TFT3224b-3.5-64k (Ver20)

液晶显示器规格书

- 介绍
- 系统结构框图
- 引脚定义
- 总线时序
- 坐标与像素映射关系
- 寄存器描述
- 显示数据读写方式
- 接口电路
- 软件编写
- 机械尺寸与布局
- 提高功能

中大显示科技
www.ViewTech.cn

1 介绍

TFT3224b-3.5-64k 是专门针对单片机用户而设计的液晶显示器(带触摸屏)，采用 3.5 英寸、分辨率为 320x240 的真彩 TFT 屏，提供一个简单的高速 16 位总线与单片机连接，支持 64k 色。可以直接与 MCS51、MCS96、MC68、ARM 以及 DSP 相连。直接输入 X、Y 坐标，无须计算地址。

低功耗

轻薄设计（高度 8.5mm）

宽温（-20 度到 70 度）

亮度可调节（软件调节 8 种亮度）

低功耗模式（软件关断显示）

适合各种仪器仪表、工业设备的应用，其低功耗、轻薄设计亦能满足单节锂电池供电设备的需求。

软件性能全面升级，提供以下功能：

1、快速清屏功能；只需发送一条指令，控制板在 16.6 毫秒内以指定的颜色对整个画面进行清屏，清屏过程无须单片机的干预，极大地提高了开机和单一背景色的显示速度。

2、提供灵活的地址自动加一功能；地址自动加一的方向可以任意设置为 X 方向或 Y 方向。地址沿 X 方向自动加一时，遇到行尾将自动跳到下一行的行首。地址沿 Y 方向自动加一时，遇到列尾将自动跳到下一列的列首。

通过以上各种加强的功能，使得普通的单片机驱动彩色屏，也可以得到非常流畅的显示效果。

2 CPU 侧引脚定义

插座 J3：1mm 间距的 30 脚 FFC 插座

引脚	符号	功能
1	GND	
2	GND	
3	VCC	TFT3224b-3.5-64K-A +3.3V
		TFT3224b-3.5-64K-B +5V
4	/RD	读操作信号，低电平有效。
5	/WR	写操作信号，低电平有效。
6	/CS	片选信号，低电平有效
7	A0	地址
8	A1	地址
9	DATA0	数据总线
10	DATA1	数据总线
11	DATA2	数据总线
12	DATA3	数据总线
13	DATA4	数据总线
14	DATA5	数据总线
15	DATA6	数据总线
16	DATA7	数据总线
17	XR	触摸屏-右
18	YD	触摸屏-下
19	XL	触摸屏-左
20	YU	触摸屏-上
21	VCC	TFT3224b-3.5-64K-A +3.3V
22		TFT3224b-3.5-64K-B +5V
23	DATA8	数据总线
24	DATA9	数据总线
25	DATA10	数据总线
26	DATA11	数据总线
27	DATA12	数据总线
28	DATA13	数据总线
29	DATA14	数据总线
30	DATA15	数据总线

插座 J1：0.5mm 间距的 32 脚 FFC 插座

引脚	符号	功能
1	GND	
2	GND	
3	VCC	TFT3224b-3.5-64K-A +3.3V
		TFT3224b-3.5-64K-B +5V
4	/RD	读操作信号，低电平有效。
5	/WR	写操作信号，低电平有效。
6	/CS	片选信号，低电平有效
7	A0	地址
8	A1	地址
9	DATA0	数据总线
10	DATA1	数据总线
11	DATA2	数据总线
12	DATA3	数据总线
13	DATA4	数据总线
14	DATA5	数据总线
15	DATA6	数据总线
16	DATA7	数据总线
17	XR	触摸屏-右
18	YD	触摸屏-下
19	XL	触摸屏-左
20	YU	触摸屏-上
21	VCC	TFT3224b-3.5-64K-A +3.3V
22		TFT3224b-3.5-64K-B +5V
23	DATA8	数据总线
24	DATA9	数据总线
25	DATA10	数据总线
26	DATA11	数据总线
27	DATA12	数据总线
28	DATA13	数据总线
29	DATA14	数据总线
30	DATA15	数据总线
31	GND	
32	GND	

