基于 T6963C 的 LCD 模块 C51 绘图函数库设计

Design of C51 Plotting Function Library of LCD Module Based on T6963C

陈延文[1]　王登第[2]　李海峰[1]

(天水师范学院计算机科学系[1],甘肃 天水 741000;西安天城电力仪器设备公司[2],陕西 西安 710075)

摘　要:在基于 LCD 控制器 T6963C 的 LCD 模块上显示几何图形,需要把计算机图形学中的一些绘图算法移植到显示模块上。通过对 T6963C 控制器工作原理的分析,采用 C51 语言编写绘点函数,并根据图形生成算法,给出了在此类模块上绘制常用几何图形的 C51 函数,最后生成 C51 函数库。此函数库的设计提高了实际产品研发效率。

关键词:LCD 模块　图形生成算法　C51 函数库　几何图形　Bresenham 算法

中图分类号:TP334.4　　**文献标志码**:A

Abstract: In order to display geometric graphic on the LCD module based on T6963C LCD controller, it is necessary to transplant the graphic building algorithm in computerized graphics onto display module. Through analyzing the operational principle of T6963C controller, by adopting C51 language to program the plotting point function, and create the algorithm in accordance with the graphic. The C51 functions for creating commonly used graphics on such module are given, and the C51 function library is generated. Thus the efficiency for developing practical products is enhanced.

Keywords: LCD module　Graphic generation algorithm　C51 function library　Geometric graphic　Bresenham algorithm

0　引言

T6963C 是日本 Toshiba 公司设计的一款 LCD 控制器,多用于中小规模的液晶显示器件,它常被装配在液晶显示模块上,以内嵌控制器型图形液晶显示模块的形式提供给用户。由于该类 LCD 模块较易与单片机接口,目前在智能仪器仪表上应用广泛。本文就如何在此类 LCD 模块上处理几何图形进行了探讨;给出了在与 51 系列单片机接口时,用于显示常用几何图形的关键 C51 函数,这些函数可以用库的形式提供给用户。

1　T6963C 编程

T6963C 控制器内部自带 128 个字的字符发生器 ROM,可以控制访问多达 64 kB 的外部显示 RAM(简称显示 RAM);且通过指令可以很容易地为文本区、图形区及外部字符发生器 CGRAM 分配内存空间。在该类 LCD 显示器上编程显示文本、图形信息,本质上就是对其显示 RAM 进行读写。因此,必须弄清楚显示信息如何在显示 RAM 中布局。

1.1　显示信息在显示 RAM 中的布局

LCD 控制器可以工作于单屏和双屏结构。对于单屏结构,它可以在 64 kB 显示 RAM 的任何位置为文本区、图形区和外部 CGRAM 分配空间;而对于双屏结构,64 kB 显示 RAM 分为 LCD-I 和 LCD-II 两部分,LCD-I 占用 0000H ~ 7FFFH 的地址空间,LCD-II 占用 8000H ~ FFFFH 的地址空间。LCD-I 部分可以为文本区、图形区和外部 CGRAM 分配空间;在 LCD-II 部分,地址空间的地址除了最高位与 LCD-I 不同外,其余位要求与 LCD-I 完全相同[1]。如对于单屏结构 128 × 64(即 LCD 模块的一屏,横向有 128 个像素点,纵向有 64 个像素点)的 LCD 模块,假设用控制指令已经将文本区的首地址设为 0000H,图形区首地址设置为 1000H,外部字符发生器 CGRAM 的首地址设置为 2000H,则显示 RAM 的划分如图 1 所示。

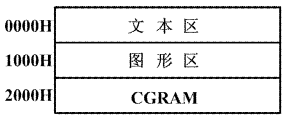

图 1　显示 RAM 的划分

Fig. 1　Determination of the display RAM

控制器 T6963C 可以在文本模式和图形模式下工作。在文本模式下,文本区每行可显示 16 个字符,每行字符占用显示 RAM 中的 16 个字节,一屏可显示 16 ×

