测证明符合技术指标且平稳。从故障现象看,光栅忽大忽小,不排除电源功率不足的可能性。因为行扫描电路在每个周期内的不同时刻输出功率不同。当在周期内功率高峰时,电源输出功率不够,而在功率低峰时能满足需要,所以造成屏幕光栅闪动。据行扫描电路的这个特点,断开电源输出端,接上一定功率损耗的假负载,且逐渐增大,未发现电源有问题。又用一台好的 5550 显示器上的电源作替换试验,(注意做这个替换要特别小心,一般应把怀疑损坏的部件拆下,

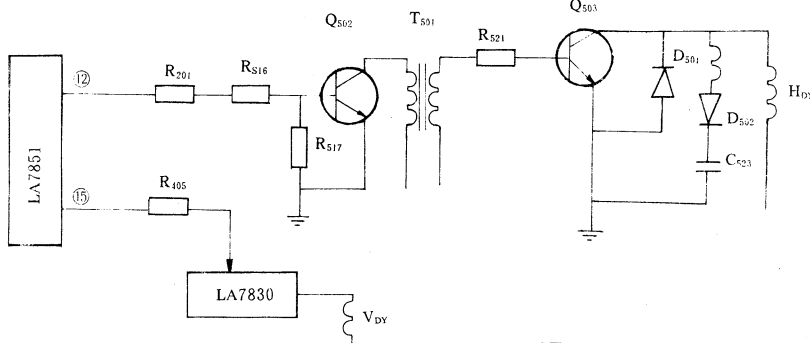


图 4

把好的部件放在被怀疑的机器的相应部位进行替换实验,而不该把怀疑有问题的部件放到好机器上去试,以免坏部件损坏好机器,使故障范围扩大)结果故障依然存在。事实证明电源没问题。电源在扫描周期内瞬间输出功率变小的可能性可以排除。

行扫描电路出现问题也可能引起此故障。如行管负载能力变差或耐压变低、行偏转线圈周期性瞬间短路等。用替换法对行管、阻尼管、偏转线圈等元器件进行替换,故障依旧。用示波器进一步检查行输出波形的幅度、频率,未发现异常现象。可以肯定,行扫描电路无问题。

显像管电路出现问题也可能引发此故障。实测显像管尾板管脚电压,除⑥脚(阴极)电压不随屏幕晃动而变外,栅极电压、灯丝电压随屏幕晃动的频率做不同幅度的变化(阳极电压一定也变,只不过是一般情况下不能定量测量而已)。可以肯定故障出现在显像管电路附近或本身。可能是显像管源电路发生问题,也可能是显像管座内极间瞬间短路。经检查后者无问题。据长期工作经验,显像管本身出现故障的概率极低,所以这种设想可以排除。剩下的只能是给显像管提供工作电压的高压包了。

高压包给显像管阳极提供近万伏的电压以吸引阴极发射的电子束迅速到达荧光屏。同时,栅极电压、灯丝电压均来自高压包。所以高压包工作电压高、负载重,发生故障的可能性也大。用替换法一试,果然不出所料。

从以上分析检测的结果可以看出,由于高压包次级绕组发生周期性瞬间短路,也就是说在行振荡脉冲的高峰时刻,高压包次级绕组被击穿,造成供给显像管电路的各路电压下降,阳极和栅极对阴极电子束的吸引能力发生周期性的瞬间降低,致使光栅忽大忽小晃动、字符充满毛刺。

行输出管的集电极接高压包初级绕组,当高压包发生周期性的瞬间短路时,由于高压包的隔离作用,行管只不过是瞬间加大了一些负载而已。所以当用示波器观看行管集电极波形时,幅度、频率均正常。其次上述周期性瞬间短路发生过程极短,只是在脉冲高峰时

发生,当高峰期过后,短路又不存在了,所以此短路电流均值很小,在高压包初级没有明显反映,波形正常仅仅是行管电流略有增加而已。

现在常见的高压包均采用一体化技术制造,即初级、次级和升压电路全被密封在耐高温高压的塑料壳内、上例中到底是次级绕组中匝间瞬间短路还是绕组间短路因不便打开高压包细看,只能依据高压包损坏(初级波形、电压正常、而次级不正常)的事实分析推测了。

例3:故障机型号:BRIUP牌CGA彩显

故障现象:屏幕无光栅、电源指示灯不亮,有间断的吱吱叫声。

分析诊断:从故障现象看似乎机内电源有问题。该机标称电压为两档,分别为110V(红引线)和60V(棕引线),实测均为零。断开电源输出端接上假负载,输出电压达到标称值。以上事实说明电源无问题。

仔细检查线路,发现最大负载行管集电极连接110V电源。断开电源连线,测集电极对地电阻,其值为零,证明有明显短路存在。因为行管输出端和行偏转线圈、高压包初级接在一起,他们当中的任一个发生短路均会影响行管集电极。所以分别断开行偏转线圈、高压包初级测量,行管集电极对地电阻仍为零,问题明显在行管本身。焊下行管2SC1942测量,行管未损坏,再仔细检查,发现散热片和和行管之间的云母垫片有一个小洞,换上相同规格的云母片并用硅胶粘合好,重新固定在散热片上,焊好连线再测,短路现象消失,上电试验,显示器工作正常,电源输出端的电压为标称值。

以上事实说明,由于行管和散热片之间的云母垫片击穿,致使行管集电极对地短路,使电源执行保护出现上述故障现象。本机电源保护功能可靠,否则,此故障也可能引起电源损坏,使故障范围变大。

从例2、例3可以看出,当其它模块发生短路后引起电源保护时,由于保护的方式不同,技术指标不同,前者电源电压迅速下降,但不为零,而后者为零,这点请读者注意。

(待续)

电源引起的特殊故障一例

武汉中南民族学院计科系(430070) 王 斌

故障现象:本单位一台IBM系列兼容机,开机后,自检正常,自检完毕后,硬盘指示灯一闪,随即熄灭,然后进入死机状态。

故障分析与检修方法:大家知道,硬盘的引导过程是当机器上电或复位时,CPU进入复位状态,并置

寄存器CS的值为FFFFH,IP值为0000H。当自检程序检查有硬盘ROM BIOS时,转去执行硬盘ROM BIOS初始化程序,INT 19H的向量改变,指向其中的硬盘引导装入程序硬盘ROM BIOS中的引导装入程序把硬盘0柱0磁头1扇区的主引导程序装入到内存

0000:7C00H,并转移到该处执行主引导程序,主引导程序检查4个分区的引导标志,把唯一一个引导标志是80H的分区即PC-DOS分区引导程序装入内存0000:7C00H处,并转移到该处执行PC-DOS分区引导程序,PC-DOS分区引导程序把根目录区第一扇区内容读入内存0050:0000H处,并检查第一、第二根目录是否含有IBMBIO.COM、IBMDOS.COM两个隐含系统文件(对于MS-DOS系统是名为IO.SYS、MSDOS.SYS的两个隐含文件),若有两个文件就把第一个系统文件IBMBIO.COM装入0070:0000H处,并从该处执行IBMBIO.COM程序。由IBMBIO.COM程序依次调用IBMDOS.COM文件和COMMAND.COM文件的常驻内存部分。COMMAND.COM文件中的初始化程序调COMMAND.COM常驻内存部分,COMMAND.COM的常驻内存部分处理CONFIG.SYS和AUTOEXEC.BAT文件(若有这两个文件的话),然后显示系统提示符。这样硬盘启动完毕,系统引导成功。根据故障现象初步判定为与硬盘有关的故障。因此,为缩小故障范围,于是断开连结硬盘的电缆。用软盘引导系统,开机后自检正常,读软盘,指示灯亮,一会儿熄灭,软盘引导DOS系统失败。这样,似乎故障范围不局限于硬盘及周围电路有问题。为缩小故障范围,拔出软、硬控制卡后,重新开机,可引导ROM BASIC,这说明CPU控制总线时系统板工作正常,且说明内存刷新工作也正常即DMA应答

电路也是好的。而此故障特征为驱动器灯亮,马达转动且读盘灯亮,过后灯灭,磁盘引导失败。由于软驱卡、软驱故障以及8237 DMA控制器及周围电路故障均可引起此故障现象,于是通过交换法检查软盘驱动器及卡均无问题,8237 DMA控制器及周围电路也无问题。通过仔细观察分析,发现软盘驱动器马达转速比正常转速要慢,这样,故障原因逐步引向主机电源供电系统上来,用电表测量电源输出端各档电压有所降低。拆开电源检查,此电源电路为自激式脉宽调制变换器型稳压电路,发现电路板及元器件表面无损坏痕迹。通电后,不接负载时,测量电源输出端各档电压,基本正常。对于引起电源输出欠压的主要原因引起:1. 整流滤波电路中滤波电容性能变差或整流二极管性能不好造成电流过大;2. 功率开关管增益偏低,从而振荡脉冲幅度减小;3. 输出电路中某一路滤波电容漏电或整流特性不好;4. 稳压控制电路或限流保护电路的取样电阻或基准电压分压电阻有故障。通电动态检查各个可能发生问题的元件,发现两个串联滤波电容两端实际直流电压小于正常值(300V)。焊下滤波电容检查,发现有一个滤波电容漏电严重,更换后,通电,测量输出端电压正常。接负载后,电源输出端无欠压现象,重新装好机器,机器工作正常。由此可见,滤波电容性能变差,使电路滤波性能变差,输出功率减小,造成主机电源负载能力差,导致机器出现上述故障。

对《单片机旅馆客房门卫系统》中P₂口使用的异议

大连石油化工公司仪表车间(116031) 秦化勃

《电子与电脑》91年7期《单片机旅馆门卫系统》一文,笔者对P₂口的使用有不同的看法。作者将P₁口的0、1、2、3位用作位扫描,将P₂口的4、5、6位作键输入口。这里有一个问题值得探讨:

