

COG12864C102

使用说明书

目 录

序号	内 容 标 题	页码
1	概述	2
2	特点	2
3	外形及接口引脚功能	3~4
4	基本原理	4~5
5	技术参数	5~6
6	时序特性	6~9
7	指令功能及硬件接口与编程案例	9~23



1. 概述

12864C102可以显示128列*64行点阵单色图片，或显示8个/行*4行16*16点阵的汉字，或显示16个/行*8行8*8点阵的英文、数字、符号。

2. 12864C102液晶模块的特性

2.1 结构牢：背光带有挡墙，焊接式 FPC。

2.2 IC 采用矽创公司 ST7565R, 功能强大，稳定性好

2.3 功耗低:5 - 100mW（不带背光 5mW, 带背光不大于 100mW）；

2.4 显示内容：

- 128*64 点阵单色图片；

- 可選用 16*16 点阵或其他点阵的图片来自编汉字，按照 16*16 点阵汉字来计算可显示 8 字/行*4 行。按照 12*12 点阵汉字来计算可显示 10 字/行*4 行。

2.5 指令功能强:可组合成各种输入、显示、移位方式以满足不同的要求；

2.6 接口简单方便:可采用 4 线 SPI 串行接口，或选择并行接口。

2.7 工作温度宽:-10℃ - 60℃；

2.8 可靠性高:寿命为 50,000 小时(25℃)。



3. 外形尺寸及接口引脚功能

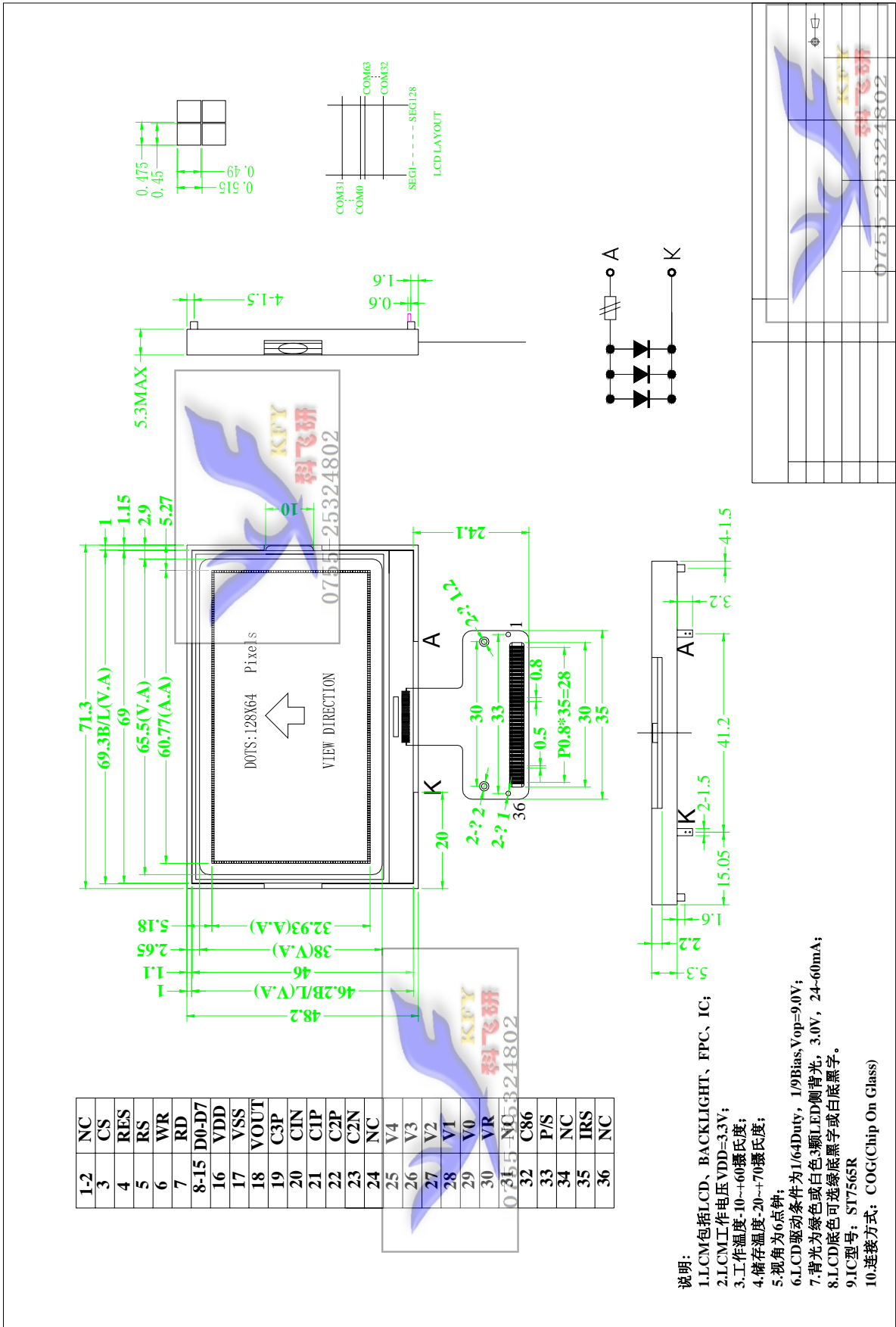


图 1. 外形尺寸

模块的接口引脚功能

引线号	符号	名称	功能
1	NC	空脚	空脚
2	NC	空脚	空脚
3	CS	片选	低电平片选
4	RES	复位	低电平复位, 复位完成后, 回到高电平, 液晶模块开始工作
5	RS	寄存器选择信号	H:数据寄存器 0:指令寄存器
6	R/W(//WR)	读/写, 或“写”	并行接口时并且选择 6800 时序时: H:读数据 L:写数据 并行接口时并且选择 8080 时序时: 写数据, 低电平有效. 串行接口时: 接 VDD
7	E(/RD)	使能信号, 或“读”	并行接口时并且选择 6800 时序时: 使能信号, 高电平有效. 并行接口时并且选择 8080 时序时: 读数据, 低电平有效. 串行接口时: 接 VDD
8-13	D0-D5	I/O	数据总线 DB0~DB5 串行接口时: 接 VDD
14	D6 (SDA)	I/O	并行接口时: 数据总线 DB6 串行接口时: 串行时钟 (SDA)
15	D7 (SCLK)	I/O	并行接口时: 数据总线 DB7 串行接口时: 串行数据 (SCLK)
16	VDD	供电电源正极	供电电源正极
17	VSS	接地	0V
18	VOUT	LCD 倍压输出	外接升压电容: 方法见“7. 指令功能及硬件接口与编程案例”
19	C3P	倍压电路	
20	C1N	倍压电路	
21	C1P	倍压电路	
22	C2P	倍压电路	
23	C2N	倍压电路	
24	NC	空脚	空脚
25	V4	偏置电压	LCD 驱动偏置电压。各与 VSS 之间接电容。方法见“7. 指令功能及硬件接口与编程案例”。 电压关系: $V_{OUT} > V_0 > V_1 > V_2 > V_3 > V_4 > V_{SS}$ 。
