产品使用说明书

产品类型: 点阵型LCD液晶显示模组

产品型号: 12864B带中文字库

客户：

客户编号:

日期:

确认（盖章）

制造商	客户

目　　录

1 . 修订记录

版本	发行日期	新制/修订内容
V2.0 V2.2	2016-7-25 2020-6-3	新制 增加多功能接口

注：升级版本向下兼容，不做另行通知，如遇兼容问题影响性能请联系本公司解决

2.概述

12864B 是一款带中文字库的点阵型液晶显示模块，可用于显示字母、数字符号、中文字符以及自定义图形。分4位并行、8位并行和串行数据传输方式。提供显示RAM、中文字库和字符发生器。

模块内部共有2M中文字库ROM（CGROM），包含8192个16x16的中文字形，还有32K半宽字母字符ROM（HCGROM），有126个8x16的字母符号字形，还有一块128x64的绘图RAM（DGRAM）可以实现文字和图形混合显示，4x16x16的自定义字符RAM（CGRAM）可提供造字功能，4x16x16的显示RAM（DDRAM）。

显示分辨率: 128 X 64dots

显示颜色及背光颜色: STN 蓝,黄绿,灰; 背光 黑,白,黄绿

偏光膜:全透/半透

观察角度: 6:00

显示占空比: 1/32　驱动偏压: 1/6

中文字符 ROM (CGROM): 2M bits (8192个中文字形)

字符发生器 ROM (CGROM): 16K bits (126个半宽字符)

显示数据 RAM (DDRAM) :64X16 bits (最多4行x16个字，LCD屏最多

显示4行x 8个汉字或16个半宽字符)

尺寸 (Unit: mm)