该文采用了单片机最小应用系统,P₂口的0、1、2、3位作为2732的高四位地址线,虽然P₂口仅使用了四根线,但空余的四位已不宜作通用I/O线。笔者在研制某产品时,亦想利用P_{2.4}、P_{2.5}作输入口,曾编制一段小程序、并做了一台样机(样机亦采用单片机最小应用系统,程序存储器采用EPROM2732)用于调试、检验P_{2.4}能否起输入口作用。

用DSG-51-I仿真器联样机调试。正常情况下,如P_{2.4}口作输入口并能检测出信号,程序执行到8107处应向下执行810A指令。但实际调试时,程序却跳转到8111处,这说明SETB P_{2.4}指令未起作用,P_{2.4}未被置1。试将P_{2.4}换成其它I/O口(如P_{1.n}或

P_{3.n}),再执行程序,则一切运行正常。

因此,在P₂口用作程序存储器高位地址总线时,我们应避免使用剩余的其它P₂口线作通用I/O口,否则会给软件编写和使用上带来相当多的麻烦。

程序段如下:

```

8100:79 00          MOV    R1, #00H
8102:C2 B0 SEP3.0: CLR    P3.0
8104:09           INC    R1
8105:D2 A4         SETB  P2.4
8107:30 A4 07     JNB   P2.4, AM0
810A:B9 02 F6     CJNE  R1, #02H SEP3.0
810D:78 00        MOV    R1, #00H
810F:D2 B0        SETB  P3.0
8111:D2 20 AM0:   SETB  20H
8113:A2 A4        MOV    C, P2.4
8115:B3           CPL    C
8116:92 B0        MOV    P3.0,C

```


第五讲 游戏程序的编程特技(下)

山东苍山机械电子化学工业局(277700) 于 春

三、结合特技

1. 字符显示与音响的合成

诸如“人间兵器”、“恶魔城传说”、“激龟忍者传”等游戏,在打印字符提示的同时,伴随着滴滴的音响,把字符逐个打印在屏幕上,使字符显示新颖活泼。现在介绍字符打印与音响合成的方法。

如我们欲在屏幕中开一个横座标 6~22、纵座标 4~15 的窗口,窗口内打印如下一段文字:

```
Wind and rain escorted spring's departure,
Flying snow welcomes spring's return.
On the ice-clad rock rising high and sheer,
A flower blooms sweet and fair.
```

(毛主席诗词“风雨送春归,飞雪迎春到。已是悬崖百丈冰,犹有花枝俏。”)

则可以编制如下程序实现。见程序 No. 5-18。

```
5 REM No. 5-18
10 CLS
20 PL. "T1M1V5Y2"
30 DIM A$(10)
40 F.I=0 TO 10
50 IF I MOD 3=0 T. A=A+1
60 LOC. 6,4+I+A;READ A$(I)
70 F. J=0 TO LEN(A$(I))
80 C$=ML. (A$(I),J+1,1)
90 P. C$;:PL. "O3G2"
100 N.
110 PAU. 50;N.
120 PL. "O3E2DCDE";LOC. 0,22;E.
130 D. "Wind and rain es-", "corted spring's", "depar-
ture,", "Flying snow welc-", "omes spring's", "re-
turn."
140 D. "On the ice - clad", "rock rising high", "and
sheer,", "A flower blooms", "sweet and fair."
```

程序中引入 50 行,是为了在打印时,每打印一句空一行,以便于阅读。

2. 背景图形与程序的合成

有些游戏软件直接把背景图案写在语句中。这样做的优点是:①程序直观,便于阅读理解;②可极大地

简化绘图程序;③可提高背景图形的打印速度;④便于背景图形整块移植;⑤在游戏换场中,可把每一场的图形装入一组下标串变量中,换场时用一循环语句打印,可迅速完成背景换场。但是,我们知道,背景图形只能通过使用 PRINT CHR\$($\times\times\times$)语句打印在屏幕上,怎么能写入程序中呢?

实现的方法如下:

1)用 BGTOOL 语句调出 BGGGRAPHIC 绘图画面,进行键控绘图。

2)绘图完毕,先按“ESC”键,再按“STOP”键返回 BASIC 状态。

3)用 VIEW 语句调出绘制的画面。

4)把画面中每一行的图形作为一个字符串装入字符串下标变量中。一般每一场画面用一个串数组。如 A\$(20)、B\$(20)等。每一行前面加入一个行号。

5)在 0~20 行图形均装入程序中后,当调用该场画面时,只要使用

```
FOR I=0 TO 20:P. A$(I);N.
```

就可立即打印出背景画面。

采用这种方法绘制背景画面。一般可节约内存 60%,绘图速度提高 20 倍。

在使用背景装入程序的方法时要注意以下四点:

1)当用 VIEW 调图形时,可能出现电脑语句执行完毕提示符“OK”破坏画面的情况。这时可用 LOC. X, Y;VIEW,把“OK”移到其它位置。

2)对某行开始(即横向 0 位)就有图形的情况,可用“Ins”键移出打印行号和变量名及等号的空位。

3)每行图形的开始写入位置要统一,否则将破坏画面的完整统一。

4)使用这种方法绘图时,注意不要使用图形库中前四组的图形。

3. 卡通与背景的合成

在“敲冰块”、“冒险岛”、“超级玛丽”等游戏中的主人翁可在操纵器的控制下,跳上台阶,越过深涧、爬上天梯、跃下高山悬崖、在背景画面中蹦跳随意,运动自如。这种效果就是卡通运动与背景图案巧妙结合而产生的。欲产生这一效果,在卡通的每一个动作之前,首

先要检查卡通的立足点是什么背景,其次再检查在其动作方向上一步位置又是什么背景,根据两次检查情况做出恰当的处理。如卡通的立足点是天梯,其动作方向前一步位置也是天梯,则可执行方向键的上、下命令而作上、下楼梯的动作;若这时按了左、右运动键,而左、右方向又都是空间,则卡通只能做跃下的动作,左跃下或右跃下;若卡通的立足点是平地,其前一步也是地,则可执行方向键的左、右运动命令,向前走一步;若这时方向键是向上或向下,而在向上一歩又没有梯子,则必须令卡通原地不动。总之,在背景中卡通的运动已不再是机械地执行操纵器的指令,而是具有一定的人工智能判断,根据实际情况,命令合理则执行,命令不合理则拒绝执行。下面介绍卡通与背景通用合成程序 No. 5-19。该程序是一个通用的工具程序,只要配上适当背景就可以操纵卡通进行奔跑、散步、跳起、跃下、升高、降低、探险取宝等各种动作。

```

5  REM No. 5-19
10  CG. 1,0;ON ERR. G. 265
20  PAL. B 3,13,48,54,22
25  SP. O.
30  DE. SP, 0, (0, 1, 0, 1, 0) = CH. (25) + CH. (24) + CH.
    (27) + CH. (26)
35  K $ = CH. (12); R $ = CH. (13); H $ = CH. (14); U $
    = CH. (27)
40  DE. SP. 1, (0, 1, 0, 1, 0) = R $ + K $ + U $ + H $
45  DE. SP. 2, (0, 1, 0, 0, 0) = K $ + R $ + H $ + U $
50  DE. SP. 3, (0, 1, 0, 0, 0) = CH. (20) + CH. (21) + CH.
    (22) + CH. (23)
55  R $ = CH. (215); H $ = CH. (210); U $ = CH. (192);
    DIM Q(3), D(3)
60  X = Y = V = XL = YL = S = S1 = S2 = K = G; S $ = G $
70  X = 116; Y = 127; POS. 7, X, Y, V. ; G. 170
80  S1 = S; S = STI. (0); IF S > 2 T. S = 3 - (S > 7)
90  X = XP. (7); Y = YP. (7)
95  XL = X/8 - 1; YL = (Y - 2)/8
100  ON S G. 175, 175, 140, 155
105  V = 0; K = 0; CUT 7
110  IF SCR. (XL, YL) = " " T. IF SCR. (XL, YL - 1) <>
    H $ T. GOS. 260; G. 210
115  IF STR. (0) <> 8 T. 80
120  IF SCR. (XL - 1, YL) = U $ T. 215
125  IF SCR. (XL, YL) = U $ T. 215
130  IF SCR. (XL + 1, YL) = U $ T. 215
135  G. 80
140  V = 5; G $ = SCR. (XL, YL)
145  IF G $ <> H $ T. IF G $ <> U $ T. S = 0; V = 0; G.
    210
150  G. 160
155  V = 1; G $ = SCR. (XL, YL - 1)
160  IF G $ <> H $ T. S = 0; G. 105
165  IF S1 = 0 OR K > 0 T. GOS. 260; K = 0
170  DE. M. (7) = SP. (0, V, 1, 4); M. 7; SP. S2; PAU. 5; G.
    80
175  V = 4 * S - 1; K = 1 - (K + (K > 3)) * (S = S1)