26	V3	偏置电压	
27	V2	偏置电压	
28	V1	偏置电压	
29	V0	偏置电压	
30	VR	产品电压调整	电压调整, 通过 V0 与 VSS 之间设置电阻来调整电压。方法见“7. 指令功能及硬件接口与编程案例”。
31	NC	空脚	空脚
32	C86	选择 6800 或 8080	并行接口时: H:6800 系统, L:8080 系统。 串行接口时: 接 VDD
33	P/S	选串并控制接口	H:并行接口, L:串行接口
34	NC		空脚
35	IRS	内/外电阻	L: 使用外部电阻, 此时“VR”引脚起作用。 H: 使用 IC 内部电阻进行调对比度。此时“VR”引脚不起作用
36	NC		空脚

表 1: 模块的接口引脚功能

4. 基本原理

4.1 液晶屏 (LCD)

在 LCD 上排列着 128×64 点阵, 128 个列信号与驱动 IC 相连, 64 个行信号也与驱动 IC 相连, IC 邦定在 LCD 玻璃上 (这种加工工艺叫 COG)。

4.2 工作电路:

图2是12864C102模块的电路框图, 它由驱动IC ST7565R及几个电阻电容组成。

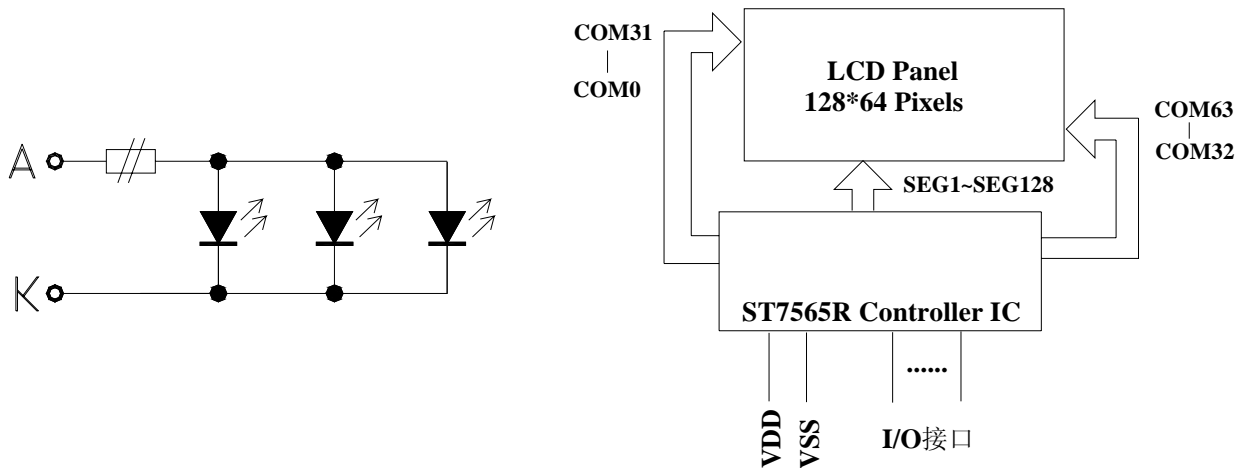


图2: 电路框图

4.2 背光参数

该型号液晶模块带 LED 背光源。它的性能参数如下:

背光板可选择绿色、白色。

正常工作电流为: $24 \sim 60\text{mA}$ (LED 灯数共 3 颗);

工作电压: 3.0V ;

正常工作条件下, LED 可连续点亮 5 万小时;

5. 技术参数

5.1 最大极限参数 (超过极限参数则会损坏液晶模块)

名称	符号	标准值			单位
		最小	典型	最大	
电路电源	VDD - VSS	-0.3		7.0	V
LCD 驱动电压	VDD - V0	VDD - 13.5		VDD + 0.3	V
静电电压		-	-	100	V
工作温度		-10		+60	°C
储存温度		-20		+70	°C

表 2: 最大极限参数

5.2 直流 (DC) 参数

名称	符号	测试条件	标准值	单位
----	----	------	-----	----



			MIN	TYPE	MAX	
工作电压	VDD		2.4	3.3	3.6	V
背光工作电压	VLED		2.9	3.0	3.1	V
输入高电平	VIH	-	2.2		VDD	V
输入低电平	VI0	-	-0.3		0.6	V
输出高电平	VOH	IOH = 0.2mA	2.4		-	V
输出低电平	VO0	I00 = 1.2mA	-		0.4	V
模块工作电流	IDD	VDD = 3.0V	-		1.0	mA
背光工作电流	ILED	VLED=3.0V	24	45	60	mA

表 3: 直流 (DC) 参数

6. 读写时序特性

6.1 串行接口:

从 CPU 写到 ST7565R (Writing Data from CPU to ST7565R)