甘肃省教育厅 2008 年度第二批科研基金资助项目(编号:0808B - 04)。

修改稿收到日期:2010 - 07 - 19。

第一作者陈延文,男,1966 年生,1989 年毕业于沈阳理工大学计算机科学系,获学士学位,副教授;主要从事嵌入式系统应用技术、并行算法方面的研究与教学工作。

8 = 128 个标准点阵字符,占用的显示 RAM 是 128 个字节。此时,LCD 模块上的显示位置与显示 RAM 单元的对应关系如图 2 所示。

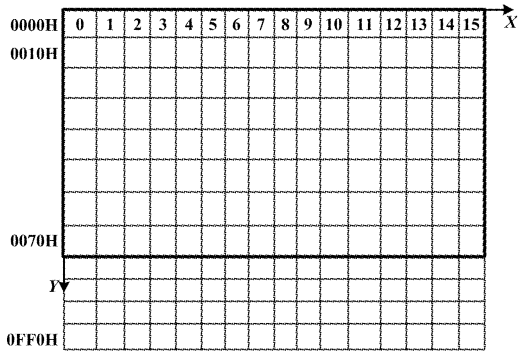

图 2　文本区显示位置与显示 RAM 单元的对应关系

Fig. 2　Text area display position vs. display RAM unit

图 2 中,*X*-*Y* 为屏幕坐标系,以字符为单位;粗线框表示一屏占用的显示 RAM,数字表示显示 RAM 的地址。为了实现滚屏效果,可以将显示的若干屏字符内容放在 0000H ~ 0FFFH 的显示 RAM 中,然后通过指令修改文本区的首地址来实现。另外,可以通过指令设置文本的显示特征,如闪烁或反白等效果,此时需要借用图形区中的显示 RAM 空间作为文本的特征区,用于存储文本特征信息[1]。

对于图形模式,一屏可显示 $128 \times 64 = 2^{13}$ 个像素点,显示一行像素点需要占用 16 个字节的内存空间,一屏数据占用 $16 \times 64 = 2^{10}$ 个字节。此时显示器上点阵的显示位置与显示 RAM 中对应位置如图 3 所示。

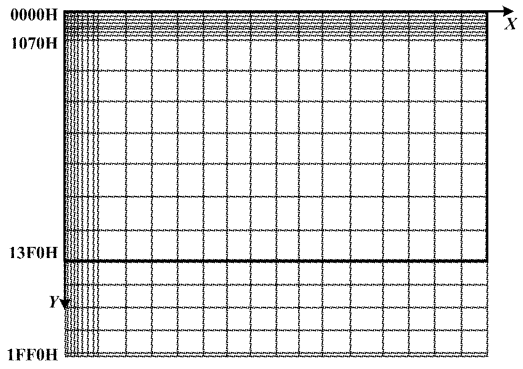

图 3　图形区显示位置与显示 RAM 单元的对应关系

Fig. 3　Graphic area display position vs. display RAM unit

图 3 中,*X*-*Y* 表示屏幕坐标系,以像素为单位,对于一个字节对应的点阵位置,字节的低位在 *X* 轴的正方向;粗线框表示一屏;*Y* 坐标方向的数字表示一行点阵在显示内存的中的首地址。同样地,将显示的若干屏图形点阵信息存储在显示 RAM 的 1000H ~ 1FFFH 空间中,然后通过指令修改一屏的首地址,可以实现滚屏效果。

外部字符发生器 CGRAM 的起始地址为 2000H,可以将用户生成的点阵字库存放到 2000H 起始的显示 RAM 中。对于基于 T6963C 的其他显示模块显示位置与显示 RAM 中位置的对应关系类似,不再赘述。

1.2　T6963C 的指令系统

T6963 的指令集很简单,其指令形式包括无参数、一个参数和两个参数三种,每条指令都是先送参数(如果没有则不用送),再送指令码。在每次操作之前都要先检测其状态字。有关指令的详细内容请参阅参考文献[1]。