```

```

180  G $ = SCR. (XL + 1 + 2 * (S = 2), YL); S $ = SCR.
    (XL, YL)
185  IF G $ <> U $ AND G $ <> H $ T. X = X + 6 * (S $
    <> " ") * ((S = 1) - (S = 2)); G. 210
190  IF S $ = " " T. S = S1; G. 210
195  IF STR. (0) = 8 T. IF G $ = U $ OR S $ = U $ OR
    SCR. (XB, YL) = U $ T. 215
200  IF K < 4 T. DE. M. (7) = SP. (0, V, 4 - K, 100); M. 7;
    SP. S2
205  G. 80
208  SP. S; SP. S1. ; SP. S2; ERA. 7
210  G = 2; Y = Y + 3; G. 220
215  G = -9; X = X + (2 * K + 6) * ((S = 2) - (S = 1)); Y =
    Y - 12
220  SP. S, X, Y; ERA. 7; SP. 3; POS. 7, X, Y; S2 = S; K = K
    * ((S = 2) - (S = 1))
225  G = G + 3 + 3 * (G > 6); X = X + K; Y = Y + G; SP. S, X,
    Y; POS. 7, X, Y
230  XL = X/8 - 1; YL = (Y - 2)/8; G $ = SCR. (XL, YL); S
    $ = SCR. (XL, YL - 1)
235  IF G $ = U $ T. GOS. 255; K = 1; G. 170
240  IF S $ = H $ T. GOS. 255; SP. 3, X, Y; SP. S2; S2 = 3;
    G. 80
245  IF G $ = H $ T. GOS. 255; V = 0; G. 170
250  G. 225
255  GOS. 260; K = 0; SP. S, X, Y; RE.
260  CUT 7; X = XL * 8 + 12; Y = YL * 8 + 7; POS. 7, X, Y;
    RE.
265  IF ERL = 180 T. XL = -27 * (XL < 1); GOS. 260; M.
    7; RES. 200
270  IF ERL = 110 T. XL = -27 * (XL < 0); GOS. 260; S =
    S1; RES. 175
275  IF YL > 23 T. RES. 208
280  IF XL > 27 T. XL = 0; GOS. 260; RES.
285  IF XL < 0 T. XL = 27; GOS. 260; RES.
290  IF XB > 27 T. XB = 0; M. 7; RES.
295  IF XB < 0 T. XB = 27; M. 7; RES.
300  IF XL + 1 > 27 T. XL = 0; GOS. 260; RES.
305  IF XL - 1 < 0 T. XL = 27; GOS. 260; RES.
310  IF X < 12 T. X = 228; RES.
315  IF X > 228 T. X = 12; RES.
320  E.

```

程序说明:

1) 10~70 行为初始化赋值程序。其中 30 行定义玛丽跌下, 40 行定义玛丽面向右跳, 45 行定义玛丽面向左跳, 50 行定义玛丽背面, 55 行选 192 号背景为地面或台阶, 210 号为天梯, 215 号为宝贝(用于采宝得分)。

2) 80~110 行为读方向键程序。

3) 115~165 行为按 A 键(跳跃)后的有关处理和按上、下键的处理。

4) 175~205 行为按左、右键的处理程序, 同时考虑按 A 键的情况。

5) 210~250 行为上、下动作或跳跃后的处理。其中 210 行令卡通下移; 215 行令卡通跳起, 若同时按

左、右键则在 X 方向产生位移;225 行令卡通下移一次后再下移 5 点或者跳起后降落时分次降落,每次降 3 点。

6)265~315 行为出错处理程序。

7)170 行确定 7# 动作的上下运动方向和动作启动。

8)255、260 行为确定运动卡通起点座标程序。

9)200 行确定 7# 动作左、右运动方向和动作的启动。

为压缩程序量,该程序采用顺序结构编写,程序的执行过程如下:

程序运行后,首先检测 1# 操纵器方向键,若按左、右键则转 175 行,根据键值确定运动方向,对横向运动标志 K 赋值,检查 7# 动作立足点和前一位的背景,若均为地则执行 200 行,启动 7# 动作移动一步。若有一处不是地,则令 7# 下跃。下跃一次后测试落脚点是否为地,若是地则转 170 行,按指挥方向运动,否则,继续下跃,直到落脚点是地为止。

第二、若按方向键的上键则转 155 行,置运动方向 V=1。检查落脚点的上一位是否有梯子,若有则上移一步。否则原地不动。

第三、若按方向键的下键则转 140 行,置 V=5。检查落脚点是否有梯子,若有则下移一步。否则不动。

第四、若未按方向键而按 A 键,若落脚点和其前、后点都是地,则转 215 行令其跳起后再下落。

第五、若按方向键的同时又按 A 键,则通过 215 行在向上跳起的同时在 X 方向产生位移。

程序中引入变量 G 的目的是为了使 7# 动作的下落动作分段进行,逐渐下落到地。注意——程序 No. 5-19 只有配上背景后才能使用。读者欲演示该程序可用 BGTOOL 语句调键控绘图状态,使用 192 号背景画地面、台阶,201 号画梯子。背景设计好后,返回 BASIC 状态,运行本程序即可演示。

4. 动态背景与卡通的合成

本讲的第一部分介绍了动态背景的形成。现在介绍动态背景与卡通的合成技巧。演示程序见 No. 5-20

5 REM No. 5-20

10 CLS:SP.O.:CG.0,1

20 PL.“V15Y2T2;V15Y3T2”

30 DE.M.(7)=SP.(7,1,1,1,0,2):POS.7,80,40:M.7

40 DIM A\$(15):A\$(15)=CH.(199):B\$=CH.(244)

50 F.I=0 TO 14:A\$(I)=CH.(207+I):N

60 LOC.0,21:P.A\$(RND(16)):B\$

70 IF D=2 T.LOC.RND(15)+2,22:P.A\$(RND(16)):D=0

80 LOC.17,23:P.B\$;A\$(RND(16))

90 T=T+1:IF T MOD 10=0 T.LOC.20,20:P.“SC=”T
“ ”

100 IF T=1000 T.PL.“O3E0DCDE”:ERA.7:CLS:LOC.8,10:P.“VERY GOOD!”:LOC.0,20:E.

110 D=D+1:S=0

120 K=STL.(0):IF(K=1)+(K=2)=0 T.160

130 IF K=1 T.S=3

140 IF K=2 T.S=7

150 IF F=1 T.F=0:G.190

160 PX=(XP.(7)-16)18

170 L\$=SCR.(PX-1,2):R\$=SCR.(PX+1,2):F\$=SCR.(PX,2):

180 IF L\$<>“ ”OR R\$<>“ ”OR F\$<>“ ” T.F=1:G.210

185 IF S=0 T.195

190 DE.M(7)=SP.(7,S,1,4,0,2):POS.7,PX*8+16,1:M.7

195 PL.“O2#G0”

200 G.60

210 PL.“O1E1C;O0#E1C”

220 T=0:LOC.20,22:P.“AGAIN!!!”:LOC.20,20:P.“SC= 0”:G.60

运行程序,屏幕左侧显示一条太空通道,一艘太空船在太空通道上端发出滴滴的声音飞速向下降落,各种形状的太空陨石从飞船两边飞过,犹如飞船钻入陨石雨中。游戏者用 1# 操纵器控制飞船左右移动,躲避陨石的撞击,计数器记录着飞船降落距离。若降落 1000 千米没有与陨石碰撞则降落成功,打印贺词,停机。若降落过程中发生碰撞,则计数器从 0 开始。程序中引入变量 D 是为了控制陨石密度。D 值越大,陨石越少。

本程序短小精悍,自成游戏,游戏过程激烈有趣,富有冒险性。若再加入标题、音乐、开始、结束画面,则能构成一个完整的游戏程序。游戏名称可定为“火星探险”,留给读者练习。

(接 50 页)

灵科 MIS 管理新工具

程序结构管理及菜单生成系统 零售价:500 元

辅助生成结构图,程序文档;自动生成 WINDOWS、C 语言、dBASE 菜单源程序;支持鼠标操作,模拟显示。

程序流程图管理系统 零售价:500 元

支持顺序判断(IF..ELSE)、循环(WHILE)、选择(SWITCH)逻辑结构图及相应文档;自动生成 C 语言逻辑调用程序;支持鼠标操作;支持 6 层嵌套。

经 销:中国电子器材华北公司

地 址:北京市海淀区万寿路西街五号

通信地址:北京市 144 信箱

邮政编码:100036

联系人:石立军 魏 国

电 话:81.1810 81.0920

本公司经营计算机及外设通信设备,欢迎来电函联系。

实用程序——彩色电视信号发生器

福建霞浦城关西北角 8 号(355100) 王长宾

本程序可产生如下电视信号:白光栅、红基色光栅、蓝基色光栅、绿基色光栅、彩条、黑白棋盘格(两种)、文字符号、伴音等信号,是修理调整电视的理想信号源。几种光栅及彩条信号可用于通道、彩色解码及色纯等故障的检修与调整,棋盘格信号可用于调整场线性、场幅及行线性、行幅,三重电子演奏的音乐伴音信号,主要用于伴音通道的检修,由于输出的文字符号笔划较细,所以文字符号信号可用于检修图象中放及视放部分的高频特性,以及聚焦等的调整。本程序设计用游戏机的主手柄按键控制这些信号,调用方便,按十字键上为绿基色,下为红基色,左为蓝基色,右为白光栅,选择(SELECT)键为加伴音,启动(START)键为加文字符号,A 键为彩条,B 键为棋盘格。棋盘格信号有两种,当按下 B 键出现第一种棋盘格,在该棋盘格全部显现(约 1.5 秒)前持续按 B 键,将显现第二种棋盘格。如果按 B 键不放,这两种棋盘格将交替出现。第一种棋盘格用于国内制式的电视机调整,第二种棋盘格可用于国内制式电视机改制时的调整。目前许多进口的录像机可放 NTSC 录像带,但输出的信号却为仿 PAL 信号,国内制式的电视机不能正常收看(主观表现为场频、场幅不同),改制时,可用第二种棋盘格来调整场幅,但不能调整场频。