外形尺寸: 93X70X12.5

可视区域 : 72X40

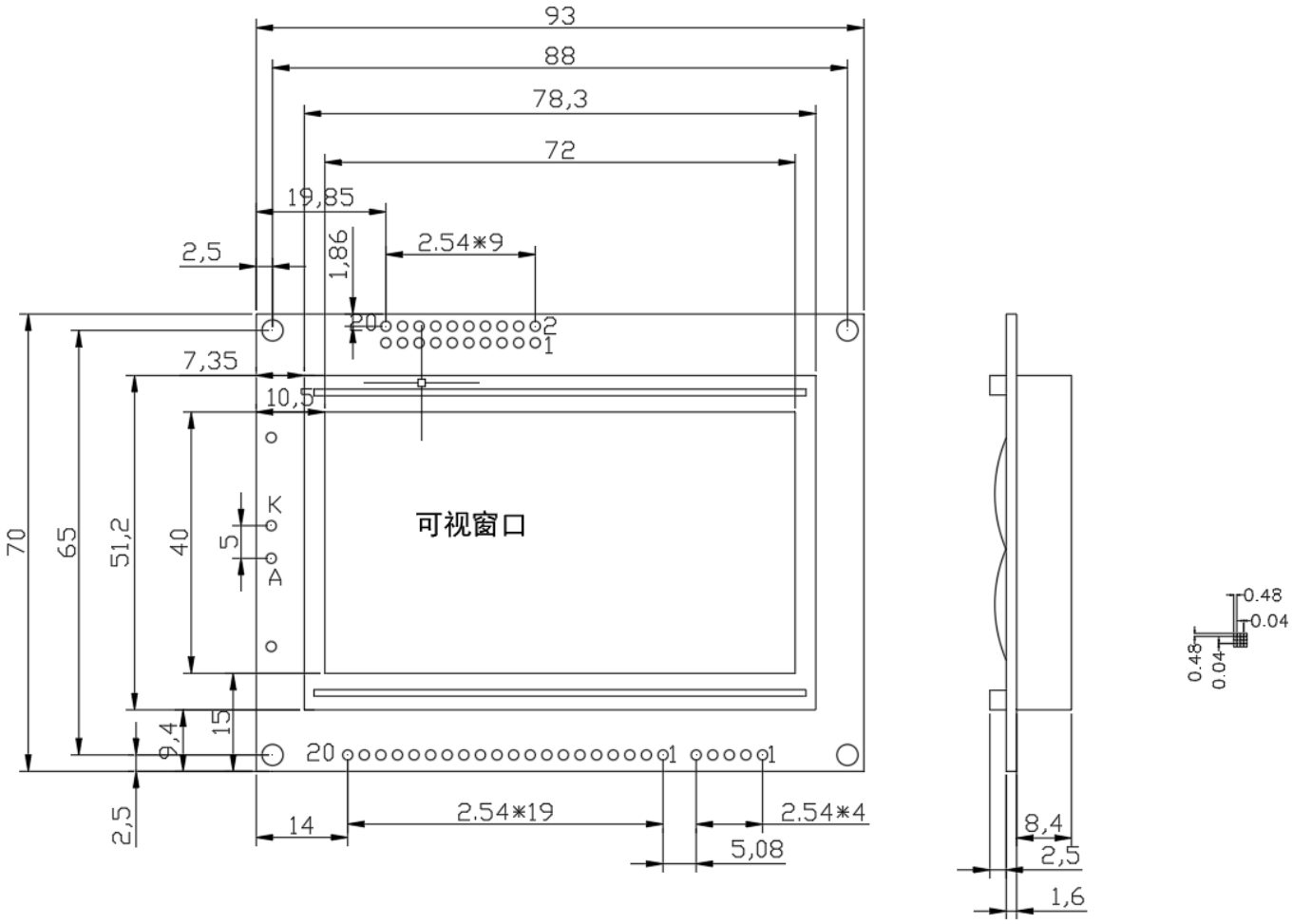
字符尺寸:66.52X33.24　　点尺寸:0.48X0.48

像素间距: 0.52X0.52

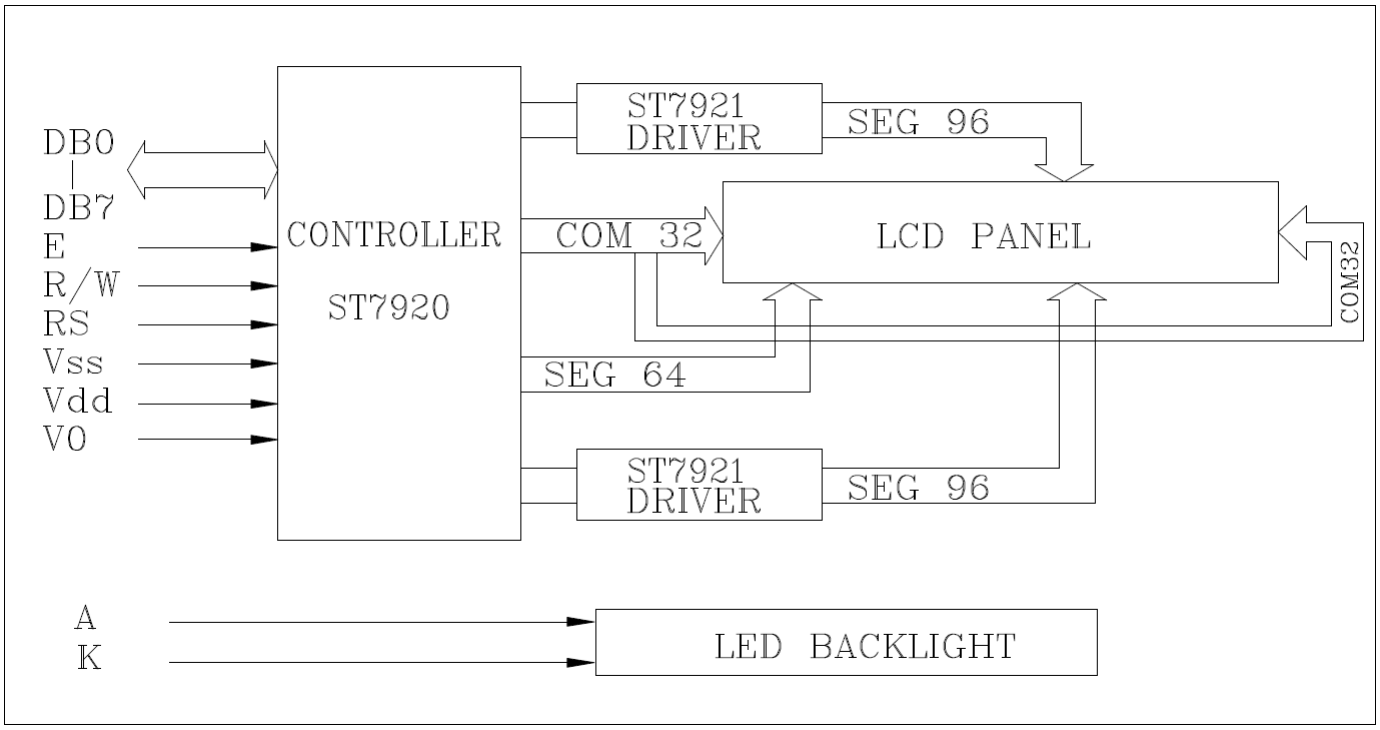
重量：　　g　　对比度：V0外部调节或内部固定对比度

工作电压：　+3. 3V或+5V 默认5V

3. 外形尺寸:



4. 硬件方框图:



5.电气特性

5.1极限参数

参数名称	符号	条件	典型值		单位
			最小值	最大值	
电源电压	Vdd		-0.3	+6.0	V
LCD驱动电压	Vlcd		-0.3	+7.0	V
输入电压	Vi		-0.3	Vdd+0.3	V
工作温度(T)	Top	-	-20	70	℃
储存温度(T)	Tstg	-	-30	80	℃

5.2.1 直流参数1(Ta=25°C,Vdd=4.5V~5.5V)

参数名称	符号	条件	标称值			单位
			最小	典型	最大	
电源电压	Vdd-GND	-	4.5	5.0	5.5	V
工作电流（不包括背光）	Idd	Vdd=5V	-	0.45	0.75	mA
LCD驱动电压	Vdd-V5		3.0	-	Vdd	V
LED背光工作电流	If	Vf=2.8~3.0V	51	54	60	mA
LED背光功耗	Pd		230	270	330	mW
输入高电平	Vih		0.7Vdd	-	Vdd	V
输入低电平	Vil		-0.3	-	0.6	V
输出高电平	Voh	Ioh=-0.1mA	0.8Vdd	-	Vdd	V
输出低电平	Vol	Io1=0.1mA	0	-	0.4	V

5.2.2 直流参数2(Ta=25°C,Vdd=2.7V~4.5V)

参数名称	符号	条件	标称值			单位
			最小	典型	最大	
电源电压	Vdd-GND	-	2.7	3.3	4.5	V
工作电流（不包括背光）	Idd	Vdd=3.3V	-	0.25	0.45	mA
LCD驱动电压	Vdd-V5		3.0	-	7	V
LED背光工作电流	If	Vf=2.8~3.0V	51	54	60	mA
LED背光功耗	Pd		138	179	270	mW
输入高电平	Vih		0.7Vdd	-	Vdd	V
输入低电平	Vil		-0.3	-	0.6	V
输出高电平	Voh	Ioh=-0.1mA	0.8Vdd	-	Vdd	V
输出低电平	Vol	Iol=0.1mA	-	-	0.1	V

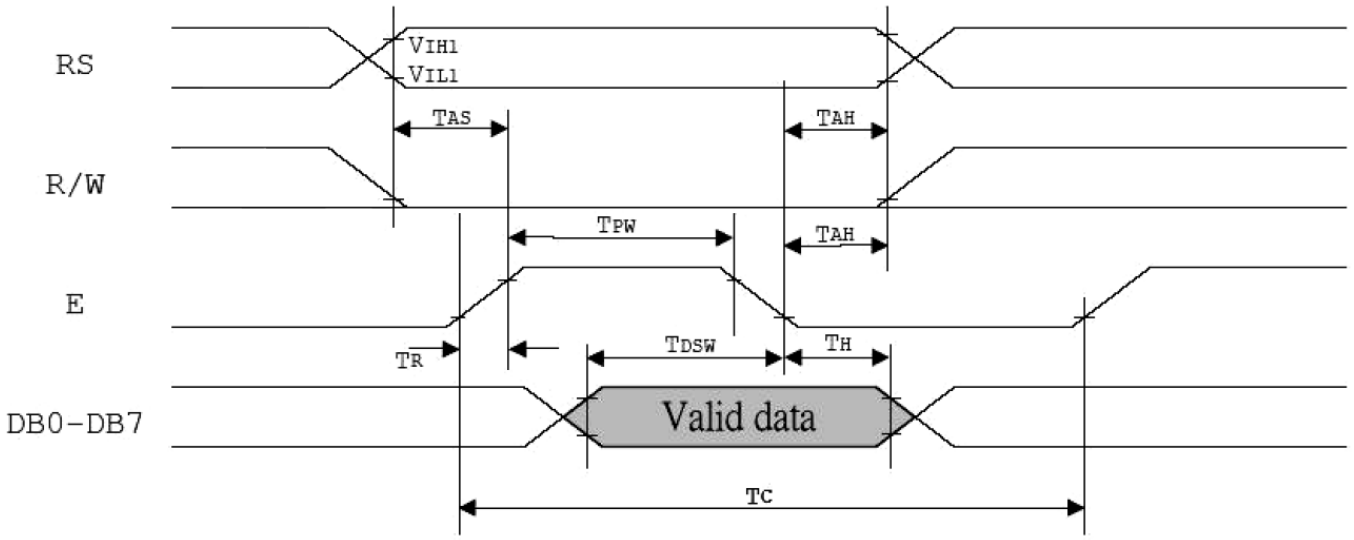
5.3.1 交流参数1(Ta=25°C,Vdd=4.5V~5.5V)并行接口模式

