上 海 市

电子计算机应用技术资料汇编

第 十 辑

上海科学技术文献出版社

上　海　市
电子计算机应用技术资料汇编

第　十　辑

本汇编编辑部　主编

上海科学技术文献出版社

上海市电子计算机应用技术资料汇编

（第十辑）

本汇编编辑部　主编

*

上海科学技术文献出版社出版

（上海市武康路2号）

新华书店上海发行所发行

上海商务印刷厂印刷

*

开本 787×1092 1/16　印张 17　字数 424,000

1986年10月第1版　1986年10月第1次印刷

印数：1—3,800

书号：15192·439　定价：3.50元

《科技新书目》114-230

目　　录

计算机硬件与接口

数学模型与应用软件

数据处理与情报检索

微　机　应　用

数 值 计 算

本汇编第十一辑要目预告

1. 中文仓库管理软件…… 上海交通大学
2. 三维立体图形显示软件包…… 上海宝钢电厂
3. 大电力用户电费管理软件系统的分析和设计…… 上海市计算技术研究所
4. 卫星转发器多载波互调产物的计算机模拟…… 邮电部第一研究所
5. 下颌骨颏部受击的计算机分析… 上海第二医科大学 上海煤矿机械研究所
6. 小麦赤霉病数据管理系统 …… 上海市计算技术研究所 上海农业科学院植物保护研究所
7. 应用于智能绘图仪上的一种自动开窗技术…… 上海飞机制造厂
8. 高考试卷命题质量的计算机辅助分析…… 华东师范大学
9. 热介质循环养护坑微机控制系统…… 同济大学
10. 文件档案管理系统…… 上海机械学院
11. 吊风扇电气参数智能测试仪…… 上海科学技术大学
12. 船舶营运统计分析与预测教学系统…… 上海海运学院

微机图形显示接口和软件设计

上海市计算技术研究所　周　正　周光华　严洪范　茅培生　蒋静华

在模式识别系统中常常要用到图形显示器，一般的光栅扫描图形显示器都需要一个大容量的帧存贮器，其价格昂贵，对于一些以微机构成的小型模式识别系统来说，这常常会成为一个难以解决的问题。

本文介绍了一种在056微机系统上配置的简易光栅扫描式图形显示器所用的硬件和软件。该显示器实际上只是一个显示接口，它以内存为刷新存贮器，显示存贮在内存中"光栅数组"里的图形，不需要另外的帧存贮器。有了该显示接口只需一台普通电视机即可显示图形，因此造价低廉。显示接口配备了基本图形软件，用户可以直接调用这些软件，因此使用也很方便。

基本显示软件虽然数量不多，但经灵活搭配使用，仍然会有较强的功能，既可显示图形数字化器输入的图形，也可显示应用程序处理过的或生成的图形。改变窗口，可以任意比例显示图形，或显示全貌或显示细节。在同一屏上可以开几个观察口，同时显示几"小块"图形。可同时定义几个光栅数组，轮翻显示得到"动态"图形。

目前，该接口已用于一个检测复杂外形的小型模式识别系统中，效果良好。

本文所介绍的基本图形软件设计方法具有一般意义，该方法同样可适用于接入056系统的任意一种光栅扫描图形显示器，可用来开发系统软件，使之具有较强的功能。

一、概　述

要显示的图形以矩阵的形式存贮在内存里，我们把它称为"光栅数组"。若将一幅图像分成m行n列，则光栅数组的形式如下：

$$\begin{bmatrix} f(1,1) & f(1,2) & f(1,3)\cdots f(1,n) \\ f(2,1) & f(2,2) & f(2,3)\cdots f(2,n) \\ \vdots & \vdots & \\ f(i,1) & f(i,2) & f(i,3)\cdots f(i,n) \\ \vdots & \vdots & \\ f(m,1) & f(m,2) & f(m,3)\cdots f(m,n) \end{bmatrix}$$

上面的数组中，$f(i,j)$表示第i行、第j列上那个像素的灰度等级。倘f取2^1级，就是显示只有黑白两级灰度的图形。倘f取2^4级，就是显示有16个灰度等级的图像。

若电视行扫描正程时间为T_H，场扫描正程时间为T_v，并且像素的灰度级为2^G级，计算机读出一个字节的时间为t_d，则光栅数组的行数和列数应满足下列关系：

$$n\leqslant\left[\frac{T_H}{t_d}\right]\left[\frac{8}{G}\right],\quad m\leqslant\left[\frac{T_e}{T_H}\right]$$

式中[ ]表示截去小数取整。

显而易见，n 决定了图像的分辨率。考虑到电视是一帧分二场隔行扫描的，利用这一特点，我们采用"隔列显示"的方法，将 n 提高一倍。即一帧中的二场分别交替地显示以下两个光栅数组(我们分别称它们为偶数组和奇数组)：

$$\begin{bmatrix} f_e(1,1) & f_e(1,2) & f_e(1,3) & \cdots & f_e(1,n) \\ f_e(2,1) & f_e(2,2) & f_e(2,3) & \cdots & f_e(2,n) \\ & \vdots & & & \\ f_e(m,1) & f_e(m,2) & f_e(m,3) & \cdots & f_e(m,n) \end{bmatrix}$$

$$\begin{bmatrix} f_Q(1,1) & f_Q(1,2) & f_Q(1,3) & \cdots & f_Q(1,n) \\ f_Q(2,1) & f_Q(2,2) & f_Q(2,3) & \cdots & f_Q(2,n) \\ & \vdots & & & \\ f_Q(m,1) & f_Q(m,2) & f_Q(m,3) & \cdots & f_Q(m,n) \end{bmatrix}$$