程序清单(注意数字“0”与英文字母“O”的区别)。

```
5 CLS
10 S=STICK(0);T=STRIG(0)
15 IF T>0 THEN 70
20 IF S=0 THEN 10
25 IF S=1 THEN 45
30 IF S=2 THEN 50
35 IF S=4 THEN 55
40 IF S=8 THEN 60
45 CLS;PALETS 0,16,0,0,0;GOTO 10
50 CLS;PALETS 0,18,0,0,0;GOTO 10
55 CLS;PALETS 0,22,0,0,0;GOTO 10
60 CLS;PALETS 0,26,0,0,0;GOTO 10
70 IF T=1 THEN 100
75 IF T=2 THEN 200
80 IF T=4 THEN 300
85 IF T=8 THEN 400
100 FOR A=32 TO 255
105 PRINT CHR$(A);
110 NEXT
115 GOTO 10
200 PLAY "M0V14Y3T4;M0V10Y2T4;M0T4"
205 PLAY "O2G6G3A5O3C;O2E6E3F5A;O1C7F"
210 PLAY "D4E1DC3DE7;B5O3EC7;GA"
215 PLAY "G6E3EDC5;E6C3C5C;BA"
220 PLAY "D9;O2B7G3A03CD;G9"
225 PLAY "E6G3G5E3G;G6E3E5G;C7G"
230 PLAY "C6D3D7;O2A6B3B7;FG"
235 PLAY "O2G6O3E3E5D;O3D6C3C5O2B;FG"
240 PLAY "C9;O3E7C3O2BAG;C9"
245 PLAY "D6D3E5D3C;B6B3O3C5O2B;G9"
250 PLAY "O2A5G3A03C7;O3C5DCO2A3G;F"
255 PLAY "O2A6O3C3D5E;O3C6O2A3B5O3C;E"
260 PLAY "G9;E6G3G5F3E;D"
265 PLAY "G6G3A5G3E;E6E3C5E3C;C7F5G"
270 PLAY "E5D3CO2A7;G5GC7;C5EF7"
275 PLAY "G6O3E3E5D;O2B6O3C3C5O2B;GG"
280 PLAY "C9;O3E9;C9"
285 GOTO 10
300 CLS;PALETS 0,15,0,0,0
305 A$=CHR$(253);B$=CHR$(32)
310 C$=A$+A$+A$
312 C$=C$+B$+B$+B$
314 C$=C$+C$+C$+C$
316 D$=C$+A$
318 C$=D$+A$+A$
320 FOR B=0 TO 16 STEP 8
324 LOCATE 0,B;PRINT C$
326 LOCATE 0,B+1;PRINT C$
328 LOCATE 0,B+2;PRINT C$
330 LOCATE 0,B+3;PRINT C$
332 LOCATE 3,B+4;PRINT D$
334 LOCATE 3,B+5;PRINT D$
336 LOCATE 3,B+6;PRINT D$
338 LF B+6=22 THEN 345
340 LOCATE 3,B+7;PRINT D$
345 NEXT
350 T=STRIG(0);IF T<>4 THEN 10
352 CLS;C$=A$+A$
354 C$=C$+B$+B$
356 C$=C$+C$+C$+C$+C$+C$
358 C$=C$+A$+A$
360 FOR B=0 TO 20 STEP 4
362 LOCATE 0,B;PRINT C$
364 LOCATE 0,B+1;PRINT C$
366 IF B+1=21 THEN 375
368 LOCATE 2,B+2;PRINT C$
370 LOCATE 2,B+3;PRINT C$
375 NEXT
380 GOTO 10
400 CLS;PALETS 0,15,0,0,0
405 E$=CHR$(253);G$=CHR$(254);H$=CHR$(255)
410 M$=E$+E$+E$+E$
412 P$=G$+G$+G$+G$
414 Q$=H$+H$+H$+H$
416 M$=M$+P$+P$+M$+P$+Q$+Q$
420 FOR C=0 TO 22
425 FOR D=0 TO 3
430 READ E,F
435 DATA 4,2,12,3,16,3,20,2
440 COLOR E,C,F;COLOR E+2,C,F
445 NEXT;RESTORE
450 LOCATE 0,C;PRINT M$;NEXT
455 GOTO 10
```

二 传真机原理(上)

——传真图象数据的编码和解码

张建军 后俊堂 张景生

传真信号是将整个画面分解成相同单位的象元来发送的,所以从传输效率来考虑,显得非常冗长。根据 CCITT 有关建议,为提高传真机传递文件的发送速度,三类传真机在信源处理上使用冗余度压缩编码,缩短传送时间的频带压缩方式。下面将讨论频带压缩方式。

一、图象数据的压缩方式

从前面所述的传真通信过程中可知,文件传真是将文件上黑白的变化,经过光电转换,变成相对应的电信号,再经放大、模/数变换,转换成图象数据。这样,每幅图象对应的原始数据量是很大的。如果对一张 A4 幅面(210mm×297mm)的文件,在主扫描上以每毫米 8 象素、副扫描上以每毫米 3.85 条扫描线进行传输,可以计算出,大约有二百万个数据比特。如果不进行数据压缩,以每秒 4800 比特/秒的速率传输,需要将近七分钟的传输时间。由于文件中存在着连续是白色场或黑色场的现象,在传输中,也是一个象素一个象素的传输,延长了传输时间。如果用某些代码代替,在接收方也能还原成原来的图象。

在实际中,不同的传真图象具有不同的图象黑白变化规律,即黑、白象元的数量,黑、白游程长度出现的概率,以及符号间的相关性和各种迁移概率等均不同。根据有关单位对七种典型样张的统计,结果是白象素平均出现的概率是 93.3%。即使在文字很密的样张上,白象素的概率也达 88.8%,黑象素只占 11.2%。在这些象素中,约有三分之二的黑象素是相邻的,有 97% 的白象素是相邻的。再看相同颜色象素出现的持续长度(游程长度):在黑游程长度中,有 50% 小于 4 个象素;80% 小于 6 个象素;90% 小于 9 个象素。白游程长度中,有 50% 小于 18 个象素;80% 小于 61 个象素;90% 小于 164 象素。从这些统计特性中可以看到,一是黑象元出现的概率很小;二是相邻象元相关性很强;三是黑持续长度集中在较短部分,白持续长度没有黑象元那样集中。

利用上面的统计特性对黑白持续长度进行编码,对出现概率大的持续长度用较短的码字表示,对出现概率小的用较长的码字表示,就可以压缩大量的数据。

目前,传真的频带压缩技术发展很快,压缩的方法很多,归纳起来,可以分成三类:

1. 削减信号传输冗余度的模拟压缩方式——采用高效率的调制方式或采用多值残余边带方式等,这类方式用于二类机中。

2. 减少信源(传真信号)冗余度的码化数据压缩方式——进行信源编码。

3. 利用可变扫描速度减少信号冗余度的压缩方式。

这三类方式可概括于下表 1—1。

表 1—1 频带压缩方式的分类

充分利用传输频带	{ 多值化方式 AM—PM 方式
减少信源 冗余度的 编码方式	{ 一维游程长度编码方式 { 自然二进制方式 改进的霍夫曼编码
	{ 二维游程长度编码方式 { 多行一起处理方式 逐行编码方式
	{ 预测编码方式
可变扫描速度方式	

上述的三种频带压缩方式,削减信号传输冗余度的模拟压缩方式用在二类传真机中;后两种频带压缩方式用于三类传真机中。

削减信号冗余度的码化数据压缩方式的方法有多种,CCITT 和我国规定,在三类机中采用改进的霍夫曼码(MH)和二维游程长度编码方式(MR),这是下面介绍的重点。而可变扫描速度方式,一般用于非标准的机器中。

可变扫描速度方式的原理简介如下:

以相同的速度扫描传真原稿时,各部分信号的变化速度有着很大的不均匀性,有些信号变化得快,相对应的频率较高;有些变化得慢,相对应的频率较低。这样,传真信号所占的频带就宽。而传输传真信号所需的通频带是根据传真信号的最高图象频率来决定的,即由传真信号变化最快的部分来确定的。因此,在传输变化比较慢的信号时,实际通路利用率比较低。可以设想,如果用较快的速度去扫过图象色调变化较慢的那部分,就能较充分地利用给定的通频带。可是若以同样的速度去扫描图象色调变化较快的部分,则由于相应的传真信号频率提高了,因而不能通过给定带宽的通路。要是用较慢的速度扫过图象色调变化较快的部分,便可降低相应的传真信号频率,压缩信号频带,使它能通过给定带宽的通路。也就是说,采用可变扫描速度的方法,使信号变化的快慢在时间上被均化,便能起到压缩频带的作用。

二、信源编码方式的基本组成

在发送端,发送的原稿经扫描和光电变换后成为连续的电信号,再经模/数转换电路转换成数字信号,送入编码器进行编码,从而削减信号中的冗余度。从编码器输出的不均匀码化数据,通过缓冲存储器使之变为均匀信息,最后经调制解调器将它调制成适合于在

模拟信道上传输的信号发送出去。在接收端,对来自信道的信号进行解调、缓冲存储、译码等相应的反变换,送至记录器复制出与原稿相似的复制件。

编码器的作用是对信号进行编码,以削减信源的冗余度。但这样将会因为去除了信号间的相关性而使通信的可靠性下降,为此由信道编码电路增加部分码字,以提高通信的可靠性。

为了提高传输效率,码字之间不需另加间隔就能区分,为此编码器输出的电码应符合下述要求:

1. 任何一个电码都不是另一个电码的延长——即为非延长电码,也就是说不能在任何一个电码的后面添上一些符号而构成一个新的电码。

例如,由 0,1 两个符号构成的电码 $C = \{0, 10, 100\}$,就不是非延长码,因为 10 添上 0 就是 100,即 100 是 10 的延长。而电码 $C = \{0, 10, 11\}$,则是非延长码。

2. 编码器输出的应是平均码长为最短的单义电码。

所谓单义电码,就是任意一个有限长的电码序列,它只能被唯一地分割成一个个的电码。例如: $C = \{0, 10, 11\}$ 就是一种单义电码,因为任意一串有限长的 C,比如 100111000,只能被分割成 10, 0, 11, 10, 0, 0,任何其它分割方法都不是 C 的电码。如划分成 10, 01, 11, 00, 0 时,其中 01, 00 就不是这个电码集中的电码,而不是 C 中的电码也就不能代表消息。所以,它只有唯一的划分方法,因而是单义电码。又如 $C = \{0, 01, 10\}$ 就不是单义电码,因为对于一串有限长的 C,比如 0010,既可以分为 0, 0, 10;也可以分为 0, 01, 0, 它不能被唯一地分割成一个个的码字。