The 4-line SPI Interface

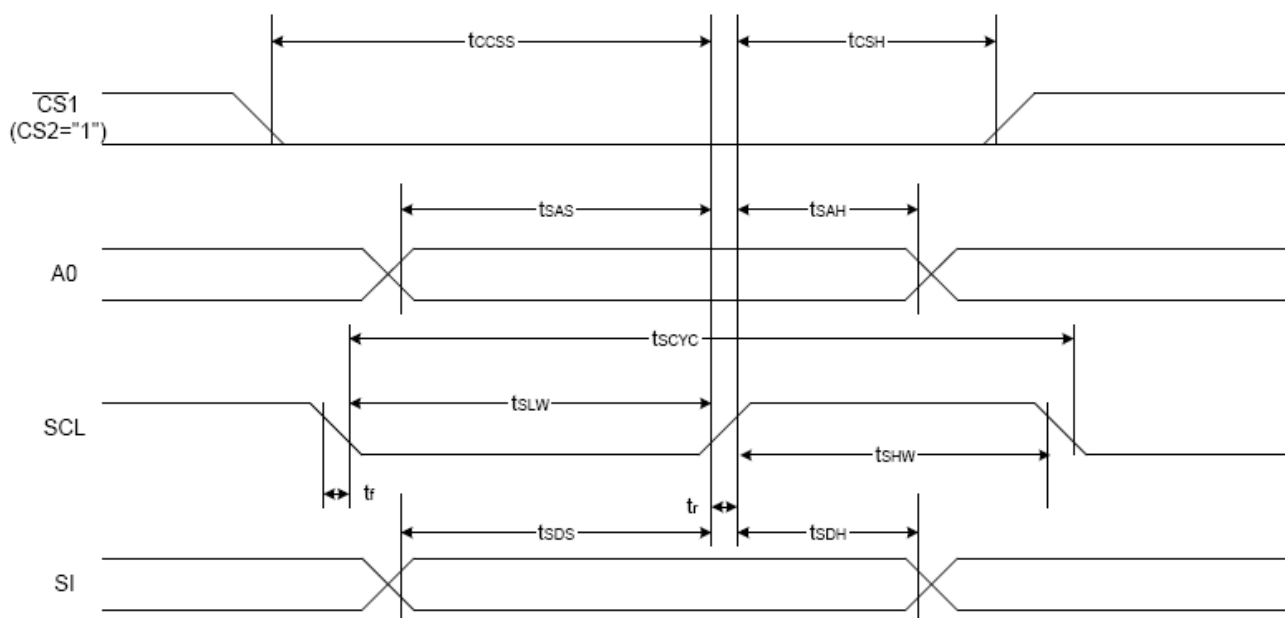


图 4. 从 CPU 写到 ST7565R (Writing Data from CPU to ST7565R)

6.2 串行接口: 时序要求 (AC 参数):

写数据到 ST7565R 的时序要求:

表 4.

项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
4线 SPI串口时钟周期 (4-line SPI Clock Period)	T _{scyc}	引脚: SCK	50	--	25	ns
保持SCK高电平脉宽 (SCK "H" pulse width)	T _{shw}	引脚: SCK	25			ns
保持SCK低电平脉宽 (SCK "L" pulse width)	T _{slw}	引脚: SCK	25			ns

地址建立时间 (Address setup time)	T_{SAS}	引脚: RS	20	--	--	ns
地址保持时间 (Address hold time)	T_{SAH}	引脚: RS	10	--	--	ns
数据建立时间 (Data setup time)	T_{SDS}	引脚: SI	20	--	--	ns
数据保持时间 (Data hold time)	T_{SDH}	引脚: SI	10	--	--	ns
片选信号建立时间 (CS-SCL time)	T_{CSS}	引脚: CS	20			ns
片选信号保持时间 (CS-SCL time)	T_{CSH}	引脚: CS	40			ns

$VDD = 3.0V \pm 5\%$, $T_a = 25^\circ C$

6.3 并行接口:

从 CPU 写到 ST7565R (Writing Data from CPU to ST7565R)

System Bus Read/Write Characteristics 1 (For the 8080 Series MPU)

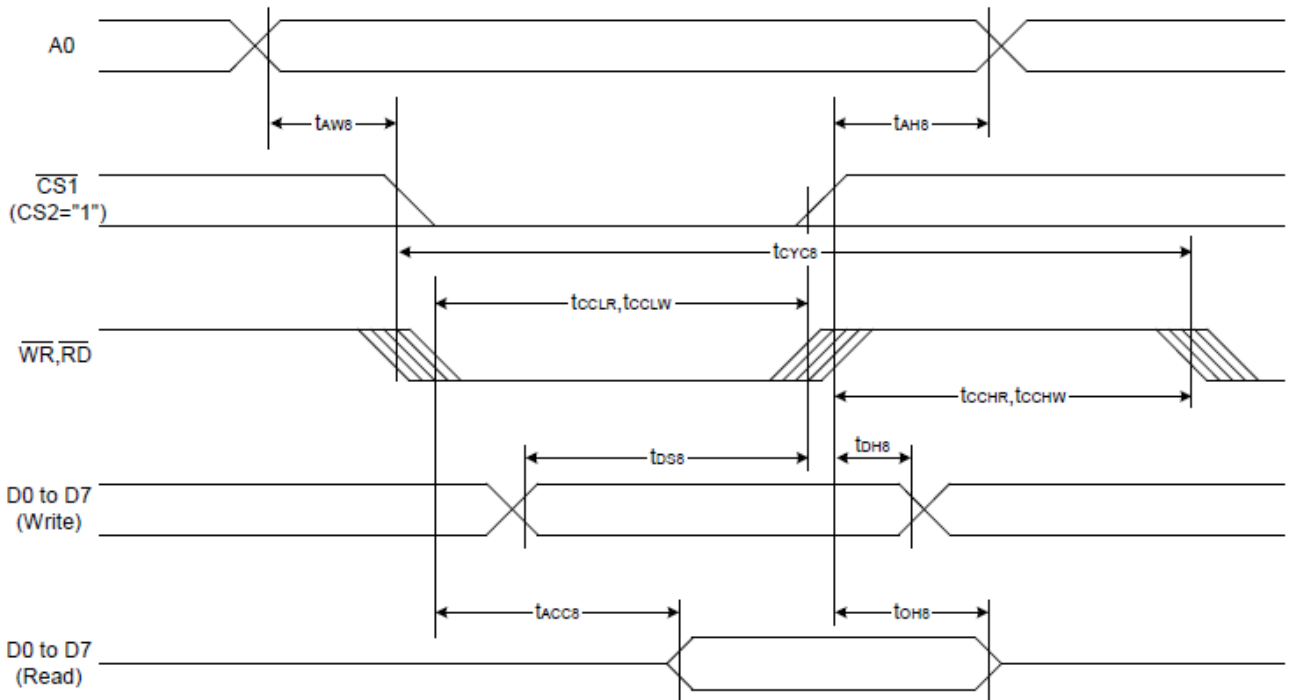


图 5. 从 CPU 写到 ST7565R (Writing Data from CPU to ST7565R)