在屏上看到的将是如下一幅 $2n\times m$ 个像素的图像：

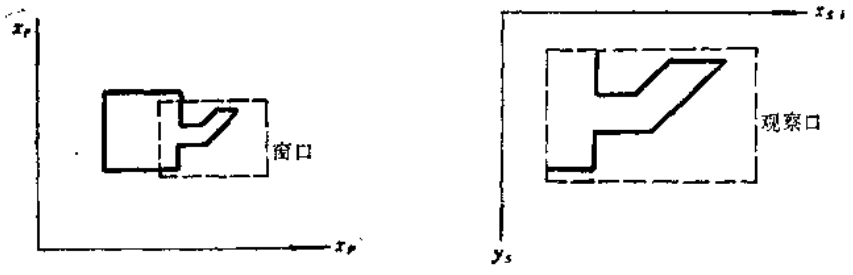
$$\begin{bmatrix} f_e(1,1) & f_Q(1,1) & f_e(1,2) & f_Q(1,2) & \cdots & f_e(1,n) & f_Q(m,n) \\ f_e(2,1) & f_Q(2,1) & f_e(2,2) & f_Q(2,2) & \cdots & f_e(2,n) & f_Q(2,n) \\ & & \vdots & & & & \\ f_e(m,1) & f_Q(m,1) & f_e(m,2) & f_Q(m,2) & \cdots & f_e(m,n) & f_Q(m,n) \end{bmatrix}$$

在我们的接口中，$T_H=52\,\mu s$，$T_e=18.7\,ms$，$t_d=3.2\,\mu s$，$G=1$，因此 $n\leqslant128$，$m\leqslant290$。我们实际所采用的图形分辨率为 256×256。由于 $G=1$，所以我们的显示接口只显示黑白两级灰度的图形。

接口硬件所完成的工作是：用 DMA 传送方式与电视扫描同步地从内存里读出光栅数组。电视扫描一行，接口电路从光栅数组里读出一行数据。回扫期间，停止读出，直至电视扫描下一行，再开始读数组的下一行。电视扫完一场，接口电路也读完了一个光栅数组，一个 DMA 块传送周期结束。程序重新初始化接口寄存器，俟场回扫结束后，重新开始下一个 DMA 块传送周期，读出另一个光栅数组。如此交替反复地读出奇偶两个光栅数组，在屏幕上就显示出稳定的图形。

基本图形软件完成的工作是：

1. 显示设备管理：初始化接口寄存器，启动显示通道；



(a) 用户坐标　　(b) 显示屏坐标

图 1　开窗命令：把窗口内的图形送到观察口内去显示，窗口和观察口由用户指定

2. 把用户提供的基于点的坐标的图形数据转换成光栅数组，完成“扫描转换”。

基本软件提供了画点、画线以及开窗命令。用户只需给出基本点或直线的两个端点在他自己所定义的任意坐标系里的坐标值，并指定窗口，观察口的位置和大小(见图 1)，即可在显示屏上，他任意指定的观察口看到他的坐标上的窗口内的由点或直线所组成的图形。这部分软件实际上包括了平移、定比例、剪取及扫描转换等功能。开窗口命令的对图形作几何变换的能力，在一定程度上弥补了硬件分解力不高的缺点。

基本图形软件所用的方法具有一般意义，对于高性能的带图像存贮器的光栅扫描显示器可以同样方法配置基本图形软件。

二、显 示 接 口

整个显示接口装在一块 12×6.75 in 的印刷板上，直接插在 056 微机系统的机架上，所用器件皆是 74 系列的器件。下面介绍该显示接口板的工作原理及结构。

1. DMA 数据传送过程

在我们所用的微机系统中有一块 DMA 传送控制板，它提供了传送控制信号。如系统中无此板，可在显示接口板上附加一些电路，直接从总线上产生出所需的 DMA 传送控制信号，所用方法在一般微机教材里皆有述及，这里不赘述。具体电路可参阅文献[4]。

表 1 显示接口所用的 DMA 传送控制信号(其中“/”表示低电平有效)

符　号	功　能
XFEQ RQ/	传送请求(来自显示接口)
XFEQ ACK/	传送回答(来自 DMA 板)
OUT PORT 0/	BASE+0 口的输出选通(来自 DMA 板)
DATA OUT 0/ ～ DATA OUT7/	八根输出数据线(来自 DMA 板)
RESET INTERRUPT/	块传送结束(来自 DMA 板)

显示接口所用的 DMA 传送控制信号见表 1。具体传送过程如下：

1. 程序初始化 DMA 板上的长度、地址及控制寄存器。

2. 程序用一条 BASE+0 口的输出指令将显示控制字送入显示接口的控制寄存器，启动读光栅数组。

3. 显示接口向 DMA 板发传送请求 XFER RQ/。

4. DMA 控制板将地址寄存器所指出的地址中的数据放到 DATA OUT 0～7/ 上，并产生请求回答 XFER ASK/。同时地址寄存器加 1，长度寄存器减 1。

5. 在电视行扫描正程期间每隔 3.2 μs 重复一次 3，4。直至长度寄存器为 0，DMA 板向 CPU 发 DMA 块传送结束中断请求。

6. CPU 响应中断，程序转入显示中断服务程序，该程序以一条指向 BASE+9 口的输出指令开始。该指令使 DMA 板产生 RESET INTERRUPT/ 信号，供显示接口结束一场

图形的显示。程序其后重复1的工作，返回主程序。

2．时序电路

我们这样分配时间：每场分为320行(帧扫描正程占290行，逆程占30行)，每行分为20段(行扫描正程占16段，逆程占4段)，每段分为8点，每一点就是基本时钟的时间。若基本时钟取400 ns，则一段所对应的时间为3.2 μs，一行所对应的时间为64 μs(其中行正程为51.2 μs，逆程为12.8 μs)，一场所对应的时间为20 ms(其中场扫描正程为18.5 ms，逆程为1.5 ms)。