此外,要使得输出的码字平均码长最短;概率大的消息,应编为短码;概率小的消息,应编为长码。

综上所述,传真机采用编码方式可以压缩传真信源的冗余量,编码方式是多种的。CCITT 规定在三类传真机中采用改进型霍夫曼码(MH)和改进的像素相对地址指定码。

三、改进型霍夫曼编码

霍夫曼码是根据文件中黑白游程出现的概率来分配码字的,因而是一种理想的编码,其压缩效率高。在传真机实际扫描一行或几行时,会出现不同的游程长度。这就是说,从传真机时发出的消息数远不止几条或十几条,而是成百上千条,这样,霍夫曼编码虽然从效率上来讲是理想的,但它需要很长的游程——码字对照表,即需要容量很大的存储器,因此很不经济;同时,游程长度的概率统计对于文件的每行、每页都不相同,而对于每一套概率统计都要有一个游程——码字对照表,这就使霍夫曼码实际上难以实现。CCITT 在经过对各种黑白文件中黑像素和白像素各种持续长度的大量统计后,根据各种持续长度的出现概率和霍夫曼码编码原则,规定了对应黑、白像素的各种持续长度代码,并且对霍夫曼编码方法进行了改进,提出了由组合基干码加结尾码组成的代码结构。从而弥补了霍夫曼

码实行上的困难。

改进型霍夫曼码的编码原则如下:

1. 数据

一条扫描线的数据由一串可变长度的码字组成。每个码字表示一个全白或全黑的持续长度。黑白持续长度是交替出现的,当持续长度小于或等于 63 时,用 0~63 个游程长度所对应的结尾码表示。当持续长度大于 63 时,则用一个等于或短于该持续长度的组合基干码和一个该持续长度与组合基干码所代表的长度之间的差值的结尾码字表示。例如下图 1:

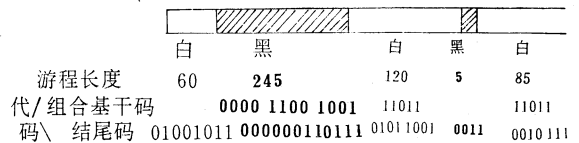


图 1. MH 编码举例

在数据中,为保持颜色同步,规定所有的扫描线的数据均由白持续长度码字开始。如果实际的扫描线是从黑持续长度开始的话,则发送一个长度为零的白持续长度。

2. 线终码(EOL)

线终码是在一条扫描线的数据之后传输的,表示一条扫描线的结束。线终码是在有效的扫描线数据中不可能出现的、唯一的码字。因此,在突发差错出现后,可能实现同步。其格式为:000000000001。

3. 填充码(FILL)

每条扫描线的拾取和记录需要一定的时间,CCITT 在制订三类传真机技术标准时,规定了全编码扫描线传输时间。标准的传输时间是 20ms。

全编码扫描线是指一条扫描线的像素在编码后,其黑白游程持续长度所对应的码字及必要的控制码字的总比数。

当一条扫描线的像素在编码后,小于规定的时间,则加入填充码来保证全编码扫描线传输时间。填充码的格式是可变长度的“0”串。

4. 转回至控制规程(RTC)

在报文传送结束,连续发送 6 个线终码,表示转回至控制规程。在 RTC 信号后,发送机将按照帧格式以及建议 T. 30 规定的控制信号速率发送各种报文后命令。

四、改进的像素相对地址指定码(MR)

改进型的相对像素地址指定码(MR)是 CCITT 推荐的在三类机使用的二维编码方案。这种编码方案不仅利用了水平方向的像素间的相关性,而且利用了垂直方向上线与线之间的像素间相关性,因此,MR 码比 MH 码压缩比更大。

MR 码因为利用了水平、垂直方向上像素间的相关性,因而容易出现差错,为了控制差错,采用 K 条扫描线为一组进行编码,在 K 条扫描线的第一条进行

MH 编码,其余 K-1 条用 MR 再根据前一条进行编码。

K 的最大取值,CCITT 规定:在标准垂直扫描线密度时,K=2;在选用的高垂直扫描线密度时,K=4。

MR 编码的方式与 MH 不同,它是对迁移象素的位置进行编码。

1. 迁移象素

迁移象素是指在同一条扫描线上,其颜色(即黑或白)与前一个象素的颜色不相同的象素。迁移象素分五种,其定义如下:

a_0 :正在编码的扫描线上的参考象素或起始迁移象素。在正在编码的扫描线的开头, a_0 是一个假想的白迁移象素,它紧靠在这一扫描线的第一个象素之前。在对此正在编码的扫描线编码时, a_0 的位置是前一个编码模式决定的。

a_1 :位于正在编码的扫描线上,紧靠 a_0 右面的下一个迁移象素。

a_2 :位于正在编码的扫描线上,紧靠 a_1 右面的下一个迁移象素。

b_1 :位于参考扫描线上,紧靠 a_0 右面且与 a_0 颜色相反的第一个迁移象素。

b_2 :位于参考扫描线上,紧靠 b_1 右面的下一个迁移象素。

2. 编码的模式

在编码过程中,在 MR 的三种编码模式中选出一一种,对正在编码的扫描线上的每一个迁移象素的位置进行编码。三种模式分别为通过模(P)、垂直模(V)、水平模(H),其定义如下:

(1)通过模(P)

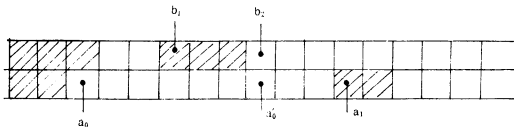


图2 通过模

当 b_2 位于 a_1 的左面,认为是通过模。

完成这个模式的编码后,把位于正在编码的扫描线上 b_2 之下的象素置为下一个编码的 a_0 (即在 a_0' 处),如图2所示。

但是,当 b_2 正好出现在 a_1 之上,则不认为这种状态是通过模。

(2)垂直模(V)

当鉴别为垂直模时, a_1 的位置是相对于 b_1 的位置来编码的。此相对距离可取以下七种数值之一,即 $V(0)$ 、 $V_R(1)$ 、 $V_R(2)$ 、 $V_R(3)$ 、 $V_L(1)$ 、 $V_L(2)$ 和 $V_L(3)$ 。

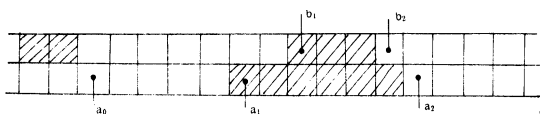


图3 垂直模

它们均用各自的码字来表示。下标 R 和 L 表示 a_1 在 b_1 的右边和左边,括号中的数字表示距离 a_1b_1 的数值。在垂直模编码后,把 a_0 置于 a_1 的位置上,如图3所示。

(3)水平模(H)

当鉴别为水平模时,如图4, a_0a_1 和 a_1a_2 两个持续长度要用码字 $H+M(a_0a_1)+M(a_1a_2)$ 来编码。H 是取自二维编码表(表2)的标志码字 001。M(a_0a_1)和 M(a_1a_2)是分别表示 a_0a_1 和 a_1a_2 的持续长度及颜色的码字,它们取自相应的黑或白的一维码表(MH)。在水平模编码后,把 a_0 置于 a_1 的位置上。

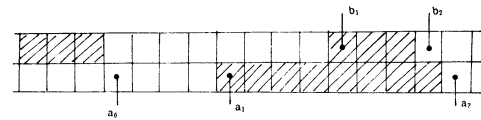


图4 水平模

表2 二维码表

模	需要编码的象元	符号	码字	
通过模	b_1b_2	P	0001	
水平模	a_0a_1, a_1a_2	H	$001+M(a_0a_1)+M(a_1a_2)$	
垂直模	a_1 在 b_1 之下	$a_1b_1=0$	$V(0)$	1
	a_1 在 b_1 的右边	$a_1b_1=1$	$VR(1)$	011
		$a_1b_1=2$	$VR(2)$	000011
		$a_1b_1=3$	$VR(3)$	0000011
	a_1 在 b_1 的左边	$a_1b_1=1$	$VL(1)$	010
		$a_1b_1=2$	$VL(2)$	000010
$a_1b_1=3$		$VL(3)$	0000010	
二维(扩充)			0000001×××	
一维(撒充)			000000001×××	

(4)编码过程

编码过程就是要识别编码行上每一迁移象素所应使用的编码模,而后从码表(表2)中找出相应的码字。编码过程的流程图如图5所示。

第一步:如果识别为通过模,就用码字 0001 来编码。然后将 a_0' 置于 b_2 之下,作为下一个编码的起始象元 a_0 。如果没有检出通过模,则作第二步处理。

第二步:确定 a_1b_1 相对距离的绝对值。

如果 $|a_1b_1| \leq 3$, a_1b_1 用垂直模编码,然后把 a_1 的位置作为下一个编码的起始象元 a_0 。

如果 $|a_1b_1| > 3$,则在水平模编码标志码字 001 之后, a_0a_1 和 a_1a_2 分别用一维改进型的霍夫曼码来编码。然后把 a_2 的位置作为下一个编码的起始象元。