2　LCD 绘图函数

目前,有关光栅图形显示器二维图形的生成算法已相当成熟,大部分算法都是以绘制基本图形为基础的。这里简要介绍一下绘制基本几何图形的算法。

2.1　画点的 C51 函数

画点是最基本的程序,所有其他几何图形的绘制都要调用画点函数来完成。画点可以用 T6963C 的“数据一次读/写方式”指令(指令格式为“D1H,C4H”,其中 D1H 为要写的数据,C4H 为指令码)实现,也可以用位操作指令实现[1]。画点的 C51 函数中选用位操作指令。假设 8000H 为数据端口地址,8001H 为指令端口地址,画点的 C51 函数如下。

```
# include < absacc. h >
# define T6963_COM XBYTE[0x8001]
                                    //定义 T6963 命令端口地址
# define T6963_DAT XBYTE[0x8000]
                                    //定义 T6963 数据端口地址
  void Putpixel(int x,int y)
          //画点函数,(x,y)为点的坐标,以像素为单位
  {unsigned int tmp1;
   unsigned int tmp;
   unsigned char dat1,dat2;
   tmp1 = y * 16 + x/8;
                  //点坐标转换成显示 RAM 的单元地址
   tmp1 = tmp1 + 0x1000;
                                    //图形区开始地址为 0x1000
   dat1 = (char)(tmp1&0x00ff);              //低 8 地址
   tmp = (tmp1&0xff00) >>8;
   dat2 = (unsigned char)(tmp);             //高 8 位地址
   Write_COM_DAT(dat1,dat2,0x24);
                                    //设置图形区的单元地址
              //该函数向 T6963C 送 dat1、dat2、0x24 指令
```

```
    dat1 = 7 - x%8;              //求出字节内的位地址
    Identify_State01();          //该函数判断状态
    T6963_COM = dat1|0xf8;
      //位操作指令,将相应的位置1,即显示一个像素
}
```

2.2 Bresenham 直线生成算法

Bresenham 算法生成直线的基本思想如图 4 所示。

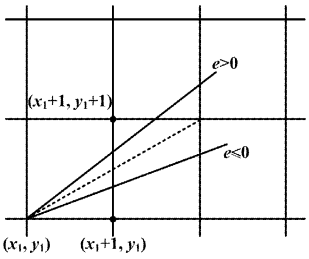

图 4　Bresenham 算法说明

Fig. 4　Illustration of Bresenham algorithm

图4 中,一个网格为一个单位,每个网格有四个交点;令 $e=k-\frac{1}{2}$。从 (x_1,y_1) 画直线,设直线斜率 $0\leqslant k\leqslant 1$[2],则下一个点的选择有两种可能,一种是选 (x_1+1,y_1+1) 点,另一种是选 (x_1+1,y_1) 点。当 $e\leqslant 0$[2],即所绘制的直线在虚线(虚线相当于基准线,其斜率为$\frac{1}{2}$)的下方时,下一个点选 (x_1+1,y_1);当 $e>0$,即所绘制的直线在虚线的上方时,下一个点选 (x_1+1,y_1+1)。

假设要在 LCD 模块上的 $P_1(x_1,y_1)$ 和 $P_2(x_2,y_2)$ 两点之间绘制直线,总共进行 (x_2-x_1) 步,每一步在所选择的网格交点位置绘制一个点,绘制出 (x_2-x_1) 个点后,即绘制出了 P_1P_2 直线。下面说明如何选择第 n 步绘制的网格交点位置。

P_1P_2 直线的斜率 $k=\frac{\Delta y}{\Delta x}$,其中,$\Delta y=y_2-y_1$,$\Delta x=x_2-x_1$,假设第 $n(n\in[1,x_2-x_1])$ 步选择网格交点 $(x+1,y)$,d_n 为该点与 P_1P_2 直线和直线 $x=x_1+n$ 的交点之间的距离,也称为误差。当 $d_n\leqslant\frac{1}{2}$时,则此点就是第 n 步选择的网格交点;当 $d_n>\frac{1}{2}$时,则第 n 步选择的网格交点为 $(x+1,y+1)$。其中,(x,y) 为上一步选择的网格交点坐标。为方便程序判断,设 $e_n=d_n-\frac{1}{2}$,则通过判断 e_n 的符号即可选择第 n 步绘制的网格交点。初始值 $e_1=k-\frac{1}{2}$,即:

$$e_1=\frac{\Delta y}{\Delta x}-\frac{1}{2} \tag{1}$$

当 $e_n\leqslant 0$ 时,选择网格交点 $(x+1,y)$,有:

$$e_{n+1}=\left[\left(e_n+\frac{1}{2}\right)+\frac{\Delta y}{\Delta x}-1\right]-\frac{1}{2}$$