若以基本时钟作为显示一个像素的时间，则在一段所对应的时间内可以显示8个像素，在这段时间内DMA通道也恰好可以从内存读出一个字节的图形数据，在一行所对应的时间内可以读出16个字节，共含有128个像素。再采用上述交替读出奇偶两个数组的办法，在一行上就可以显示256个像素。

由以上分析可见，这样的时间分配既满足了电视扫描制式的要求，也满足了DMA通道传送的时间要求。只要电视扫描也用该基本时钟来定时，则图形数据的读出与电视扫描就同步起来了。

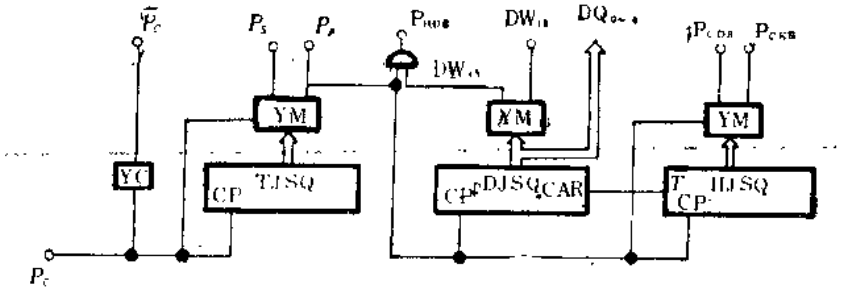


图2　时序电路

图2就是时序电路的框图，对图中的符号作一些简单的说明：

TJSQ，DJSQ，HJSQ分别为点计数器、段计数器和行计数器。P_0为基本时钟。

TJSQ为以P_0为计数脉冲的八进制计数器，记其节拍电位为TW_0～TW_7。通过译码器产生$P_s=TW_6\cdot P_0$，$P_p=TW_7\cdot P_0$，供控制电路用。

DJSQ是一个20进制计数器，以P_p为计数器脉冲，记其节拍电位为TW_0～DW_{19}。译码器产生$P_{HDB}=DW_{15}\cdot P_p$，输出给电视机作行同步脉冲。构成DJSQ的五个触发器的状态DQ_0～DQ_4以及节拍电位DW_{18}引出供控制电路用。

HJSQ是一个320进制计数器，与DJSQ串接，并行记数，记其节拍电位为HW_0～HW_{319}。译码器产生$P_{CDB}=HW_{290}\cdot P_p$，输出给电视机作场同步脉冲。$P_{CKS}=HW_{319}\cdot P_0$，标志场扫描正程开始，供控制电路用。

此外，将P_0经过100 ns的延迟产生$\bar{P}_0$，用它来产生给移位器的移位脉冲。

3．接口寄存器

接口寄存器有两个：控制字寄存器KZJCQ和数据寄存器SJJCQ，它们皆是八位寄存器，从DATA OUT 0～7线上接收DMA板送来的数据，框图见图3。

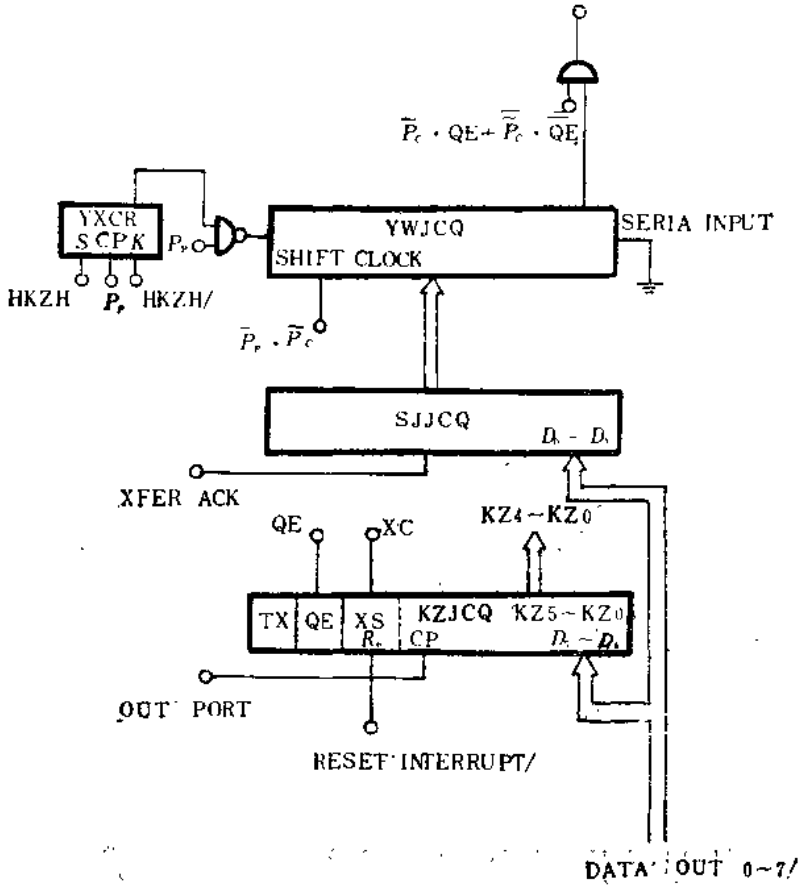


图3 接口寄存器与移位器

程序以一条指向 BASE+0 的输出指令写 KZJCQ，因此该寄存器的 OUT PORT0/ 为选通脉冲。把 $\frac{n}{8}-2$ 写入低五位 KZ0～KZ4，其中 n 为光栅数组的列数，原因见控制电路一节。第五位 XS 为显示标志位，由程序置，传送结束脉冲清。第六位 QE 为奇偶场标志，由程序写，偶数场置 1，奇数场置 0。第七位为停显位，由程序写 0，作用在软件部分介绍。