(5)扫描线的第一个和最后一个象元的处理

a、第一个象元的处理

每个正在编码的扫描线上的第一个起始象素 a_0 被假想为正好在第一个象素的前面,并被看作是白象素。

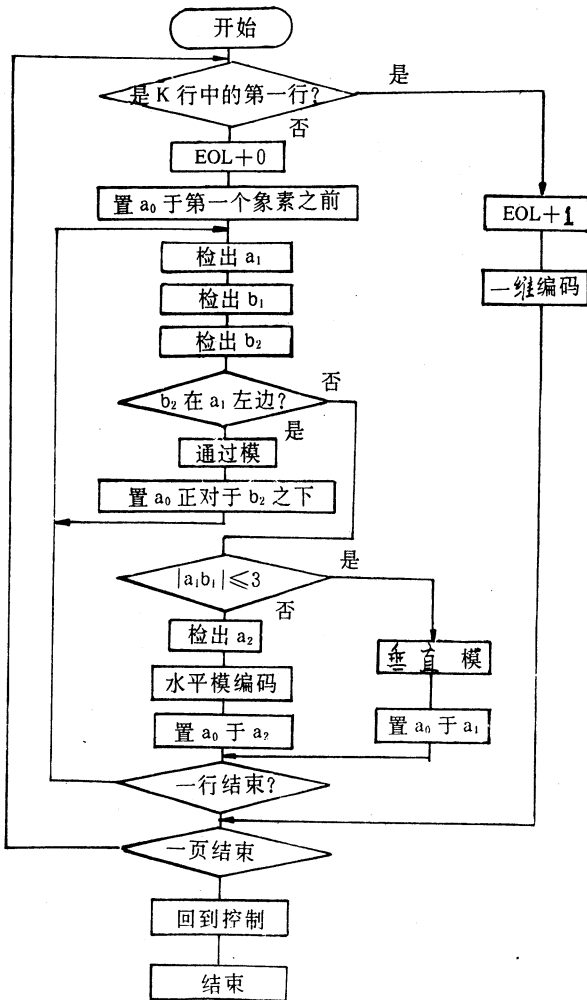


图 5

每条扫描线的第一个持续长度 a_0a_1 用 $(a_0a_1 - 1)$ 来替代。因此,如果第一个持续长度是黑的,且要用水平模编码,那末,第一个码字相当于一个长度为零的白持续长度。

b、最后一个象素的处理

编码行的编码一直要进行到把位于实际最末一个象元之后上假想迁移象素的位置编完码为止,它可以作为 a_1 或 a_0 来编码。若在这条扫描线的整个编码过程中没有检出 b_1 和 b_2 , (b_1 或者 b_2), 则在参考行上实际最末一个象元之后的假想迁移象素位置被认为是 b_1 或 b_2 。

(6)线同步码字

在每条编完码的扫描线之后要加上一个线终码字 (EOL) 000000000001。在 EOL 之后跟着一位特征比特,它指出下一行是用一维编码还是二维编码。

格式: EOL+1; 下一行用一维编码

EOL+0; 下一行用二维编码

(7)填充码

为保证全编码扫描线的最小传输时间,必须在编码数据和线同步码之间插入填充码,但不得在数据中插入。

格式: 可变长度的“0”串。

(8)转回控制规程(RTC)

RTC 所用的格式是 6 个连续的线同步码字,即 $6 \times (EOL+1)$ 。

(9)由于二维编码是参照上面一条扫描线,如果有一条扫描线出现传输差错时,后面的扫描编码会引起出错。为了对被干扰区加以限制,数字传真机采用以下方法:

a 在副扫描的线密度为 3.85 线/毫米时,采用一条扫描线为一维编码,后继一条扫描线为二维编码,以后逐条轮换。

b 在副扫描线密度为 7.7 线/毫米时,采用一条扫描线为一维编码,后继三条扫描线为二维编码,以后按此规律,每四条轮换一遍。

计图形模块生成器;任意格式报表生成器;数据字典与文档生成器;

软件服务

C-DBAG 中文数据库应用生成器 F/D * AG V5.0

零售价: 3900 元

使用本系统只需按照提示输入用户需求,不需编程各种管理软件,在功能变动时可任意改动,适用于网络及单用户环境。

功能: 生成 dBASE+, FoxBASE+ 源程序; 生成任意格式报表; 自动生成数据字典及文档; 工程图形生成与管理。

组成: 下拉/弹出式菜单生成器; 查询模块生成器; 数据录入与维护模块生成器; 数据库文件结构生成器; 彩色统

灵科汉卡

零售价: 1200 元

西文软件无需汉化即可直接使用中文,直接写屏,支持 $1024 \times 768, 640 \times 480, 640 \times 350$ 图形方式。支持中文排版功能。中文简体与繁体,中文与西文之间一键转换。支持针式打印机、激光打印机及鼠标。精密宋、楷、黑、仿宋体平滑放大输出。能识别中西文窗口。显示字库不占用内存,且显示速度大大提高。

灵科加密器

零售价: 180 元

可加密任何 EXE 和 COM 文件,也可对任何源程序实施加密,如 dBASE, FoxBASE 等,加密文件个数不限。安装方便,只要外插于打印机接口上,且对打印机工作无丝毫影响。加密手段方便,保密性极高。具有强有力的反跟踪功能,使解密成为梦想。

(转 45 页)

PC 间能传送文件的程序

申老师：上次介绍了二台 PC 机之间用简化的 RS-232C 电缆的连接方法，以及能传送键盘信息的最简单的 BASIC 程序。采用 BASIC 的原因，是由于它普及面广、易学、易调试，对于掌握 PC 间通信的知识是很好的工具。

这次介绍一个能在 PC 间传送文本文件的简单程序。硬件的连接还是跟上次一样，电缆可以使用简化的只有三根线的电缆，但如果能使用九根线全部连齐的电缆则更好。

发送程序如下(取文件名为 S2. BAS)：

```
10 REM Send text file demo
20 INPUT "Send file name: ", SFILE$
30 ED$ = " * * * END OF DATA * * * "
40 ON ERROR GOTO 200
50 OPEN "COM1:1200,N,8,1,RS,CS,DS" AS #1
60 OPEN SFILE$ FOR INPUT AS #2
120 IF EOF(2) THEN 210
130 LINE INPUT #2, T$
135 PRINT T$
140 PRINT #1, T$ : GOTO 120
200 PRINT "Error occurs, exit from program"
210 PRINT #1, ED$
215 PRINT #1, " "
220 PRINT ED$ : CLOSE
230 END
```

其中，标号为 20 的语句，从键盘上接收一个欲传送到对方的文本文件名。该文本文件必须是事先已经存在的，例如，批处理文件 .BAT，某些说明文件 README. TXT 等都是文本文件。文本文件也可以在 DOS 下由文本编辑程序 EDLIN、WS 等建立，还可以简单地用 COPY 命令从键盘输入而建立，办法如下(设文件名为 FN. TXT)：

C>COPY CON: FN. TXT

……(从键盘输入文件内容)

以 ctrl-z 和 ↵ 结束输入。

50 句，把通信口 COM1 打开在 #1 通道，传输速率选每秒 1200 位，无校验位，8 位数据位，1 位停止位。

60 句，把欲传送的文本文件打开在 #2 通道。对于通信缓冲区，该文件是作为输入(INPUT)用的。

120 句，是在传送过程中判断是否遇到了文件尾(End Of File)。如果是，就传送一个结束信息“ * * * END OF DATA * * * ”，令对方知道传送完毕。

130 句和 140 句，是从文本文件取一行字符，送至 COM1 发出。135 句，用于监视传送的内容。

215 句，是在发出最后一个字符串 ED\$ 后，再往

COM1 送一个空格符。这个空输出语句，保证了最后一个字串能发到对方。

接收程序如下(取文件名为 R2. BAS)：

```
10 REM Receive file demo
20 RFILE$ = "TMP. TXT"; ED$ = " * * * END OF
DATA * * * "
50 OPEN "COM1:1200,N,8,1,RS,CS,DS" AS #1
60 OPEN RFILE$ FOR OUTPUT AS #2
90 PRINT "Receiving……"
100 LINE INPUT #1, T$; IF T$ = ED$ THEN 200
105 PRINT T$
110 PRINT #2, T$; GOTO 100
200 PRINT ED$; CLOSE
210 END
```

20 句取一个文件名 TMP. TXT，作接收文件用。

如果接收到有用的文件，可以用 RENAME 改成别的名。实际上，在实用软件中，文件名也是从对方传过来的。

50 句同发送方，必须与发送方参数一致。

60 句打开接收文件在 #2 通道。对于通信缓冲区来说，它是作为输出(OUTPUT)用的。

100 句和 110 句，是从通信口按字符行接收，从 #2 通道送到文件上存储。

100 句和 200 句，判断结束信息 ED\$，打印出来，并关闭 #1、#2 通道。如果不关闭，再运行时会在 50 句出现“文件已打开”的信息而停止运行。

现在我们开始做实验：

小王和小李一人一台机，按上次介绍的硬件连接方法连好，在 BASIC 下，一人输入发送程序，另一人输入接收程序。运行时，接收程序先运行，在出现“Receiving……”时，再运行发送程序，顺利地传送了一个小的文本文件 AUTOEXEC. BAT。接收端用 SYSTEM 命令退出 BASIC，在 DOS 提示符 C> 下，用 TYPE TMP. TXT 看到了接收到的文件，与发送方的完全一致。他们又传送了 CPAV 的一个使用说明文件 README. TXT，有 31KB 大，花了 4 分多钟时间才传送完。接收过程中，他们看见硬盘的灯在间断地闪亮，指示了写盘的过程。

小李：申老师，怎么要传那么长时间？

申老师：串行通信是比较慢，时间是可以计算出来的。异步串行通信中，每字符约占 10 个二进制位，本实验正好用了 10 位，其中数据位 8 位，停止位 1 位，启动位 1 位。传输速度每秒 1200 位(1200bps)，故每秒能传 $1200 \div 10 = 120$ 个字符。文件长 31KB 接近 32,000 个

字符(英文 ASCII 字符每字符占 1 个字节,若中文则每个汉字占 2 个字节),需要时间 $32000 \div 120 \div 267 \text{ 秒} \approx 4.5 \text{ 分钟}$ 。如果传输速率降低至 300bps,加之此时停止位按规定应当改为 2 位,则更慢了,传这个文件差不多要 20 分钟。但若采用象深圳股票交易系统那样的远程高速串行通信专线,传输速率为 64Kbps,则大概只需 5 秒钟。

小王:除了文本文件,还能传其它类型的文件吗?如 .EXE, .COM 等?