注意：

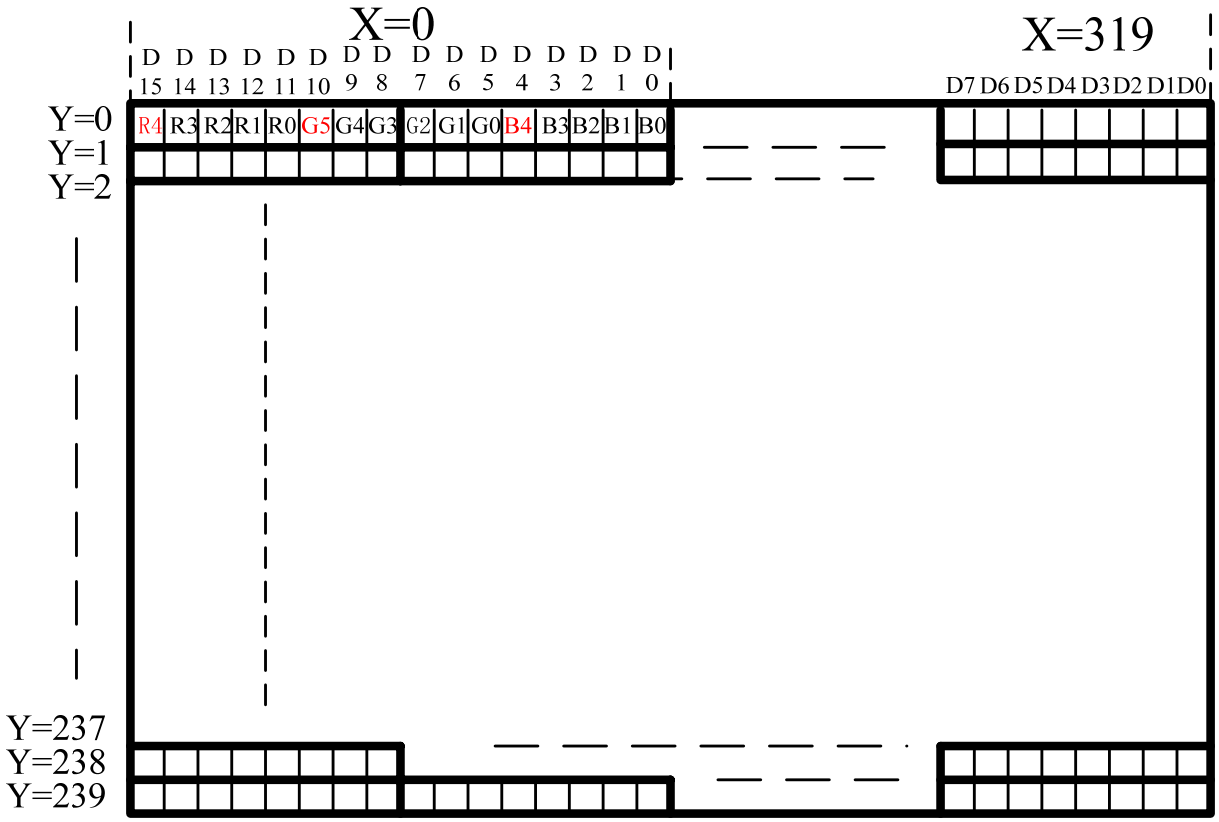
1. 控制板分 TFT3224b-3.5-64K-A 和 TFT3224b-3.5-64K-B 两种型号，主要差异是 VCC 的供电电压。未做特殊说明下，提供的都是 B 型，5V 供电。
2. 不论 A 型还是 B 型，所有接口信号（/CS、/WR、/RD、A[1：0]、D[7：]）都是兼容 3.3V 和 5V 逻辑电平的，可以与 5V 系统或 3.3V 系统直接相连接，不需要任何电平转换电路。