数据寄存器是移位器与数据线之间的缓冲寄存器，它以传送回答 XFER ACK/ 为选通脉冲，暂存数据线上的图形数据，然后用 P_s 将这数据送给移位器，实现显示与数据读出之间的同步。

4．控制电路

主要的控制触发器有两个：场工作触发器 CKZH 和行工作触发器 HKZH。在两个触发器皆置位的条件下，就向 DMA 发传送请求 XFER RQ=P_s·HKZH，每隔 3.2 μs 读出一个字节的图形数据，见图 4。

CKZH 置位表示正在读出一个光栅数组。置位条件为 XS=1，由 P_{CKS} 置。由来自 DMA 板的传送结束脉冲 RESET INTERRUPT/ 复位。

HKZH 置位表示正在读光栅数组中的一行。置位条件为 CKZH·DW_{18}，由 P_s 置。因为每一行的行扫描正程从 $DW_0\cdot P_s$ 时刻开始的，此时应将下一行的第一段数据打入移位

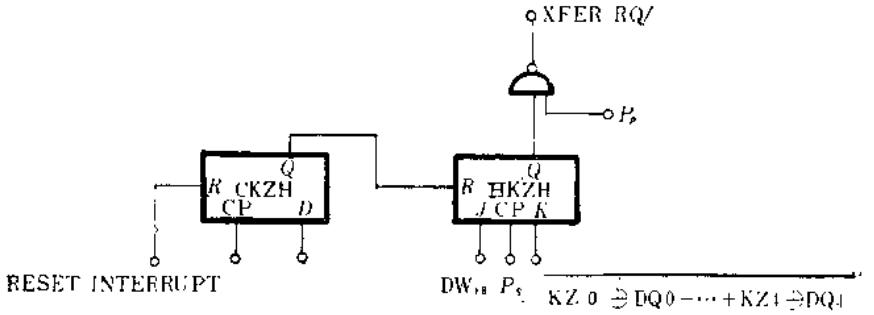


图 4　控制电路

器，因此在前一段即第 18 段的 P_5 时刻置起 HKZH，在第 19 段的 P_9 时刻发出数据请求 XFER RQ/，在 0 段的 P_9 尚未到时，数据已送到 DATA OUT0～7/ 上，由 XFER ACK/ 将其送入 SJJCQ，俟 P_9 一到，恰好将此数据从 SJJCQ 打入移位器，开始一行显示。复位条件为：已读完光栅数组中的一行，即发了 $n/8$ 次数据请求，该条件就是 $\overline{KZ0 \oplus DQ0} + \cdots + \overline{KZ_4 \oplus DQ4}$ 为真，由 P_5 复位。此后停止请求数据，俟电视回扫到行首，再读下一行。

5．移位器及视频输出

移位器 YWJCQ 是一个八位的并入串出的移位寄存器，它在每一段 P_9 时刻并行地接收一个字节，然后以 $\overline{P_5}$ 为移位脉冲作七次移位，这样就把八个像素均匀地显示在屏上，见图 3。

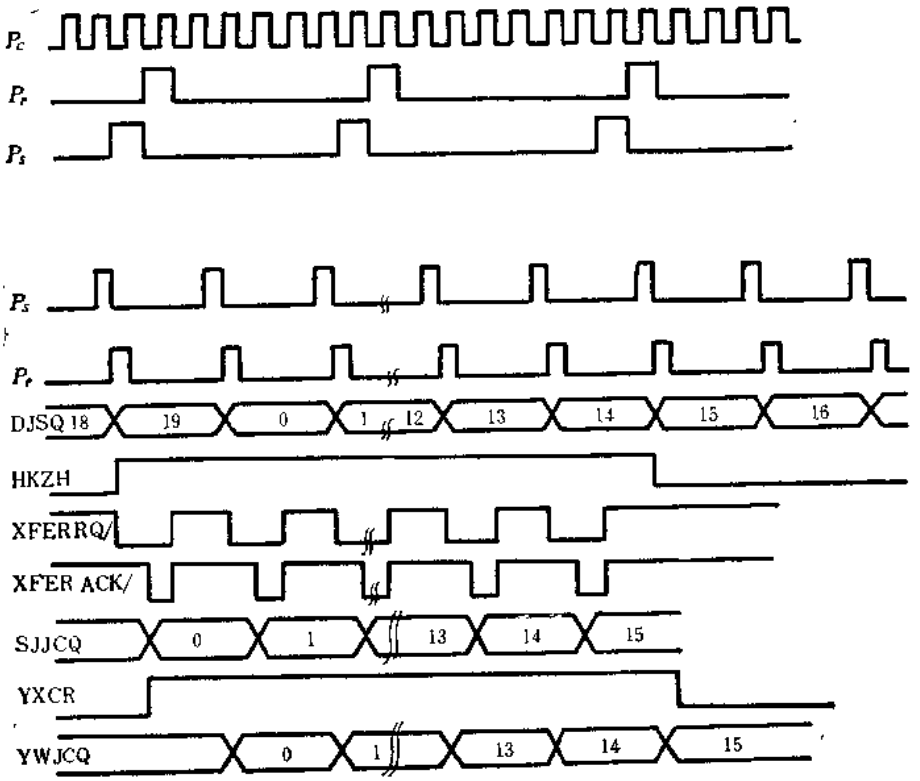


图 5　显示一行的时间图

接收并行数据条件是 $\overline{P_9 \cdot \mathrm{YXSR}}$ 为假，其中 YXSR 为允许输入触发器，它比 HKZH 迟一个段节拍电位置位和复位。因为上面已讨论过，$DW_6 \cdot P_9$ 时刻移位器才接收一行的第一段的数据，而 HKZH 在 $DW_{18} \cdot P_6$ 已置起了，因此必须另设一个允许输入触发器，它比 HKZH 迟一个段节拍电位置位和复位，由它来控制移位器的并行输入。