System Bus Read/Write Characteristics 2 (For the 6800 Series MPU)

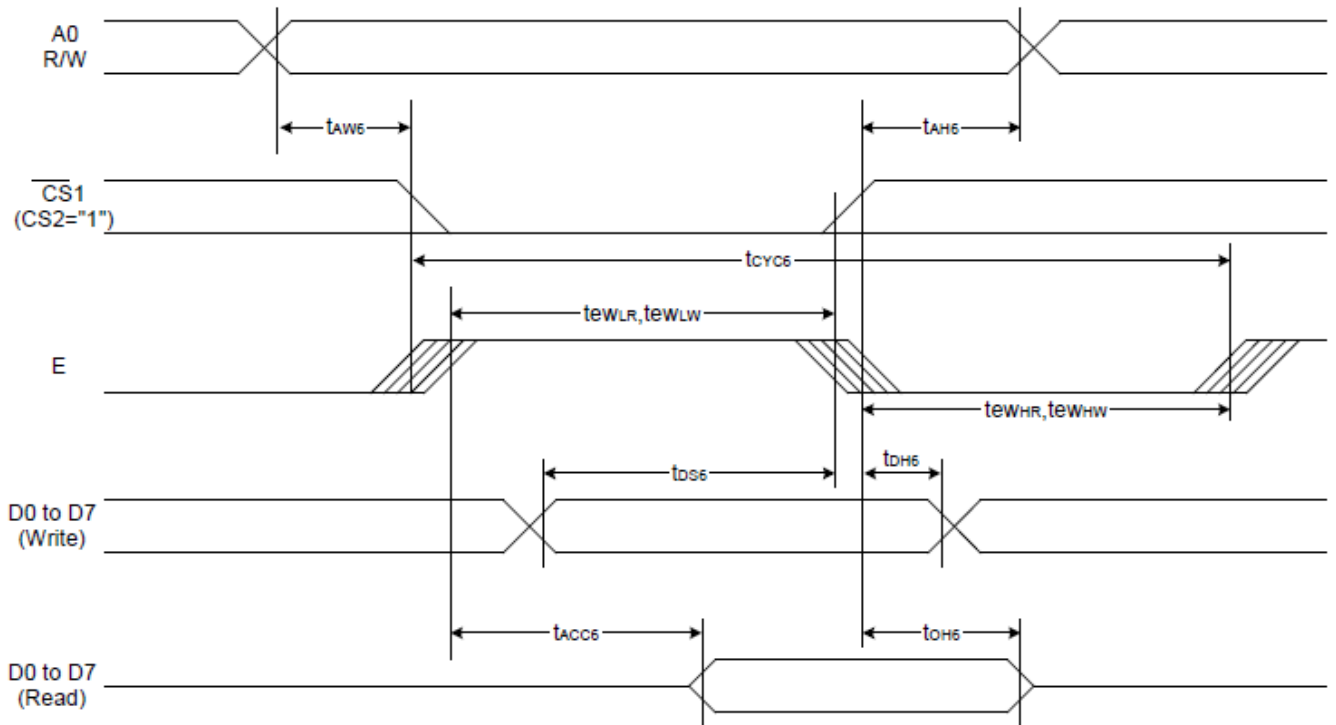


图 6. 从 CPU 写到 ST7565R (Writing Data from CPU to ST7565R)

6.4 并行接口：时序要求（AC 参数）：

写数据到 ST7565R 的时序要求：（8080 系列 MPU）

项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
地址保持时间	A0	tAH8	0	--	--	ns
地址建立时间		tAW8	0		--	ns
系统循环时间		tCYC8	240		--	ns
使能“低”脉冲（写）	WR	tCCLW	80	--	--	ns
使能“高”脉冲（写）		tCCHW	80	--	--	ns
使能“低”脉冲（读）	RD	tCCLR	140	--	--	ns
使能“高”脉冲（读）		tCCHR	80	--		ns
写数据建立时间	D0-D7	tDS8	40		--	ns
写数据保持时间		tDH8	0		--	
读时间		tACC8	--		70	
读输出允许时间		tOH8	5		50	ns



写数据到 ST7565R 的时序要求：（6800 系列 MPU）

项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
地址保持时间	A0	tAH6	0	--	--	ns
地址建立时间		tAW6	0		--	ns
系统循环时间		tCYC6	240		--	ns
使能“低”脉冲（写）	WR	tEWLW	80	--	--	ns
使能“高”脉冲（写）		tEWHW	80	--	--	ns
使能“低”脉冲（读）	RD	tEWLR	80	--	--	ns
使能“高”脉冲（读）		tEWHR	140	--		ns
写数据建立时间	D0-D7	tDS6	40		--	ns
写数据保持时间		tDH6	0		--	
读时间		tACC6	--		70	
读输出允许时间		tOH6	5		50	ns

6.5 电源启动后复位的时序要求（RESET CONDITION AFTER POWER UP）:

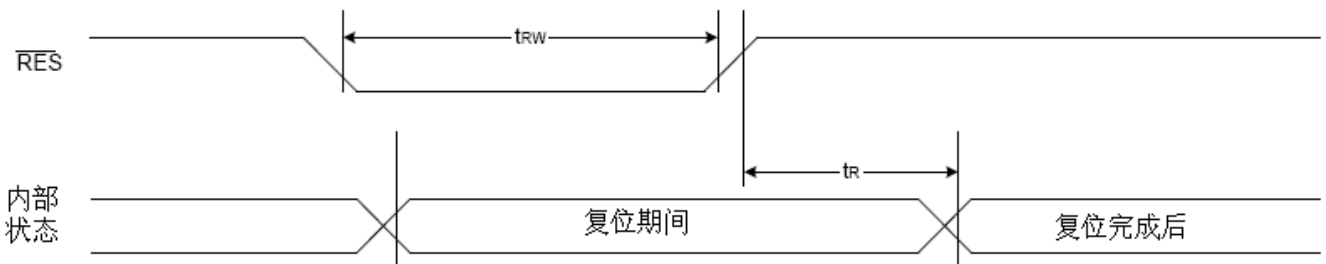


图 7：电源启动后复位的时序

表 6：电源启动后复位的时序要求

项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
复位时间	tr		--	--	1.0	us
复位保持低电平的时间	trw	引脚：RES	1.0	--	--	us