申老师:这个程序仅限于传文本文件。能传其它类型文件的程序复杂一些,这里不准备介绍了。因为已经有一些很好的实用通信软件,可以直接用来传送任何类型的文件,甚至整个目录、整个磁盘的信息。以后将介绍其中的一些软件。

小王:调试这两个程序时,要注意些什么?

申老师:一是传输速率不能选得太高,二是要有能快速存取文件的硬磁盘。这是由于通信的传输速度与文件的写盘速度有匹配问题,传得快而来不及写盘,就会出现“通信缓冲区溢出”(com buffer overflow)的出错信息,并停止传送。

小李:这有办法解决吗?

申老师:当然有。可以在两方的程序上都加上一些语句,使发送方和接收方都按 XON/XOFF 协议进行工作。其原理是这样的:接收方不断检测通信缓冲区是否快满了(查 LOC(1)的大小),一般在达到一半满(例如 128 字节)的时候,就立即向发送方发过去一个“暂停传送”(XOFF)信号,实际上就是发去一个控制字符,在 IBM-PC 中使用的是 ASCII 码为 19 的字符 CHR\$(19)。发送方接收到此字符,就立即暂停发送,直至接收方把通信缓冲区的内容处理完毕,再发来“继续传送”(XON)信号,才又继续发送,这样就不会丢失数据了。XON 信号实际上就是一个控制字符 CHR\$(17)。具体的程序,可以查阅 IBM-PC 的 BASIC 使用手册及其他有关的书籍。

通信中,规定了不少的通信规程,或称协议(Protocol),要求发送方和接收方共同遵守,才能使通信得以正确进行。XON/XOFF 就是其中最简单的一个。

小李:接收方也能发送信息吗?

申老师:可以的。你看打开 COM1 的语句,并没有指定它是 INPUT 还是 OUTPUT,它是既可以发送也可以接收的。严格地说,你们两方应称为“呼叫方”和“应答方”,只是谁先发起通信而已,两方都可以发送和接收。当然这个简单程序没有体现这点。双方都可以发送和接收,这在通信上称为“双工”,即可以双方向工作的意思。如果发送和接收只能分时(即分先后)进行,称为“半双工”;发送和接收能同时进行的,称为“全双工”。现在通信中的大多数软、硬件,都可以实现“全双工”。电缆上不是有三根线吗?有一根是信号输出线,一根是输入线,余下一根是公共信号地线,可以同时双向传送信号。

小李:我很关心电话线的传送。电话线只有一根信号线,那么远距离通信时只能“半双工”工作吗?

申老师:不是的。在一根信号线上也可以同时传送 2 路以上的信号,即也可以实现双工。使用电话线时,要使用调制解调器(MODEM),呼叫方和应答方采用不同的载波频率来载送数字信息,不同频率的载波可以在同一导线上同时向不同方向传送。这些是通信上另一方面的知识。

小李:申老师,下次开始介绍用电话线的传送方法吧?

申老师:是的,下次开始介绍使用调制解调器(MODEM),利用电话线进行远距离 PC-PC 通信的知识。最近国家信息中心的大型信息库已经开放使用,用这个方法我们就可以同信息库联上。这要求 PC 附近有电话线,还要添置一个调制解调器(MODEM)。采用可以插在 PC 内的 MODEM 卡,比用独立的 MODEM 设备要方便而且便宜得多。下期再见。

回音壁

长山先生:您好!

我是“电子与电脑”订阅者,是想了解一些适宜我国中小学学生使用的,档次较宽的、属于启蒙型的电脑。在我们平湖市相当多的一些中学生,甚至高中生,他们只知“计算器”,而不知“计算机”(电脑)是怎么回事。为此我曾订阅过“无线电”、“电脑”、“学生计算机世界”报,感到“电子与电脑”和“学生计算机世界”报较有看头。

看了 93 年一期《电子与电脑》93 年新春致读者,对贵刊“读者联谊”栏目很感兴趣,希望栏目能刊登一些以下方面的内容:本人为了想购买一台配合游戏机(任天堂型)上使用的键盘电脑,曾留意报纸,收集了①小天才 IOK-91 型电脑键盘;②天马 TM5510 型电脑键盘;③新星 KB-200 型电脑键盘;④大岛中英文电

脑键盘;⑤裕兴中英文电脑键盘;⑥科特 FCS-90 型电脑学习机;⑦金字塔 PEC-9388 型电脑学习机;以及中档机“中华学习机”等广告信息。以上说明各地研究所和厂家在向高档次的 386、486 和向低档次的电脑学习机两极发展。高档机供应给科研所及企事业单位,因此各种辅助设备和应用软件齐全。低档机供应给家庭孩子们使用,作为电脑启蒙。这样就产生了一个问题,就是电脑键盘的“BS”卡都是引进原版英文版。县级市、镇小学不教英语,小学生编程序就得先学英语,才能懂得英文指令语句。中文“BS”卡国内又没有版本。本人留意各种广告,还没有发现国内那一家科研机构在研制出中文版“BS”卡供孩子们使用。

(下转 56 页)

有必需的控制功能。

在回扫式变换器中,变压器 T_1 并不是一般意义上的变压器,因为一般变压器是把初级能量耦合到次级去,尽可能少地储存能量。而这里的 T_1 主要是作储能用,其工作原理如下:

当 Q_1 导通时,在初级线圈中流过电流,并储存能量,但由于 T_1 的输入和输出线圈的同铭端极性相反,四个二极管 CR_3 、 CR_4 、 CR_5 、 CR_6 均承受反向偏压,没有能量传送到负载上。这时输出是通过输出电容 C_9 、 C_{11} 、 C_{13} 、 C_{15} 的放电而获得的。

当 Q_1 关断时,由于线圈中磁场急剧减少,使得次级线圈的电压极性反向,二极管导通,从而向输出电容充电,并向负载传递电流。由于 Q_1 与 $CR_3 \sim CR_6$ 工作相位相反,即 Q_1 关断时, $CR_3 \sim CR_6$ 导通工作,耦合的能量经副边传送到负载。这像在 Q_1 回程时传送能量一样,故称为回扫变换器。

变压器 T_1 设计为工作于不连续导通模式,这样由于没有右半平面零点而使得补偿电路简单,并且由于功率很小,在不连续导通模式下不会形成很高的峰值电流。由于能量传送到负载经过磁的转换,电网的干扰不能直接进到负载,故线路的抗干扰能力强。

为了便于读者在实际工作中选用,表 1 列出了一些开关稳压电源用集成控制器的型号及其主要特性。在实际使用这些集成控制器时,首先要根据开关管工

作频率来确定振荡器的振荡频率,然后在补偿(⑭)端初定补偿参数,经细心调整该参数后,开关电源即能满意地工作。

表 1 开关稳压电源用集成控制器

型号	输入电压 $V_{inmax}(V)$	基准电压 $V_{ref}(V)$	振荡频率 (KHz)	输出电流 $I_{omax}(A)$	封装 型式
LM1524 LM2524 LM3524	40	5	100	0.1	DIP-16
LM3524A	40	5	500	0.5	DIP-16
TL494	42	5	300	0.2	DIP-16
SG3526	40	5	400	0.2	DIP-18
SG3527A	40	5.1	400	± 0.4	DIP-16
VC3840	32	5	500	± 1	DIP-18
MC34060 MC35060	40	5	300	0.25	DIP-14
MC34066	20	5.1	谐振型	1.5	DIP-16
GP605	20	5	2MHz	0.8	DIP-16
Si9120		4	500	0.2	DIP-16

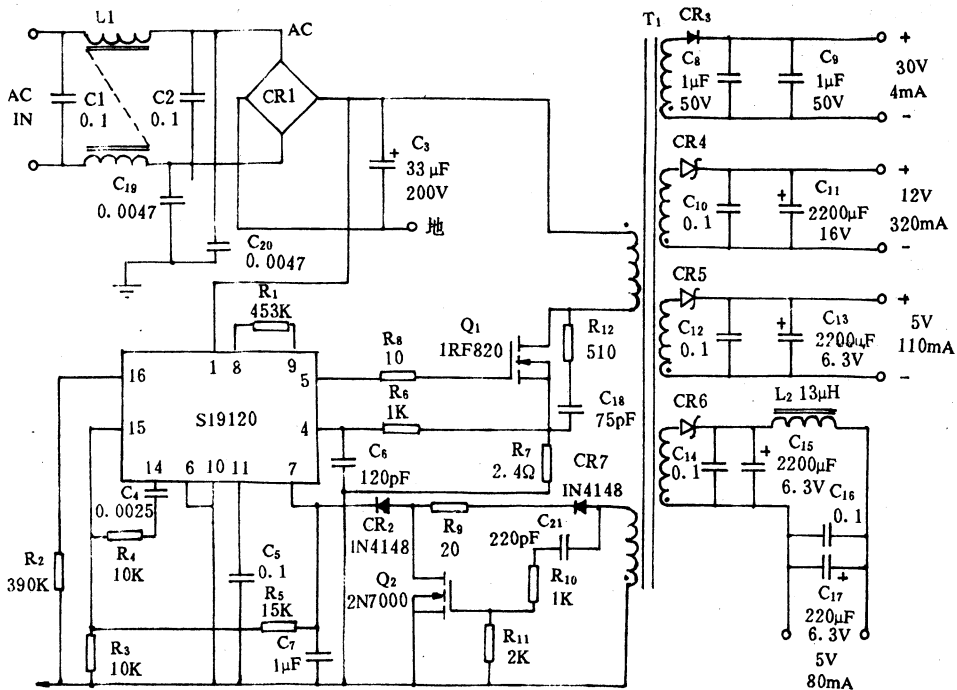


图 3

(上接第3页)