P_9 把数据并行输入移位器，这就在屏上显示了每段的第一个像素，后面再进行七次移位，显示后七个像素，因此可以 $\overline{P}_9 \cdot P_0$ 为移位脉冲。但又考虑到为了保证 $\overline{P}_9$ 的下降沿在 P_0 的上升沿之前，因此用 $\overline{P}_0 \cdot \overline{P}_9$ 作移位脉冲。

从移位器移出的数据需经过辉亮控制才送给监视器，以达到"隔列显示"的效果。偶数场用 $\overline{P}_0$ 辉亮图形数据，奇数场用 $\overline{P_0}$ 辉亮图形数据，交替地送奇偶光栅数组就得到分辨率提高了一倍的图形。

图 5 为显示一行的时间图。

三、基本图形软件

1. 图形变换

基本的图形变换主要是以下三种：平移，定比例和旋转。设图形上某点 P 在原坐标系里的坐标为 (x, y)，在变换之后的坐标系里的坐标为 (x', y')，并引入齐次坐标[2]，于是平移变换可表示成

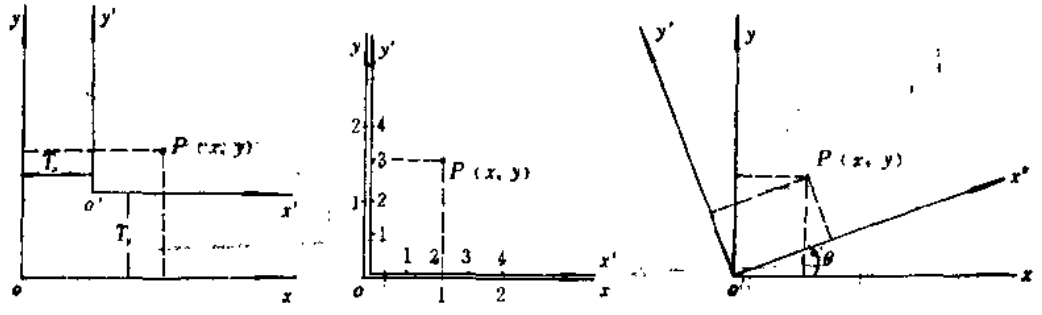
$$[x'\ y'\ 1] = [x\ y\ 1] \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -T_x & -T_y & 1 \end{bmatrix}$$

其中，(T_x, T_y) 为坐标原点移动的位置[图 6(a)]。定比例变换可表示成

$$[x'\ y'\ 1] = [x\ y\ 1] \begin{bmatrix} N_x & 0 & 0 \\ 0 & N_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

其中，(N_x, N_y) 为两个方向上坐标单位的比例[图 6(b)]。旋转变换可表示成

$$[x'\ y'\ 1] = [x\ y\ 1] \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ \sin\theta & -\cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



(a) 平移　　(b) 定比例　　(c) 旋转

图 6　三种基本图形变换

其中，θ 为坐标轴旋转的角度[图 6(c)]。

大部分常用的复杂图形变换，都可以分解成由这三种基本图形变换组成的变换序列，只要按序把构成复杂变换的那些基本变换的变换矩阵级联，即可以得到表示该复杂变换的变换矩阵。

2. 开窗剪取

(1) 开窗

在实际应用中，单用显示屏坐标来定义图形具有很大的局限性，并且也不方便。屏上的坐标只能用整数来表示，而且屏是有界的，要改变度量单位很不方便，因此我们引入两个坐标系，一个称为页坐标，这是用户用的定义图形数据的坐标，用户可采用便于自己使用的任意度量单位。另一个是屏坐标，它是与显示硬件有关的，以光栅单位为单位的光标，对于我们的显示器是 256×256。

我们把屏上用于显示图形的区域称为观察口，而把观察口对应在页坐标上的映像称为窗口，见图 1。

这样，用户可以根据实际问题，任意选定一种方便的页坐标，并在上面定义图形。当他需要观察图形时可以说："把页坐标上的这个区域送到屏上去显示。"这时他就定义了窗口和观察口(全屏)。显示软件就根据用户以窗口和观察口的位置和大小所说明的页⟶屏的变换，把页坐标上的图形数据变换到屏坐标上去，并装配成"光栅数据"进行显示。

用窗口说明图形变换的方法可以给用户带来很大方便，若将窗口开得很大，用户就可以看清图形的全貌；若要看清某些局部细节，可以将窗口开在页坐标上所需部位。用户还可以同时在屏上开两个以上的观察口分别对应于不同的窗口，这样就可以比较页坐标不同部位上的图形，甚至不同页坐标上的图形。

记$(x_P,\ y_P)$为图形上 P 点的页坐标，而 $x_s,\ y_s$ 为该点的屏坐标值。设 W_L 为窗口左边界，V_L 为观察口左边界，W_R 为窗口右边界，V_R 为观察口右边界，W_T 为窗口上边界，V_T 为观察口上边界，W_B 为窗口下边界，V_B 为观察口下边界，L_1 为窗口宽，L_2 为观察口宽。