7. 指令功能:

7.1 指令表

格式:



共 11 种指令: 1. 清除, 2. 返回, 3. 输入方式设置, 4. 显示开关, 5. 控制, 移位, 6. 功能设置, 7. CGRAM 地址设置, 8. DDRAM 地址设置, 9. 读忙标志, 10. 写数据到 CG/DDRAM, 11. 读数据由 CG/DDRAM。

指令表

表 8.

指令名称	指令码									说明
	RS	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
(1)显示开/关 (display on/off)	0	1	0	1	0	1	1	1	0 1	显示开/关: 0:关, 1: 开
(2)显示初始行设置 (Display start line set)	0	0	1	显示初始行地址, 共 5 位						设置显示存储器的显示初始行
(3)页地址设置 (Page address set)	0	1	0	1	1	显示页地址, 共 4 位				设置显示页地址(注: 每 8 行为一个页, 64 行分为 8 个页, 例 0000 为第一页, 0001 为第二页)
(4) 列地址高4位 设置	0	0	0	0	1	列地址的高 4 位				高 4 位与低 4 位共同组成列地址, 分别指定 128 列中任对应列。本液晶模块的第一列的地址为 00000001, 所以此指令表达为: 0x10, 0x01
						列地址的低 4 位				
(5) 读状态 (Status read)	0	状态				0	0	0	0	在本型号液晶模块不用此指令
(6)写数据(Display data write)	1	8 位显示数据								从 CPU 写数据到液晶模块
(7)读数据(Display data read)	1	8 位显示数据								在本型号液晶模块不用此指令
(8) 显示列地址增 减 (ADC select)		1	0	1	0	0	0	0	0 1	显示列地址增减: 0: 常规: 从左到右, 1: 反转: 从右到左
(9)显示正显/反显 (Display normal/reverse)	0	1	0	1	0	0	1	1	0 1	显示正显/反显: 0:常规: 正显 1:反显
(10)显示全部点阵 (Display all points)	0	1	0	1	0	0	1	0	0 1	显示全部点阵: 0:常规 1:显示全部点阵
(11)LCD 偏压比设置 (LCD bias set)	0	1	0	1	0	0	0	1	0 1	设置偏压比: 0: 1/9 BIAS 1: 1/7BIAS
(12) Read-modify-write	0	1	1	1	0	0	0	0	0	Column address increment At write: +1 At read: 0
13) 退出上述指令 (End)	0	1	1	1	0	1	1	1	0	退出上述 “read/modify/write” 指令
(14) 软件复位 (Reset)	0	1	1	1	0	0	0	1	0	软件复位。
(15) 行扫描顺序选 择(Common		1	1	0	0	0	0	0	0 1	行扫描顺序选择: 0: 普通顺序

output mode select)										1: 反向扫描
(16) 电源控制 (Power control set)		0	0	1	0	1	电压操作模式选择, 共 3 位			选择内部电压供应操作模式
(17) 选择内部电阻比例	0	0	0	1	0	0	内部电压值电阻设置			选择内部电阻比例 (Rb/Ra), 本液晶模块通过外置电阻设置, 此指令失效
(18) 内部设置液晶电压模式 设置的电压值	0	1	0	0	0	0	0	0	1	设置内部电阻微调, 以设置液晶电压, 此两个指令需紧接着使用
		0	0	6 位电压值数据, 0~63 共 64 级						
(19) 静态图标显示: 开/关	0	1	0	1	0	1	1	0	0	0: 关, 1: 开。本液晶屏无此图标。此指令在进入及退出睡眠模式时起作用
(20) 升压倍数选择 (Booster ratio set)	0	1	1	1	1	1	0	0	0	选择升压倍数: 00: 2 倍, 3 倍, 4 倍 01: 5 倍 11: 6 倍。本模块外部已设置升压倍数为 4 倍, 不必使用此指令
		0	0	0	0	0	0	0	2 位数设置 升压倍数	
(21) 省电模式 (Power save)										省电模式, 此非一条指令, 是由“(10)显示全部点阵”、(19)静态图标显示: 开/关等指令合成一个“省电功能”。详细看 IC 规格书第 47 页“POWER SAVE”
(22) 空指令 (NOP)	0	1	1	1	0	0	0	1	1	空操作
(23) 测试 (Test)	0	1	1	1	1	*	*	*	*	内部测试用, 千万别用!

请详细参考 IC 资料“ST7564R_V15.PDF”的第 42~49 页。

7.3 点阵与 DD RAM 地址的对应关系

请留意页的定义: PAGE, 与平时所讲的“页”并不是一个意思, 在此表示 8 个行就是一个“页”, 一个 128*32 点阵的屏分为 8 个“页”, 从第 0“页”到第 7“页”。

DB7--DB0 的排列方向: 数据是从下向上排列的。最低位 D0 是在最上面, 最高位 D7 是在最下面。

下图摘自 ST7565R IC 资料, 可通过“ST7565R_V15.PDF”之第 27 页获取最佳效果。

7.4 初始化方法

用户所编的显示程序, 开始必须进行初始化, 否则模块无法正常显示, 过程请参考程序

7.5 程序举例:

液晶模块与 MPU(以 8051 系列单片机为例)接口图如下:

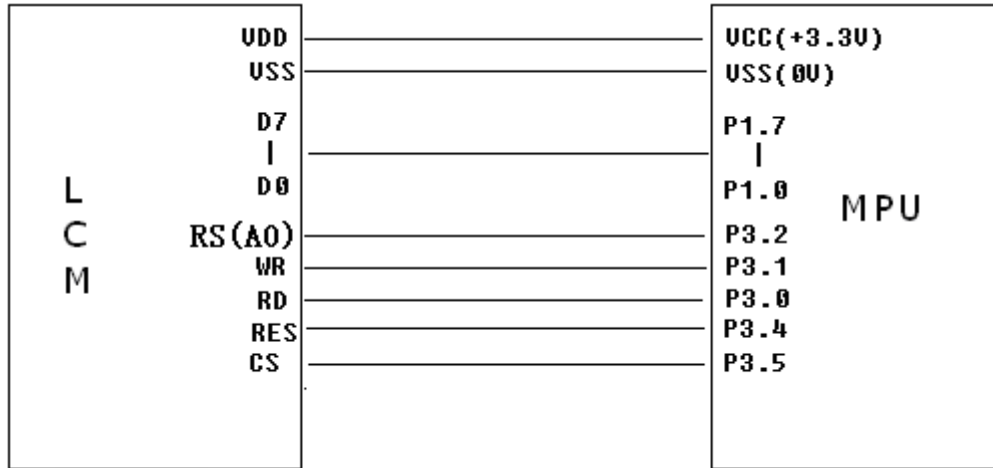
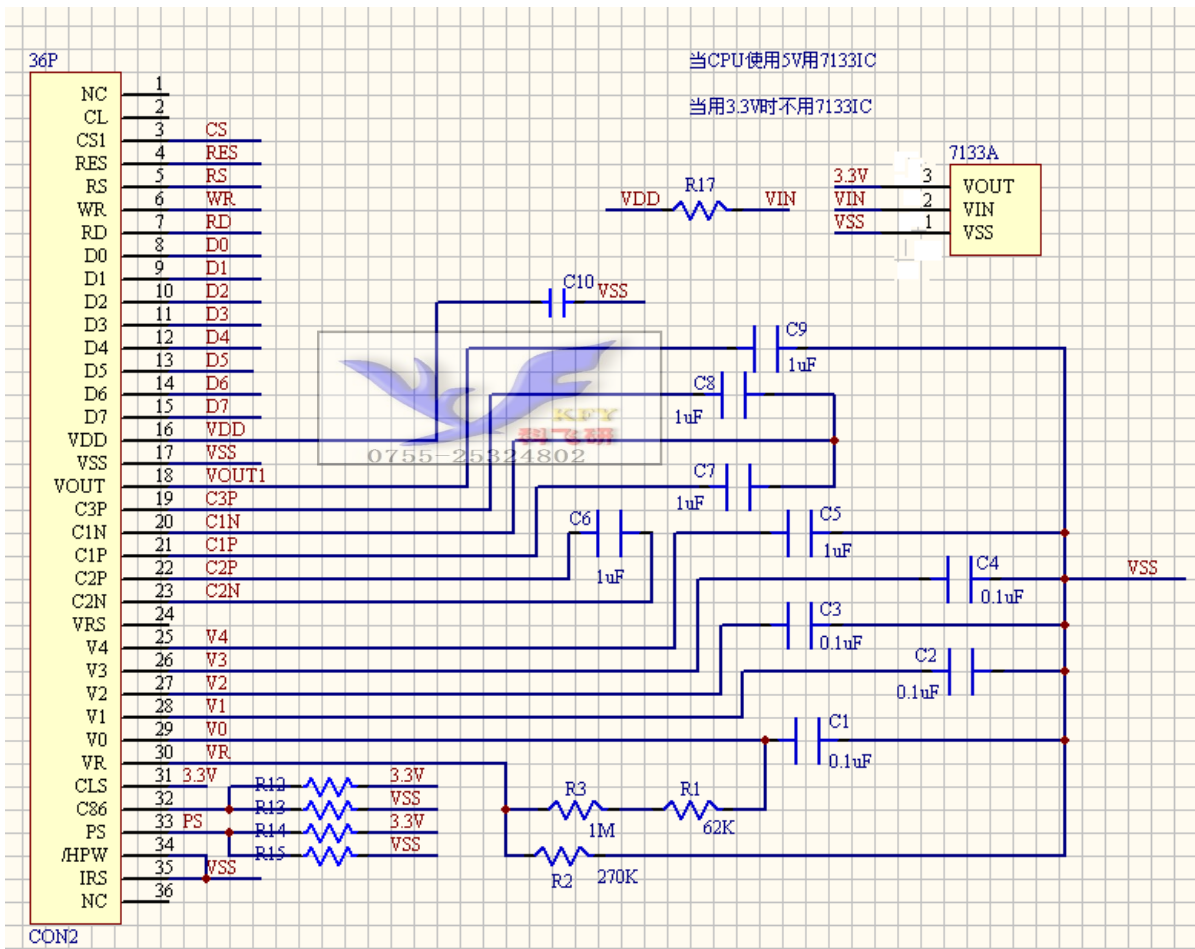


图 8. 并行接口



所有电容请使用耐压25V的无极性电容, 电容质量要好。

```

/* Test program for COG12864
   驱动 IC 是:ST7565R(or competible)

*/
#include <reg51.H>

sbit cs1=P3^5; /*接口定义*/
sbit reset=P3^4; /*接口定义*/
sbit rs=P3^2; /*接口定义*/
sbit rd=P3^0; /*接口定义*/
sbit wr=P3^1; /*接口定义。另外 P1.0~1.7 对应 DB0~DB7*/

void Transfer_data(int data1);
void Transfer_command(int data1);
unsigned char code graphic[];
unsigned char code graphic1[];
unsigned char code graphic2[];

unsigned char code graphic3[];
unsigned char code graphic4[];
void Delay(int i);
void Delay1(int i);
void DisplayGraphic();
void DisplayGraphic1();
void DisplayGraphic2();
void Initial_Lcd();
void clear_screen();
void Switch();

//=====main program=====
void main(void)
{
    int i, j, k;
    Initial_Lcd();
    while(1)
    {
        clear_screen(); //clear all dots
        DisplayGraphic1(); //display a picture of 128*64 dots
        Switch();
        DisplayGraphic2(); //display a picture of 128*64 dots
        Switch();
    }
}