即:

$$e_{n+1}=e_n+\frac{\Delta y}{\Delta x}-1 \tag{2}$$

当 $e_n>0$ 时,选择网格交点 $(x+1,y+1)$,有:

$$e_{n+1}=\left[\left(e_n+\frac{1}{2}\right)+\frac{\Delta y}{\Delta x}\right]-\frac{1}{2}$$

即:

$$e_{n+1}=e_n+\frac{\Delta y}{\Delta x} \tag{3}$$

为了方便程序处理,对式(1)、(2)、(3)等号两边同乘以 $2\Delta x$,消除除法运算,则式(1)变换为 $2e_1\Delta x=2\Delta y-\Delta x$,式(2)变换为 $2e_{n+1}\Delta x=2e_n\Delta x+2(\Delta y-\Delta x)$,式(3)变换为 $2e_{n+1}\Delta x=2e_n\Delta x+2\Delta y$。

因为该算法只与 e_n 的符号有关,令 $e'_n=2e_n\Delta x$,则可进一步化简。初值为:

$$e'_1=2\Delta y-\Delta x \tag{4}$$

当 $e'_n\leqslant 0$ 时,选择网格交点 $(x+1,y)$,且有:

$$e'_{n+1}=e'_n+2(\Delta y-\Delta x) \tag{5}$$

当 $e'_n>0$ 时,选择网格交点 $(x+1,y+1)$,且有:

$$e'_{n+1}=e'_n+2\Delta y \tag{6}$$

经过变换后,仅通过判断 e'_n 的符号,就可以推导出第 n 步绘制的网格交点。

2.3 绘制直线的 C51 函数

根据式(4)~(6),可以编写出基于 Bresenham 算法绘制直线的 C51 函数。此函数的原型为:void Bresenham_Line(int x_1, int y_1, int x_2, int y_2),其中 (x_1,y_1) 和 (x_2,y_2) 为两点的坐标。

2.4 角度数字微分算法

角度数字微分 DDA(digital differential analyzer)法的思想如下。

对于圆心为 (x_0,y_0)、半径为 R 的圆的参数方程为:

$$\begin{cases}x=x_0+R\cos\theta\\ y=y_0+R\sin\theta\end{cases}$$

将整个圆分成 N 等份,即把 2π 分成 N 等份,每一份 $\Delta\theta=2\pi/N$,则得到参数迭代公式为:

$$\begin{cases}x_i=x_0+R\cos\theta_i\\ y_i=y_0+R\sin\theta_i\end{cases}$$

式中:$\theta_i=i\times\Delta\theta$,$i$ 为迭代次数。最后对相邻的 (x_i,y_i)、(x_{i+1},y_{i+1}) 两点,调用绘制直线的算法画出直线,即可近似绘制出圆[3]。

2.5 绘制圆的 C51 函数

利用 DDA 算法,可以很容易地编写出绘制圆的 C51 函数,函数原型为:void DDA_Circle(int x_0, int y_0, int r),

其中(x_0,y_0)为圆心坐标,r 为半径。

2.6　其他几何图形的 C51 函数

其他规则曲线的生成步骤如下。首先在平面上确定一些满足条件的、位于曲线上的坐标点,然后用光滑直线分段连接相邻的点,就形成了曲线。如绘制$y=\sin(x)$函数的图形,坐标点选取好后,相邻点之间调用绘制直线的函数就可以近似地绘制出其图形。具体生成算法的描述可阅读参考文献[2]。

3　C51 函数库

形成函数库的优点是增加代码的重用性,对基于 T6963C 的 LCD 显示模块都可以使用此函数库,从而提高研发产品效率。当需要调用函数库中的函数时,只要给出函数的原型,把此函数库连接到应用程序中即可。

3.1　生成 C51 函数库

在 Keil C51 集成开发环境下生成 C51 函数库非常简单。将所有函数放在一个源文件中,并把此文件添加到工程,然后选中"Project"菜单中的"Option for Target'xxx'"命令,打开"Option for Target'xxx'"对话框,选"Output"标签,选中"Create Library"单选按钮,点击"确定"按钮后,重新执行"Build target"命令,即生成 LIB 库文件,如图 5 所示。