我们可以用三个变换构成一个变换序列，把窗口变为观察口，见图 7。

(1) 平移窗口至页坐标原点：

$$T_x=-W_L,\quad T_y=-W_B$$

(2) 坐标定比例：

$$N_x=L_2/L_1,\quad N_y=-L_2/L_1$$

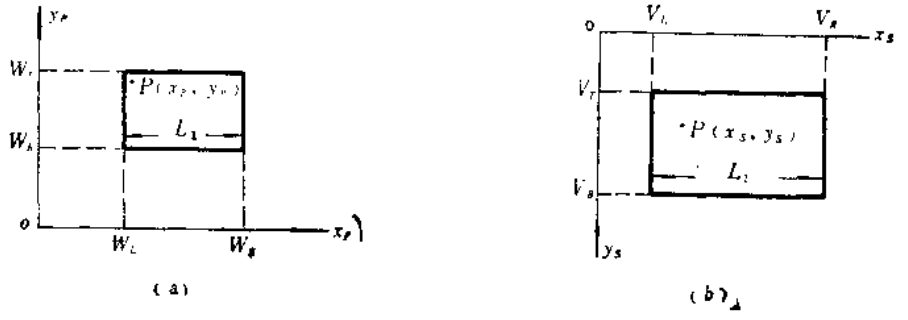


图 7　用窗口和观察口表示的从页到屏的变换

(3) 平移观察口至所定义的位置:

$$T_x=-V_L,\ T_y=-V_B$$

将三个变换的变换矩阵级联,可得到开窗变换公式:

$$[x_s y_s 1]=[x_p y_p 1]\begin{bmatrix}1&0&0\\0&1&0\\-W_L&-W_B&1\end{bmatrix}\begin{bmatrix}L_2/L_1&0&0\\0&0&L_2/L_1\\0&0&1\end{bmatrix}\begin{bmatrix}1&0&0\\0&1&0\\V_L&V_B&1\end{bmatrix}$$

得

$$\begin{cases}x_s=(x_p-W_L)L_2/L_1+V_L\\y_s=V_B-(y_p-W_B)L_2/L_1\end{cases}$$

上述开窗变换的算法很易编成子程序,在此不再赘述。

由于显示硬件没有带光笔,因此我们用键盘进行人机对话输入窗口值和观察口的值。用键盘输入窗口值虽不象光笔直接点屏那样直观,但我们在软件中加入了输入窗口值后自动显示新窗口值在现行窗口中的位置的功能,以便用户确认新窗口值是否合适。这样一来,用户用键盘输入窗口值还是比较直观的。观察口在大部分情况下为全屏,因此为了便于使用,在观察口为全屏的情况下可不必说明。

(2) 剪取

在从页坐标到屏坐标的变换过程中有这么一个问题,即有时为了看清某些局部细节而把窗口开得很小时,有很大一部分图形不在屏上,必须把它"剪"去,我们把这个工作称为"剪取"。剪取和变换的次序应如何安排呢?若先进行变换,则需将页坐标上全部数据都进行变换,而数据中有很大一部分不在屏上,这样做法浪费很大,如果在变换之前确定了哪些点和线在屏内,哪些点和线是不可见的,就可以少做许多不必要的变换。要做到这一点,只需将页坐标上的图形对着窗口进行剪取,然后将窗口内的图形数据变换到屏坐标上。

剪取的方法有很多种,对点的剪取比较简单,只要判别点是否在界内,而对线的剪取就比较复杂。目前我们采用 Cohen 算法,它仅适用于与坐标轴平行的矩形剪取边界[2]。

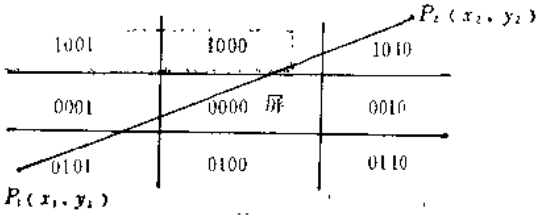


图 8

这种算法是这样的:延长剪取区域的边界把未剪取过图形占据的空间分成九个区域,每个区域以一个四位代码标志,见图 8。

这四位代码意义分别为:第一位为 1,表示该区域在顶边之上;第二位为 1,表示该区域在底边之下;第三位为 1,表示该区域在右边之右;第四位为 1,表示该区域在左边之左。

若被剪取的线段 P_1P_2 的端点坐标为 $P_1(x_1,y_1)$, $P_2(x_2,y_2)$,那么首先对线段的两个端点进行配码,即将端点所在区域的代码赋给端点,得到 C_1, C_2 配码之后就进行测试。

先判别整条线段是否全在界内,若是,则应 $C_1=C_2=0$;若不全在界内,那么再判别线段的两个端点是否在界外同侧,若是,则应 $C_1 \wedge C_2 \neq 0$,这种线段肯定全在界外,可丢弃掉;若也不是,就必须先进行细分,然后再测试。

细分:先测试始点是否在界外,若在界外求出线段与某边界的交点,代替原始点并给新始点进行配码,再重复上述测试过程,这样直至始点在界内。然后再用同样的方法处理终点。最后留下的必然只是线段的界内部分。