```

```
//=====initial
void Initial_Lcd()
{
    reset=0;                //Reset the chip when reset=0
    Delay(200);
    reset=1;
    Delay(200);
    Transfer_command(0xe2);  /*软复位*/
    Transfer_command(0x2c);  /*升压步聚 1*/
    Delay(10);
    Transfer_command(0x2e);  /*升压步聚 2*/
    Delay(10);
    Transfer_command(0x2f);  /*升压步聚 3*/
    Delay(10);
    Transfer_command(0x23);  /*粗调对比度, 可设置范围 20~27*/
    Transfer_command(0x81);  /*微调对比度*/
    Transfer_command(0x2d);  /*微调对比度的值, 可设置范围 0~63*/
    Transfer_command(0xa2);  /*1/9 偏压比 (bias) */
    Transfer_command(0xc8);  /*行扫描顺序: 从上到下*/
    Transfer_command(0xa0);  /*列扫描顺序: 从左到右*/
    Transfer_command(0x60);  /*起始行: 从第一行开始*/
    Transfer_command(0xaf);  /*打开显示*/
}

//=====clear all dot martrics=====
void clear_screen()
{
    unsigned char i, j;
    for(i=0; i<9; i++)
    {
        cs1=0;
        Transfer_command(0xb0+i);
        Transfer_command(0x10);
        Transfer_command(0x00);
        for(j=0; j<132; j++)
        {
            Transfer_data(0x00);
        }
    }
}

//=====display a piture of 128*64 dots=====
void DisplayGraphic()
{
    unsigned char *address;  //define pointer
    int i, j;
```



```

address=graphic; //data pointer:point to "graphic[]",one-dimensional array

for(i=0;i<8;i++)
{
    cs1=0;
    Transfer_command(0xb0+i); //set page address,
    Transfer_command(0x10);
    Transfer_command(0x00);
    for(j=0;j<128;j++)
    {
        Transfer_data(*address);
        address++;
    }
}

//=====display a picture of 128*64 dots=====
void DisplayGraphic1()
{
    unsigned char *address; //define pointer
    int i, j;
    address=graphic3; //data pointer:point to "graphic[]",one-dimensional array
    for(i=0;i<8;i++)
    {
        cs1=0;
        Transfer_command(0xb0+i); //set page address,
        Transfer_command(0x10);
        Transfer_command(0x00);
        for(j=0;j<128;j++)
        {
            Transfer_data(*address);
            address++;
        }
    }
}

//=====display a picture of 128*64 dots=====
void DisplayGraphic2()
{
    unsigned char *address; //define pointer
    int i, j;
    address=graphic4; //data pointer:point to "graphic[]",one-dimensional array
    for(i=0;i<8;i++)
    {
        cs1=0;
        Transfer_command(0xb0+i); //set page address,
        Transfer_command(0x10);
        Transfer_command(0x00);

```




```
        for(j=0;j<128;j++)
        {
            Transfer_data(*address);
            address++;
        }
    }
}
//=====transfer command to LCM=====
void Transfer_command(int data1)
{
    cs1=0;
    rs=0;
    rd=0;
    wr=0;
    P1=data1;
    Delay(10);
    rd=1;
    cs1=1;
    rd=0;
}

//-----transfer data to LCM-----
void Transfer_data(int data1)
{
    cs1=0;
    rs=1;
    rd=0;
    wr=0;
    P1=data1;
    rd=1;
    cs1=1;
    rd=0;
}

//=====delay time=====
void Delay(int i)
{
    int j,k;
    for(j=0;j<i;j++)
    for(k=0;k<990;k++);
}

//=====delay time=====
void Delay1(int i)
{
    int j,k;
```

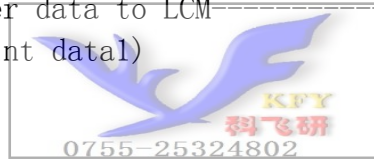
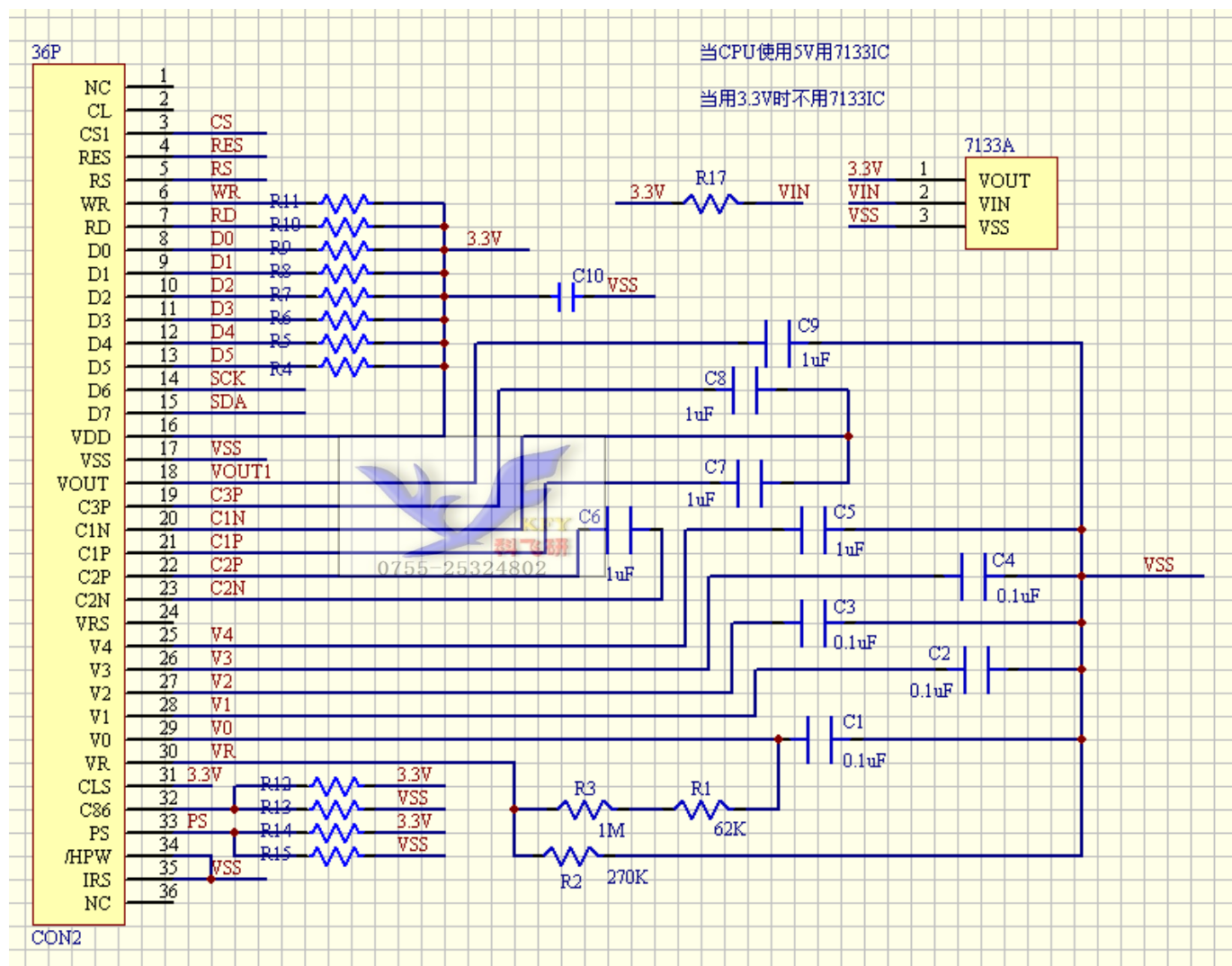