3 坐标与像素映射关系（64K 色）

列坐标（X）是以像素为单位的，每像素包含 16 位；因此，列坐标 X 取值范围是 0-319。

行坐标（Y）取值范围是 0-239。

像素格式为 R5-G6-B5，与数据总线 D[15：0]的对应关系如下图所示。



字节数据		D[15：11]	D[10：5]	D[4：0]
	颜色灰度	R[4：0]	G[5：0]	B[4：0]
基本颜色	最黑	00000	000000	00000
	亮蓝	00000	000000	11111
	亮绿	00000	111111	00000
	亮青	00000	111111	11111
	亮红	11111	000000	00000
	亮紫	11111	000000	11111
	亮黄	11111	111111	00000
	亮白	11111	111111	11111
蓝色灰度	最黑	00000	000000	00000
	较暗	00000	000000	00001
	…	…	…	…
	较亮	00000	000000	11110
	最亮	00000	000000	11111

绿色灰度	最黑	00000	000000	00000
	较暗	00000	000001	00000
	…	…	…	…
	较亮	00000	111110	00000
	最亮	00000	111111	00000
红色灰度	最黑	00000	000000	00000
	较暗	00001	000000	00000
	…	…	…	…
	较亮	11110	000000	00000
	最亮	11111	000000	00000

4 寄存器描述（基本功能-单点写）

共有 4 个寄存器，分别为列地址、行地址、状态控制寄存器、显示数据。

/CS	A1A0	/WR	功能
0	00	0	列地址寄存器 X
0	01	0	行地址寄存器 Y
0	10	0	控制寄存器 CMD
0	11	0	数据寄存器 DAT

列地址寄存器（X）：由于列地址取值范围是从 0-319，占 9bit。

-	-	-	-	-	-	-	X8	X7	X6	X5	X4	X3	X2	X1	X0

行地址寄存器（Y）：与列地址寄存器（X）相似，由于行地址取值范围是从 0-239，占 8bit。

-	-	-	-	-	-	-	-	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0

控制寄存器:（高 8 位没有使用）。在实现基本功能（单点写）时，不需要使用控制寄存器，直接将该寄存器写 0 就可以了。

0	0	0	0	0	0	0	0

数据寄存器 DAT：显示数据通过该寄存器写入和读出，每次读写操作后地址自动沿 X 方向加一。一次读写一个像素。

15	14	13	12	11	10	9	8	D7	D6	D5	D4	D3	D2	D1	D0
R[4:0]					G[5:0]						B[4:0]				

5 显示数据读写方式

首先必须指定行地址 Y，以及列地址 X。然后就可以将该行从地址 X 开始的数据连续进行读写操作，无须重新设置 X 和 Y。

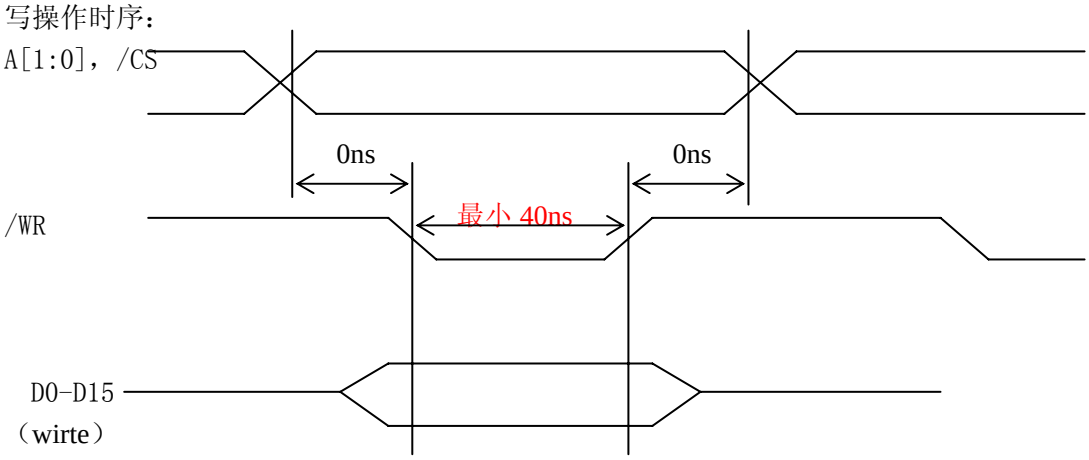
在显示数据的每次读写操作后，列地址 X 都将自动加 1。当地址加到行尾时，地址将跳