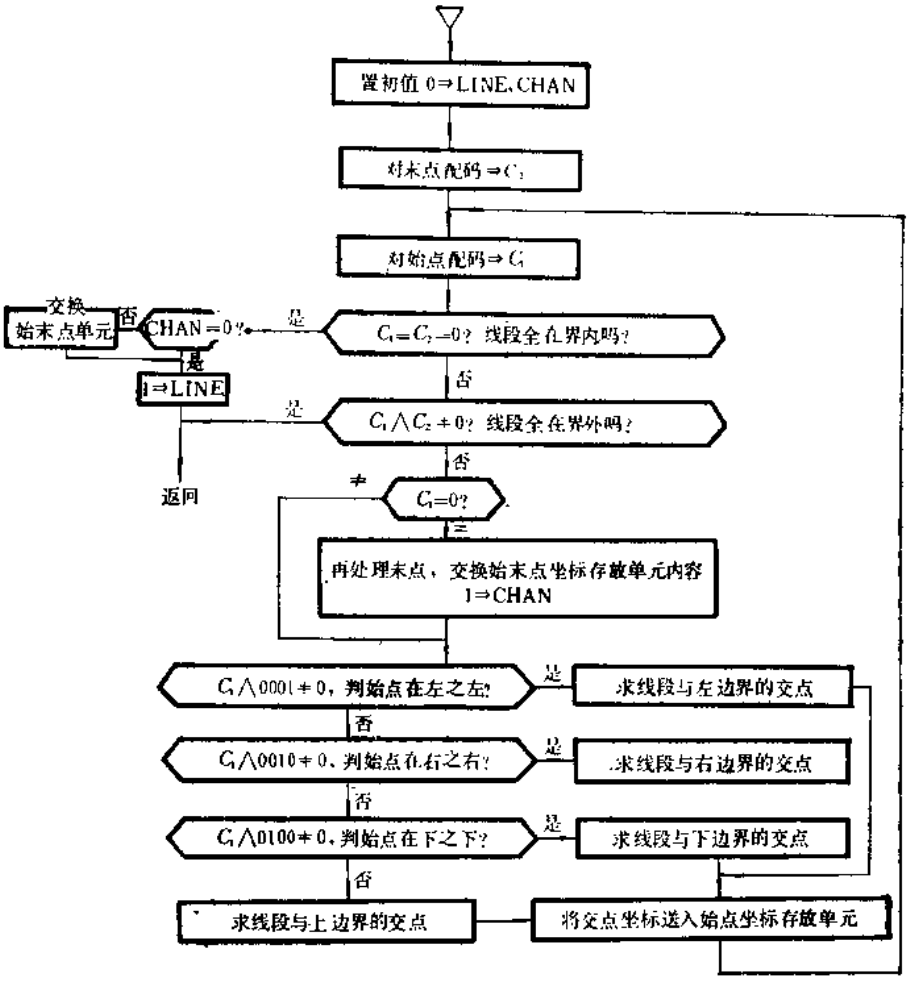


图 9　剪取子程序框图

Cohen 算法的优点是最多经过二至三次测试一定能求出界内的线段。由于子程序的递归调用，各种复杂的情况无一遗漏，效率较高。但必须注意的是这种算法在剪取边界与坐标轴不平行的情况下不能使用。

剪取子程序框图如图 9 所示。

该子程序的输入值为线段两个端点的坐标，程序返回一个标志 LINE，其值若为假，表示该线段全在界外，若为真，表示该线段有界内部分，其两个端点的坐标就在原来存放始末点坐标的单元内。框图中的 CHAN 为程序内部用的标志，其值若为真，表示现在正在处理末点。

3．画点，画线

(1) 画点

从图形数据库中取出的点的数据是页坐标，经过剪取和变换后就得到了点的屏坐标。对于随机扫描的显示器，只需将点的屏坐标装配成显示指令，就可以在屏上显示出点来。而现在是光栅扫描显示器，必须把点的屏坐标按显示硬件的要求装配“光栅数组”，即经过扫描转

换，才能在屏上显示出一个点来。

若一幅图的图宽为 m，图长为 n，则光栅数组为 $m\times n$ 的矩阵，根据目前使用的显示器的要求，我们所设计的"光栅数组"为 256×256，其中每一个像素的灰度值只占一个二进制位。056 微机一个字节为 8 位，因此光栅数组可以是一个 [1:256, 1:32] 的二维数组存放在内存里，为与显示接口按行取光栅数组的格式相对应，该二维数组采用按行存放的结构。按照这种存贮结构我们不难得到实现扫描转换的方法。

一般可用"填充"的方法来实现扫描转换，即在装配光栅数组之前，先把光栅数组充零，然后一个像素一个像素地填进去。

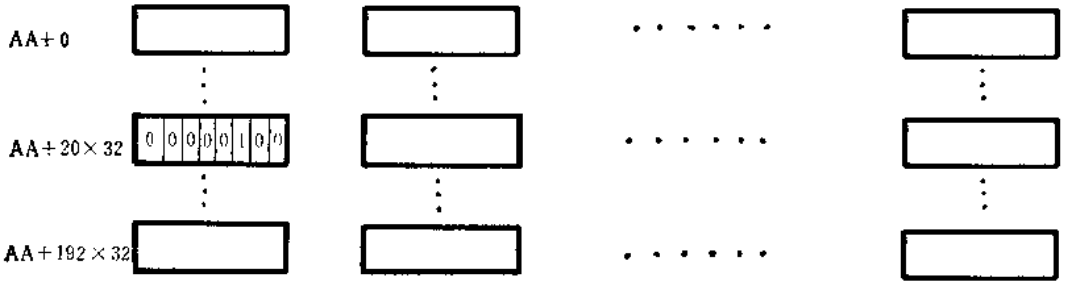


图 10　填充光栅数组

设某一像素的屏坐标为 x，y。将其灰度的二进制数值转换为布尔量并记为 g，设 g 占 G 个二进制位。存放光栅数组的二维数组的首址为 AA，上下界为 [1:M, 1:N]，每一个元素可以存放 K 个像素，则填充一个像素的算法可用一些记号形式地表示如下：

$$D(\mathrm{AA}+N*Y+X/:K):=D(\mathrm{AA}+N*Y+X/:K)\vee g\leftarrow G*(K-1-X\ \mathrm{MOD}K)$$