图 9. 串行接口



所有电容请使用 25V 电压。

并行程序：

```
/* Test program for COG12864
   driver IC is:st7565R(or competible)
```

```
*/
#include <reg51.H>
```

```
sbit cs1=P1^1;
sbit reset=P1^0;
sbit rs=P3^0;
sbit sclk=P3^1;
sbit sid=P3^2;
```

```
void transfer_data(int data1);
```

```

void transfer_command(int data1);
char code graphic1[];
char code graphic2[];

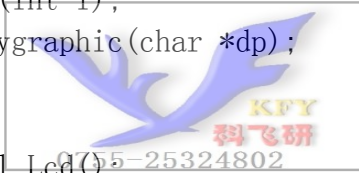
void delay(int i);
void delay1(int i);
void displaygraphic(char *dp);

void Initial_Lcd();
void clear_screen();
void waitkey();

//=====main program=====
void main(void)
{
    int i, j, k;
    Initial_Lcd();
    while(1)
    {
        clear_screen();    //clear all dots
        displaygraphic(graphic1); //display a picture of 128*64 dots
        waitkey();
        displaygraphic(graphic2); //display a picture of 128*64 dots
        waitkey();
    }
}

/*初始化子程序*/
void Initial_Lcd()
{
    reset=0;                //Reset the chip when reset=0
    delay(20);
    reset=1;
    delay(20);
    transfer_command(0xe2);    /*软复位*/
    transfer_command(0x2c);    /*升压步聚 1*/
    delay(5);
    transfer_command(0x2e);    /*升压步聚 2*/
    delay(5);
    transfer_command(0x2f);    /*升压步聚 3*/
    delay(5);
    transfer_command(0x23);    /*粗调对比度, 可设置范围 20~27*/
    transfer_command(0x81);    /*微调对比度*/
    transfer_command(0x2d);    /*微调对比度的值, 可设置范围 0~63*/
}

```



```

transfer_command(0xa2);    /*1/9 偏压比 (bias) */
transfer_command(0xc8);    /*行扫描顺序: 从上到下*/
transfer_command(0xa0);    /*列扫描顺序: 从左到右*/
transfer_command(0x60);    /*起始行: 从第一行开始*/
transfer_command(0xaf);    /*开显示*/

}

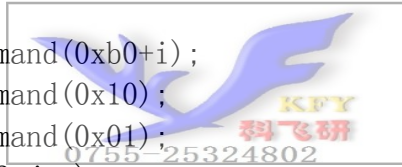
//=====clear all dot martrics=====
void clear_screen()
{
unsigned char i, j;

for(i=0;i<9;i++)
{
    cs1=0;
    transfer_command(0xb0+i);
    transfer_command(0x10);
    transfer_command(0x01);
    for(j=0;j<132;j++)
    {
        transfer_data(0x00);
    }
}
}

//=====display a picture of 128*64 dots=====
void displaygraphic(char *dp)
{
int i, j;
for(i=0;i<8;i++)
{
    cs1=0;
    transfer_command(0xb0+i);    //set page address,
    transfer_command(0x10);
    transfer_command(0x01);
    for(j=0;j<128;j++)
    {
        transfer_data(*dp);
        dp++;
    }
}
}

/*=====写指令=====*/

```



```
void transfer_command(int data1)
{
    char i;
    cs1=0;
    rs=0;
    for(i=0;i<8;i++)
    {
        sclk=0;
        if(data1&0x80) sid=1;
        else sid=0;
        delay1(1);
        sclk=1;
        delay1(1);
        data1=data1<<=1;

    }
}
```



```
/*-----写数据-----*/
```

```
void transfer_data(int data1)
{
    char i;
    cs1=0;
    rs=1;
    for(i=0;i<8;i++)
    {
        sclk=0;
        if(data1&0x80) sid=1;
        else sid=0;
        sclk=1;
        data1=data1<<=1;
    }
}
```

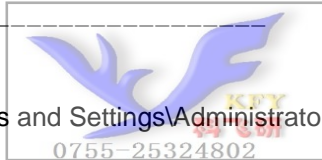
```
/*=====延时=====*/
```

```
void delay(int i)
{
    int j,k;
    for(j=0;j<i;j++)
    for(k=0;k<990;k++);
}
```

```
/*=====延时=====*/
```

```
void delay1(int i)
```

```
{
  int j,k;
  for(j=0;j<i;j++)
  for(k=0;k<10;k++);
}
/*=====等按键=====*/
void waitkey()
{
  repeat:
    if (P2&0x01) goto repeat;
    else delay(1);
    if (P2&0x01) goto repeat;
    else;
}
//-----
char code graphic1[]={
  /*-- 调入了一幅图像：C:\Documents and Settings\Administrator\桌面\12864.bmp --*/
  /*-- 宽度x高度=128x64 --*/
```



省略

此代码是用取模软件“zimo221”取得。如需要可向我司索取该软件。