到下一行的行首。

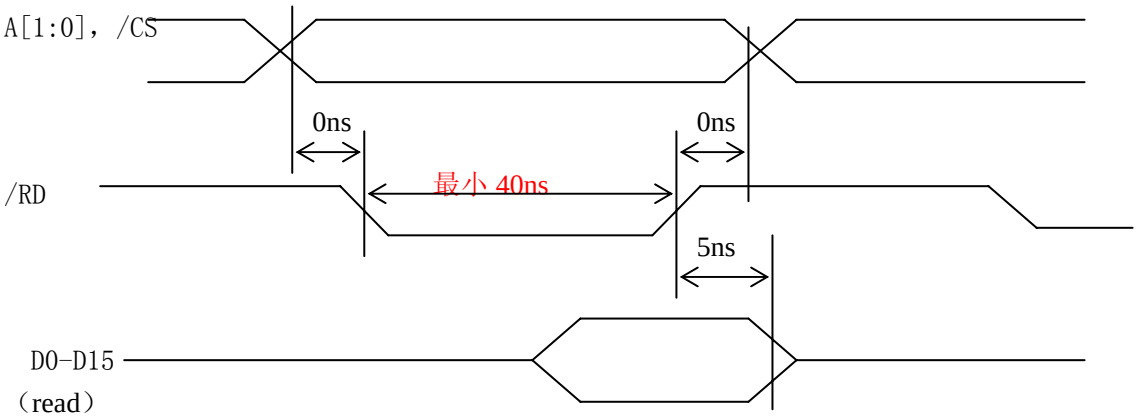
当要读写一个新的行时，必须重新设置 X、Y。

6 总线时序

写操作时序：

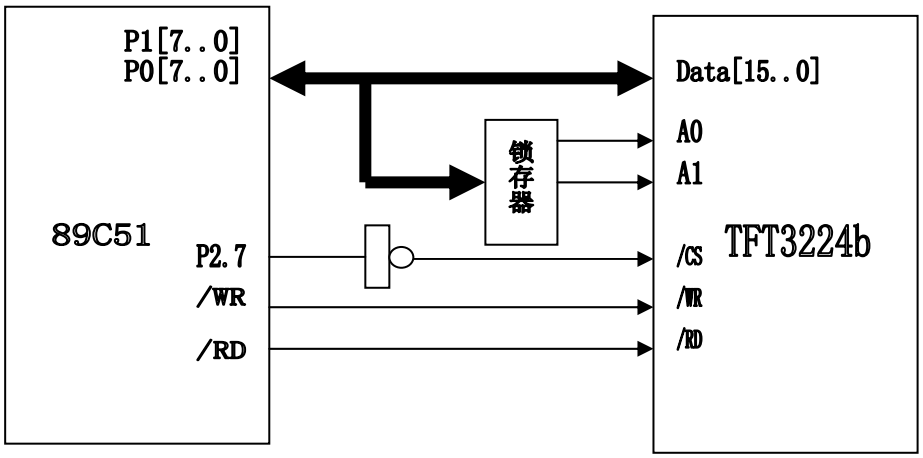


读操作时序：

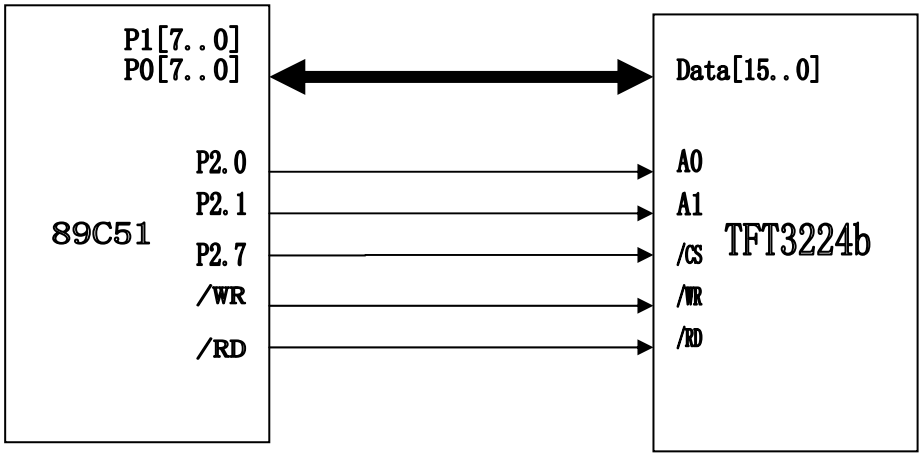


7 接口电路（以 MCS51 单片机为例）

典型接口电路：



DEMO 板接口电路：（省去了地址锁存器和译码器）

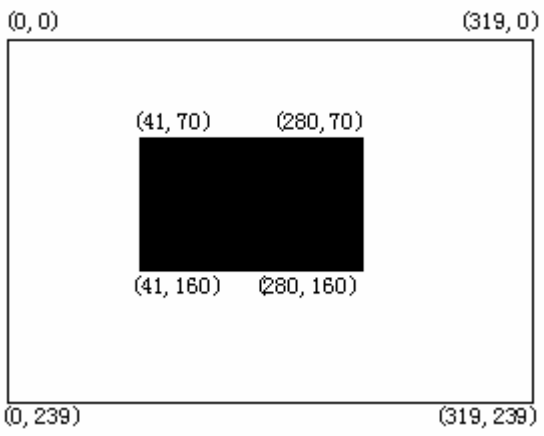


以上两种接口电路的端口地址分别为：

寄存器名	端口地址（典型）	端口地址（DEMO）
列地址寄存器	8000H	0000H
行地址寄存器	8001H	0100H
控制寄存器	8002H	0200H
读写显示数据	8003H	0300H

8 软件编写

图片数据的显示：



将图中间区域填充成蓝色。
采用行操作模式。
在每行的写操作前，先设置 X、Y。

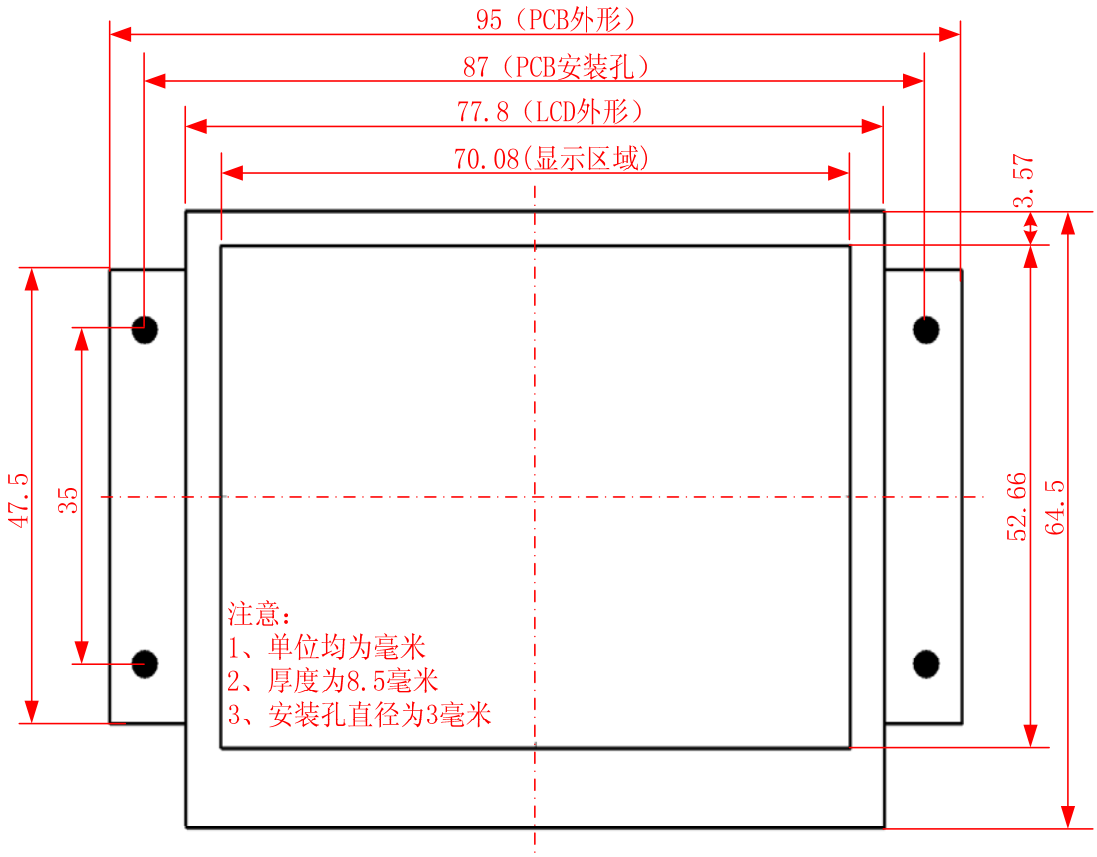
```
#include <reg51.h>
#include <absacc.h>
#define X_ADDR XBYTE[0x0000]
#define Y_ADDR XBYTE[0x0100]
#define CMD XBYTE[0x0200]
#define DAT XBYTE[0x0300]

main()
{
    unsigned int x,y;
    //clear panel
    CMD=0;
    P1=0;  X_ADDR = 0;
    P1=0;  Y_ADDR = 0;
    for (y=0;y<240;y++)
        {
          for (x=0;x<320;x++)    {P1=0xff; DAT = 0xff;}
           }

    //fill pattern
    CMD=0;
    for (y=70;y<161;y++)
     {    x=41;
          P1= x/256; X_ADDR = x;
          P1= y/256;Y_ADDR = y;
```