其中，记号 $D(P)$ 表示地址为 P 的内存单元中的内容，现在定义为布尔量。

$A\leftarrow B$ 表示算术左移运算，其运算优先级高于"∨"运算。其中，A 为被移位的布尔变量，B 为移位次数。其它记号的意义与 ALGOL 语言中相同。

在我们现在的情况下，$M=256$，$N=32$，$G=1$，$K=8$，g 恒为"00000001"。填充公式简化为：

$$D(\mathrm{AA}+32*Y+X/:K):=D(\mathrm{AA}+32*Y+X/:K)\vee\text{"00000001"}\leftarrow(7-X\ \mathrm{MOD}8)$$

例 $X=5$，$Y=20$，则

$D(\mathrm{AA}+640):=D(\mathrm{AA}+640)\wedge 00000100$，见图 10。

(2) 画线

画线子程序的框图见图 11。其中，"用 DDA 算法装配光栅数组"一框，是指用 DDA 算法求出构成该线段所有点的屏坐标的值，然后每一点都调用一次上述扫描转换子程序，这样在光栅数组中就装配出所需的线段来。下面介绍一下 DDA 算法。

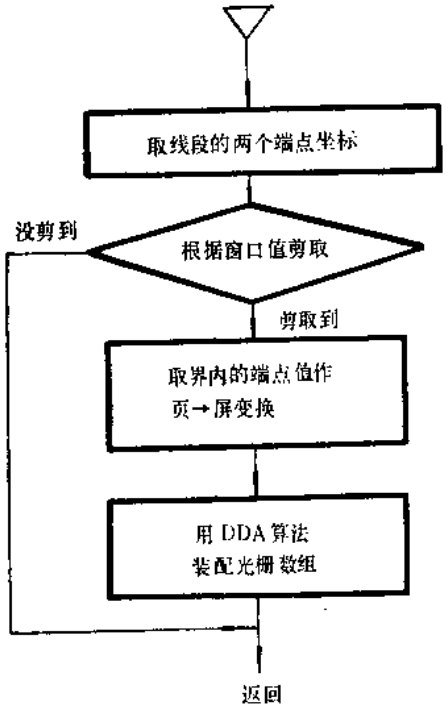


图 11　画线子程序框图

直线段的方程用参数形式表示是最方便的。设直线起点坐标为 (x_1, y_1)，终点坐标为

(x_2, y_2)，则直线段可以表示为：

$$\begin{cases} x = x_1 + (x_2 - x_1)t \\ y = y_1 + (y_2 - y_1)t \end{cases} \quad 0 \leqslant t \leqslant 1$$

可是用这公式来计算直线上的点的坐标要做乘法，比较麻烦，可以将该方程写成微分形式，并记

$$\Delta x = x_2 - x_1$$

$$\Delta y = y_2 - y_1$$

则得

$$\begin{cases} dx = \Delta x\, dt \\ dy = \Delta y\, dt \end{cases}$$

$$\frac{dy}{dx} = \frac{\Delta y}{\Delta x}, \quad \text{边界条件: } y|_{x=x_1} = y_1$$

因为 $\Delta y/\Delta x$ 为常数，可以用差商精确地代替微商，于是有：

$$\frac{y_{i+1} - y_i}{x_{i+1} - x_i} = \frac{\Delta y}{\Delta x}$$

解方程得

$$\begin{cases} y_{i+1} = y_i + h_i \cdot \Delta y/\Delta x \\ h_i = x_{i+1} - x_i \end{cases}$$

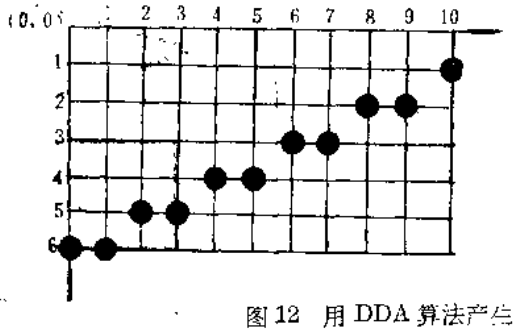
取 h_i 为增量单位，则可以得到下列递推公式：

$$\begin{cases} x_{i+1} = x_i + 1 \\ y_{i+1} = y_i + \Delta y/\Delta x, \ y|_{x=x_1} = y_1 \end{cases}$$

为了使显示点取得尽可能地密，取变化量大的那个坐标为单位增量，每次增加一个光栅单位，从起始点开始，逐点递推，直至终点。

因为显示器只接受整数屏坐标，因此送去填充光栅数组的 x, y 坐标必须取整，用 x_{display} 或 y_{display} 表示。

用此原理，我们只需要知道线段的两个端点坐标，即可在光栅数组中填出整条线来，见图 12。当然还有其它产生线的方法，但 DDA 算法产生的线的质量较高。



$$\begin{cases} x_1 = 0 \\ y_1 = 6 \end{cases} \quad \begin{cases} x_2 = 10 \\ y_2 = 1 \end{cases}$$

图 12　用 DDA 算法产生线

4. 显示通道的启动

显示通道的启动方式是由接口硬件所决定的。我们的显示接口用的是 DMA 传送方式。

为了尽可能提高分辨率，显示接口采用了“隔列扫描”的方式（见概述部分），因此在启动显示之前，必须先把已配好的光栅数组拆成奇偶两个数组，即将数组相邻两个字节的奇数位