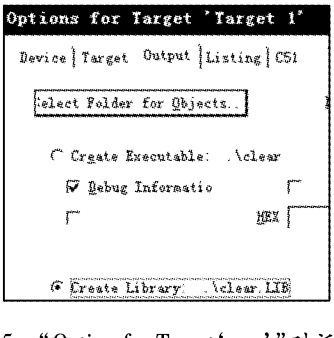

图 5　"Option for Target'xxx'"对话框

Fig. 5　Dialog box of"Option for Target 'xxx'"

3.2　使用 C51 函数库

C51 函数库生成后就可在工程项目中使用。为了便于使用,伴随该函数库创建了一个头文件,并在头文件中声明函数原型。当工程中使用 C51 函数库中的函数时,只要在源文件中包含此头文件并连接函数库即可。在 Keil C51 集成开发环境下,要将生成的函数库连接到工程中,需要运行"Project"菜单中的"Components, Environments, Book"命令,打开"Components, Environments and Books"对话框,选择"Add Files"按钮,把函数库添加到工程中。

4　结束语

基于 T6963C 的 LCD 模块被广泛应用于智能仪器仪表中。通过对该芯片的详细介绍,结合计算机图形学的知识,本文给出了在这类显示模块中绘制几何图形的 C51 函数,最后生成了 C51 函数库,便于在以后的产品开发中使用,从而提高产品的研发效率。C51 函数库在产品开发中得到了很好的应用,实践应用证明了它的高效性。

参考文献

[1] 孙俊喜. LCD 驱动电路、驱动程序设计及典型应用[M]. 北京:人民邮电出版社,2009:61 - 69.

[2] 王汝传,邹北骥. 计算机图形学[M]. 北京:人民邮电出版社,2003:93 - 103.

[3] 王云平. 液晶显示器绘图编程算法的实现[J]. 辽宁化工,2003,32(1):41 - 43.

[4] 李银华,姬光锋. T6963C 点阵式液晶显示模块的应用研究与编程[J]. 液晶与显示,2008,23(5):560 - 566.

[5] 袁满. 基于 T6963C 的 LCD 液晶显示的实现[J]. 自动化技术与应用,2007,26(9):110 - 111.

[6] 宋俊杰,原冬梅,金海龙,等. 基于 MSP430 的内置 T6963C 液晶显示模块控制技术[J]. 液晶与显示,2010,25(1):110 - 113.

[7] 李荣标,徐向前,李国明,等. 浅析 T6963C 液晶显示模块与 PIC 单片机的接口技术[J]. 现代显示,2009,20(3):34 - 37.

(上接第 73 页)

[3] Bernstein J. An overview of MEMS inertial sensing technology[EB/OL]. [2010 - 11 - 30]. http://www.sensorsmag.com/sensors/acceleration-vibration/an-overview-mems-inertial-sensing-technology-970.

[4] 杨金显,袁赣南. 基于 MIMU/GPS 的组合导航设计及实验[J]. 光学精密工程,2008,16(2):285 - 294.

[5] 吴盘龙,杜国平,薄煜明. 空地制导弹药的 MIMU/GPS 组合导航系统研究[J]. 电光与控制,2008,15(10):65 - 68.

[6] 张峥. 惯性步态测量系统的研究[D]. 武汉:武汉理工大学,2006.

[7] 杨金显,袁赣南,徐良臣,等. 基于微机械陀螺的 MIMU 关键技术[J]. 中国惯性技术学报, 2007,15(1):28 - 30.

[8] 朱欣华,姚天忠,邹丽新. 智能仪器原理与设计[M]. 北京:中国计量出版社,2002:45.

[9] 刘育浩,黄新生,谭红利. 基于高性能 DSP 和高精度 A/D 的导航计算机系统[J]. 中国惯性技术学报,2008,16(2):140 - 143.

[10] 朱飞,杨平. AVR 单片机 C 语言开发入门与典型实例[M]. 北京:人民邮电出版社,2009:22 - 31.