- for(;x<281;x++) {P1=0x00; DAT = 0x1f;}
- }

9 机械尺寸与布局



10 提高功能

真彩色 LCD 控制板性能全面升级，提供以下提高功能：

1、快速清屏功能；只需发送一条指令，控制板在 16.6 毫秒内以指定的颜色对整个画面进行清屏，清屏过程无须单片机的干预，极大地提高了开机和单一背景色的显示速度。

2、提供灵活的地址自动加一功能；地址自动加一的方向可以任意设置为 X 方向或 Y 方向。地址沿 X 方向自动加一时，遇到行尾将自动跳到下一行的行首。地址沿 Y 方向自动加一时，遇到列尾将自动跳到下一列的列首。

通过以上各种加强的功能，使得普通的单片机驱彩色屏，也可以得到非常流畅的显示效果。

提高功能是通过控制寄存器(A1A0=10)和数据寄存器(A1A0=11)的复用来实现的。

控制寄存器：(只使用低 7 位)

-	-	Power_Off	Inc_dir	Clear_en	Mode[1]	Mode[0]	-

Power_Off：控制寄存器 bit[5]，初始值为 0；

低功耗模式控制位。为 0 正常工作，功耗见第 4 节的描述。为 1 进入低功耗模式，关断显示并保存画面，当重新回到正常工作模式下时，画面保持不变。低功耗模式的功耗见第 2 节的描述。

Inc_dir：控制寄存器 bit[4]，初始值为 0；

设定地址自动加一的方向，为 0 沿 X 方向自动加一，为 1 沿 Y 方向自动加一。地址沿 X 方向自动加一时，遇到行尾将自动跳到下一行的行首。地址沿 Y 方向自动加一时，遇到列尾将自动跳到下一列的列首。

Clear_en：控制寄存器 bit[3]，初始值为 0；

清屏使能位。该位为 1 时，启动清屏操作，控制板将自动按照定义的背景色颜色（见 bit[2]）填充整个画面，该过程需要耗时 16.6 毫秒。在填充过程中，无须单片机的干预。单片机使能该位后，等待 16.6 毫秒，再将该为写为 0，重新回到正常模式工作。可见，在进行清屏操作前，必须先设置背景色颜色。