结和发送。按下“微调”按钮可保持图像顶部清晰地展现。在各个按钮上方的发光二极管是开关指示器，它们在进行相应操作时点亮。

在处于自动方式时，每隔 38 秒电视电话就接通电话线达 12 秒。所以，如果当电视电话接通时你将听到一个忙音信号。不过，如果你等 12 秒后再拨号，电话线就不会占线，因为 12 秒已经过去，要等 38 秒后它才会再次占线。在实际应用中，你能够在接收到一个良好的电视图像后在遥远的地方挂断电话。

这种自动方式可以使一个有趣的应用成为可能，即能够使用电视电话从一个加油站遥控监视你的房屋或商店或者是书籍储藏。设立一个电视摄像机观察一块区域，电视电话的自动方式就会每隔 38 秒通过一条普通的电话线发送一个新电视图像。不过，这个计划只有当你打另一个电视电话，并且走开时没有把话筒挂上时才能实现。一天 24 小时使用一条电话线的费用很昂贵，但是，对于某些确实需要的监视位置，这个费用也许可以接受。使用屋内电话线，遥远地监视某些东西而无需占用新的电线也许是一种便利的方法。进行监视的另一种方法是把电视电话的输出与一个具有控制功能的电话应答机连接起来。这样就能在监视场地控制远方电视电话的工作，而不必一天 24 小时地占用电话线了。

电视电话的应用非常广泛，例如，你可以到电话公司所设的电视电话中心，在那里能看到同样坐在某个电话亭中的人与你通话。虽然费用比较昂贵，但对方的音容笑貌俱在眼前，所以这也不失为一种很好通讯方法。它还有许多潜在的用途，而不是简单地让祖父母们看见他们的子孙。这些用途可以是一个商业艺术家和专栏作家在一个目录专页显示方面进行合作；也许是一个建筑领班在他周游世界的半途中把建筑场地详细情况显示给坐在办公室里的建筑师看，了解建筑状况；还可以象本文前面所提到的那样，用 SSTV 系统从遥远的太空传回图像。

另外，电视电话已逐渐在办公室中扮演极为重要的角色。人们可以利用它的画面文字传输功能，传送实时照片和文字的动像通信，在办公室的商务会议、商品咨询业务、印刷品的校对等方面获得极大的收益。对于参与商讨决策的人来说，需要收集现场的最新信息，而抽出出差时间又不是轻而易举的，电视电话便可用来进行小型电视会议和各种远距离活动场面等的演示。甚至用它组成桌上电视会议系统，在自己的办公室召开电视会议，让参与会员在各自现场参加就行了。

时至今日，电视电话又有了长足发展。去年 11 月在北京举行的计算机新技术展览会上，日本一些公司展示了它们的新产品。其中的彩色动像电视电话机，采用薄膜晶体管彩色液晶显示器，分辨率为 352×240。值得注意的是，它的编码方式采用了数据压缩技术，传送速度高达 15 帧/秒，是本文样机的 180 倍，已经接近了人眼所认为的连续画面。不过它的价格也相当昂贵，

达到数万美金，一般人还难以承受。

这就是关于电视电话的全部情况，过去只有迪克·特雷西才能拥有的东西，如今已经大众化了。可以预见，电视电话必将完全取代电话，走进办公室，走进千家万户，活跃于人们生活的广阔领域。到那时，象大诗人白居易在《长恨歌》中“一别音容两渺茫”那样的感叹也就会永远地成为历史了吧。



咨询

▲甘肃候明问：

本人近日买了一台 M—2024 针式打印机，但不知如何和中华机配合使用，如何连接(打印卡已从贵部买到)和怎样打印汉字及图形。

答：目前 24 针打印机仍采用与九针打印机兼容的 Centronics 接口标准，连接打印机的插头各脚旁边有号码，各脚的接法如下：

- 1 脚 DATA STROBE 数据选通 主机→打印机
- 2~9 脚 DATA1~DATA8 对应 8 位数据线 主机→打印机
- 10 脚 ACKNLG(打印机回答信号) 打印机→主机
- 11 脚 BUSY (打印机忙) 打印机→主机
- 12 脚 PAPER EMPTY(无纸信号) 打印机→主机
- 13 脚 SELECT(联机状态) 打印机→主机
- 14 脚 AUTO FEED(自动走行信号) 主机→打印机
- 16 脚 SIGNAL GND(信号地)
- 17 脚 FRAME GND(机架地)
- 18 脚 +5 伏
- 19~30 脚 SIGNAL GND(信号地)
- 32 脚 FAULT(故障信号) 打印机→主机
- 33~36 SIGNAL GND 信号地

为了驱动打印机，除信号地之外，最低限度需要的控制信号有 DATA STROBE, BUSY 或 ACKNLG 之一，再加上 8 位数据线。实际上 Apple 或中华机的打印机接口电路就只用这些连线，因此，不作线路改变，直接将打印机插头插入 M2024 打印机，便可以在西文方式下打印文本数据。至于汉字打印或图形硬拷贝，则与打印机在位元映象工作状态下的控制码和送数格式有关，由于不同型号的打印机采用的控制码不尽相同(我们没有这方面的资料)，所以需要自编打印驱动程序。图形硬拷贝的打印驱动程序可参考本刊 92 年 11 期张益贵同志的一篇《CEC 与 1724 打印机配接图形硬拷贝程序》编制。

(苏华)

佳海——计算机用户的朋友

一、通用图文数据管理软件 是一种既不必懂英语,也无需编程的“傻瓜”型软件,集排版、数据库管理、自动制表与绘制统计图等功能为一身。它的最大特点是全中文菜单提示,无需二次编程,操作简便,一学就会。凡具有初中文化水平的用户,均能在一天内学会使用,数周后即能熟练掌握。是专为广大的非专业计算机工作者设计的,适用于各行业办公事务管理自动化的需要。

90年以来连获“长城杯”、“浪潮杯”大奖,开发后有地矿、纺织、汽车、有色工业、工程兵部等近万用户使用,受到普遍欢迎,又经反复修改,功能已相当完善。

本部对用户实行软件终生保用与免费更新版本的优质服务。欢迎来电咨询或索要资料。

二、以优惠的价格推出286、386、486原装机及兼容机,并进行电脑主板更新换代业务,保证售后服务。

常年提供维修计算机及外围设备业务。

三、销售办公自动化设备、复印机、传真机、纸张油墨、电子元器件等,以旧换新各牌号复印机,旧机的酌情折价。并设有常年维修业务。

常年代培五笔字型及激光照排人员,学期二十天,包教包会,期满择优录用。

四、可承担计算机房等室内装修工程,符合设计规范,保证安全。

佳海愿与各界通力合作,欢迎来电来人洽谈。

北京佳海计算机技术公司

地址:北京阜成路南五楼(钓鱼台国宾馆西墙外)

邮编:100037 电话:8310866

联系人:和京

中软总公司库压品门市部

优惠向您提供下列产品

一、计算机盘片:

3M. BASF. 五英寸,单面/双密,20元/盒。100片以上,15元/盒;3M. 五英寸,单面/双密,硬分段,16扇区,15元/盒。CE. BASF 五英寸,双面/双密,28元/盒。100片以上24元/盒。DATALIFE. 五英寸,单面/双密,20元/盒。100片以上18元(塑料盒);DATALIFE. 五英寸,单面/双密,30元/盒。100片以上28元/盒(塑料盒);BASF. MASTER,八英寸,单面/单密,60元/盒;3M 八英寸,单面/双密。80元/盒;JANUS. 五英寸软加密盘。双面/双密,10元/张(低版本)。

二、盒式磁带:

BASF. 1/2×1200英尺。60元/盘;

3M. 1/2×1200英尺。80元/盘;

三、8086单板机,500元/台,9250智能中英文打字机4200元/台;铅字打字机,油印机;日本1550九针打印机,1200元/台。60M磁带机,1000元/台。24针打印机9400,2800元/台;

四、绘图仪:

SPL——400,3000元/台;LP——3700,2万元/台;DXY——800,3000元/台;数字化仪,WT——4000,3000元/台。

五、显示器:

12英寸彩显,800—1000元/台。14英寸彩显,1000—1300元/台。

六、各种配置计算机:

IBM PC/XT. AT/286. 中华 386/16. 长城 GW0520CH. GW286.

七、主板机、各种显示卡、长城打印机汉卡;UPS400W、

UPS1000W。

八、其它:

0520A 电源,TP—80 电源,打印机电缆线;光电鼠标器、稳压电源500W、1000W,扁线压接工具、实验版、起拔器、1500K 盒式色带、P3/P7 盒式色带、元器件;5英寸、8英寸盘盒。

联系人:温友良 孟秀华

电话:8317722—1112

地址:北京市海淀区学院南路55号

电脑大厦112房间

邮政编码:100081

乘车路线:16路汽车电脑大厦下车

(上接52页)

如果留意一下各地百货商店和家电商店,柜台内陈列的游戏卡,从20多元到200多元一盒,应有尽有,就是没有“浮点BASIC—BS”卡和“中文BS卡”。为什么厂家不生产一些小学、初中、高中的语文,数学,理化,英语等的“BS”卡呢!(相当于中华学习机的教学软件)。

浙江平湖家具厂俞文水

俞先生:

裕兴和大岛中英文游戏机电脑键盘均带有英语、数学等教学演示程序。

它所附的二合一中西文BS卡,带有中文功能,借助拼音可完成汉字输入。

F BASIC语言作为一种计算机语言经过实践,只能用英文。编辑程序也只能用英文。但经常用的指令也不过十几条,很好记。

孙立军

孙立军为裕兴机械研究所人员。对电脑键盘有兴趣的读者可直接与其联系。

地址:北京西城区西教场35号

邮编:100035 电话:2252309

长 山