Mode[1：0]：控制寄存器 bit[2：1]，初始值为 00；

数据寄存器 DAT（A1A0==11）的功能定义。

Mode = 00 ： 数据寄存器 DAT 是像素数据写入寄存器。

一次写入 1 个像素，数据格式是 R5-G6-B5（参见第 4 节描述）；

Mode = 01 ： 未定义；

Mode = 10 ： 数据寄存器 DAT 是背景色颜色写入寄存器。

背景色颜色用于清屏和 16 点写入模式。数据格式是 R5-G6-B5。

Mode = 11 ： 数据寄存器 DAT 是亮度值写入寄存器。

最低 3 位为有效位 Brightness[2:0]，定义 8 种亮度。第 7 级亮度最强，第 0 级亮度最低。在不同的亮度下，VCC 的电流是不同的，详细情况见第 3 节。

-	-	-	-	-	Bright[2]	Bright[1]	Bright[0]

提高功能的例程：

```
#define X_ADDR XBYTE[0x0000]
#define Y_ADDR XBYTE[0x0100]
#define CMD XBYTE[0x0200]
#define DAT XBYTE[0x0300]

unsigned char code zk[32] = {//请
0x00, 0x47, 0x20, 0x23, 0x00, 0xEF, 0x20, 0x23, //left
0x22, 0x23, 0x22, 0x23, 0x2A, 0x32, 0x22, 0x02,
0x48, 0xFC, 0x40, 0xF8, 0x40, 0xFE, 0x08, 0xFC, //right
0x08, 0xF8, 0x08, 0xF8, 0x08, 0x08, 0x28, 0x10};

unsigned char code picture[];
main()
{
unsigned int x,y;
unsigned int i;
unsigned char j,k,z,m,n;

//////////////////////以下是测试基本功能/////////////////////////////////////
CMD=0x00;
for (y=0;y<480;y++)
  {   x=0;
      P1 = x/256;X_ADDR = x;
      P1 = y/256;Y_ADDR = y;
```

```
        for (;x<640;x++)    {P1=0; DAT = 0x1f;}
      }

////////////////////////以下是清屏功能//////////////////////////////////////
//用蓝色清屏
CMD = 0x04;
P1=0x00; DAT = 0x1f;//背景色
CMD = 0x08;//启动填充操作
for(y=0;y<250*10;y++);//延时 16.6 毫秒
CMD = 0x00;//退出填充操作

for(y=0;y<1;y++){x=1;while(x!=0)x++;}
for(y=0;y<1;y++){x=1;while(x!=0)x++;}
for(y=0;y<1;y++){x=1;while(x!=0)x++;}

//用绿色清屏
CMD = 0x04;
P1=0x07; DAT = 0xe0;//背景色
CMD = 0x08;//启动填充操作
for(y=0;y<250*10;y++);//延时 16.6 毫秒
CMD = 0x00;//退出填充操作

for(y=0;y<1;y++){x=1;while(x!=0)x++;}
for(y=0;y<1;y++){x=1;while(x!=0)x++;}
for(y=0;y<1;y++){x=1;while(x!=0)x++;}

//用红色清屏
CMD = 0x04;
P1=0xf8; DAT = 0x00;//背景色
CMD = 0x08;//启动填充操作
for(y=0;y<250*10;y++);//延时 16.6 毫秒
CMD = 0x00;//退出填充操作
////////////////////////以上是清屏功能//////////////////////////////////////

for(y=0;y<1;y++){x=1;while(x!=0)x++;}
for(y=0;y<1;y++){x=1;while(x!=0)x++;}
for(y=0;y<1;y++){x=1;while(x!=0)x++;}

//写单色图片
for (y=0; y<80; y++)
        {
        P1=0;X_ADDR = 0;
        Y_ADDR = y;
        for ( n=0; n<10; n++)
```

```
                {j= picture[y*10+n];
                 for ( m=0; m<8; m++)
                       {
                        if(j&0x80){P1 = 0x00;DAT = 0x00; }
                        else {P1 = 0xff;DAT = 0xff; }
                        j=(j<<1);
                       }
                }
            }

for(y=0;y<1;y++){x=1;while(x!=0)x++;}
for(y=0;y<1;y++){x=1;while(x!=0)x++;}

//写汉字
for ( n=0; n<=15; n++ )
{
        P1 = 0; X_ADDR = 0;
        Y_ADDR = n;
        j    = zk[n];
                     for ( m=0; m<8; m++)
                       {
                        if(j&0x80){P1 = 0x00;DAT = 0x00; }
                        else {P1 = 0xff;DAT = 0xff; }
                        j=(j<<1);
                       }
        j = zk[n+16];
             for ( m=0; m<8; m++)
                       {
                        if(j&0x80){P1 = 0x00;DAT = 0x00; }
                        else {P1 = 0xff;DAT = 0xff; }
                        j=(j<<1);
                       }



}
unsigned char code picture[800]={//zhongda，80x80
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
```

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0xC0, 0x00, 0x60, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x01, 0x80, 0x00, 0xC0, 0x03, 0xFF, 0x03, 0xFF, 0xC0,
0x00, 0x01, 0x80, 0x00, 0xC0, 0x06, 0x06, 0x00, 0x00, 0x00,
0x00, 0x7F, 0xFC, 0x01, 0x80, 0x06, 0x06, 0x00, 0x00, 0x00,
0x00, 0x63, 0x0C, 0x7F, 0xFF, 0x07, 0xFC, 0x00, 0x00, 0x00,
0x00, 0x63, 0x18, 0x01, 0x80, 0x0C, 0x0C, 0x3F, 0xFF, 0xC0,
0x00, 0xC6, 0x18, 0x03, 0x80, 0x0F, 0xFC, 0x00, 0x60, 0x00,
0x00, 0xC6, 0x30, 0x03, 0x80, 0x03, 0x60, 0x07, 0xD8, 0x00,
0x00, 0xFF, 0xF0, 0x07, 0x80, 0x36, 0xCC, 0x06, 0xD8, 0x00,
0x00, 0x0C, 0x00, 0x0D, 0x80, 0x36, 0xDC, 0x0C, 0xCC, 0x00,
0x00, 0x0C, 0x00, 0x0C, 0xC0, 0x1E, 0xF0, 0x39, 0x86, 0x00,
0x00, 0x0C, 0x00, 0x38, 0xE0, 0x1D, 0xE0, 0x61, 0x86, 0x00,
0x00, 0x18, 0x00, 0x60, 0x70, 0x0D, 0xC0, 0xC1, 0x80, 0x00,
0x00, 0x18, 0x01, 0xC0, 0x31, 0xFF, 0xFC, 0x07, 0x00, 0x00,
0x00, 0x00, 0x07, 0x00, 0x00, 0x00, 0x00, 0x06, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x03, 0xCF, 0xFE, 0x38, 0x20, 0x47, 0xB8, 0x00, 0x00,
0x00, 0x01, 0x87, 0x32, 0x60, 0x60, 0xCC, 0xD8, 0x00, 0x00,
0x00, 0x00, 0xC4, 0x30, 0xC0, 0xE1, 0xCC, 0xD8, 0x00, 0x00,
0x00, 0x00, 0xC4, 0x31, 0xF0, 0xE1, 0xCE, 0x9E, 0x00, 0x00,

```
0x00, 0x00, 0x68, 0x31, 0x99, 0x62, 0xC7, 0x9B, 0x00, 0x00,
0x00, 0x00, 0x68, 0x31, 0x9A, 0x64, 0xCD, 0xDB, 0x00, 0x00,
0x00, 0x00, 0x30, 0x31, 0x9B, 0xF7, 0xEC, 0xDB, 0x00, 0x00,
0x00, 0x00, 0x30, 0x31, 0x98, 0x60, 0xCC, 0xDB, 0x00, 0x00,
0x00, 0x00, 0x30, 0x78, 0xF0, 0x60, 0xC7, 0x96, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
 };
```

以上原程序可到www.viewtech.cn上下载。