参数	符号	测试条件	最小	典型	最大	单位
写模式（从 MPU 写数据到 LCM ）						
E 周期	T_C	E	1200	-	-	ns
E 脉冲宽度	T_{PW}	E	140	-	-	ns
E 上升/下降时间	T_R，T_F	E	-	-	25	ns
地址建立时间	T_{AS}	RS、RW、E	10	-	-	ns
地址保持时间	T_{AH}	RS、RW、E	20	-	-	ns
数据建立时间	T_{DSW}	DB0～DB7	40	-	-	ns
数据保持时间	T_H	DB0～DB7	20	-	-	ns
读模式（从 LCM 读数据到 MPU）						
E 周期	T_C	E	1200	-	-	ns
E 脉冲宽度	T_{PW}	E	140	-	-	ns
E 上升/下降时间	T_R，T_F	E	-	-	25	ns
地址建立时间	T_{AS}	RS、RW、E	10	-	-	ns
地址保持时间	T_{AH}	RS、RW、E	20	-	-	ns
数据延迟时间	T_{DDR}	DB0～DB7	-	-	100	ns
数据保持时间	T_H	DB0～DB7	20	-	-	ns

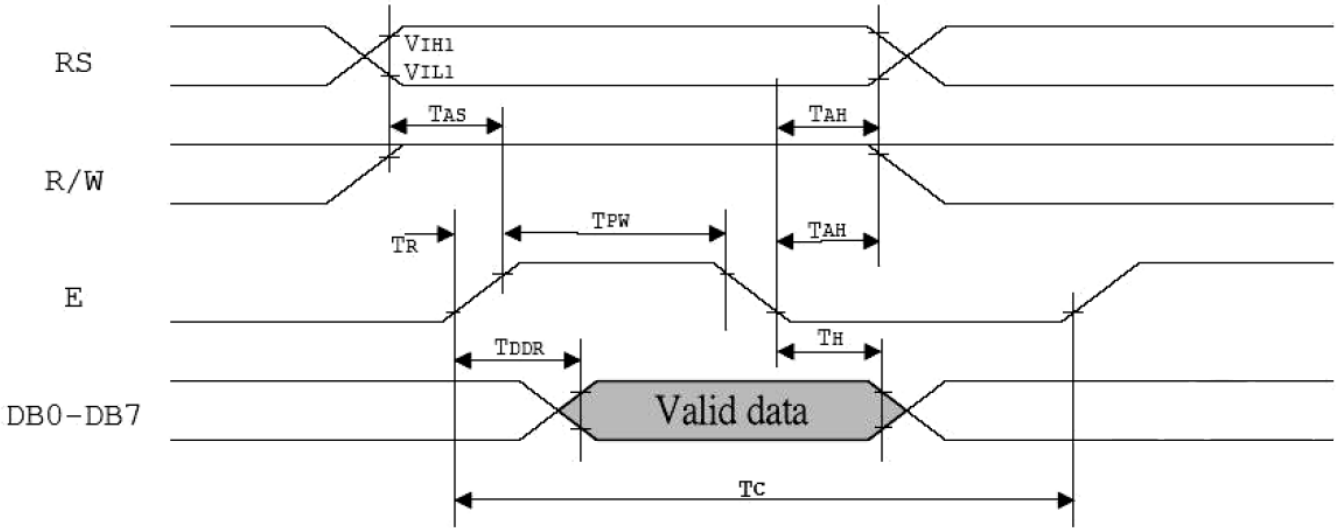
5.3.2 交流参数2(Ta=25°C,Vdd=2.7V~4.5V)并行接口模式

参数	符号	测试条件	最小	典型	最大	单位
写模式（从 MPU 写数据到 LCM ）						
E 周期	T_C	E	1800	-	-	ns
E 脉冲宽度	T_{PW}	E	160	-	-	ns
E 上升/下降时间	T_R，T_F	E	-	-	25	ns
地址建立时间	T_{AS}	RS、RW、E	10	-	-	ns
地址保持时间	T_{AH}	RS、RW、E	20	-	-	ns
数据建立时间	T_{DSW}	DB0～DB7	40	-	-	ns
数据保持时间	T_H	DB0～DB7	20	-	-	ns
读模式（从 LCM 读数据到 MPU）						
E 周期	T_C	E	1800	-	-	ns
E 脉冲宽度	T_{PW}	E	320	-	-	ns
E 上升/下降时间	T_R，T_F	E	-	-	25	ns
地址建立时间	T_{AS}	RS、RW、E	10	-	-	ns
地址保持时间	T_{AH}	RS、RW、E	20	-	-	ns
数据延迟时间	T_{DDR}	DB0～DB7	-	-	260	ns
数据保持时间	T_H	DB0～DB7	20	-	-	ns

并口模式测试波形图



写模式



读模式

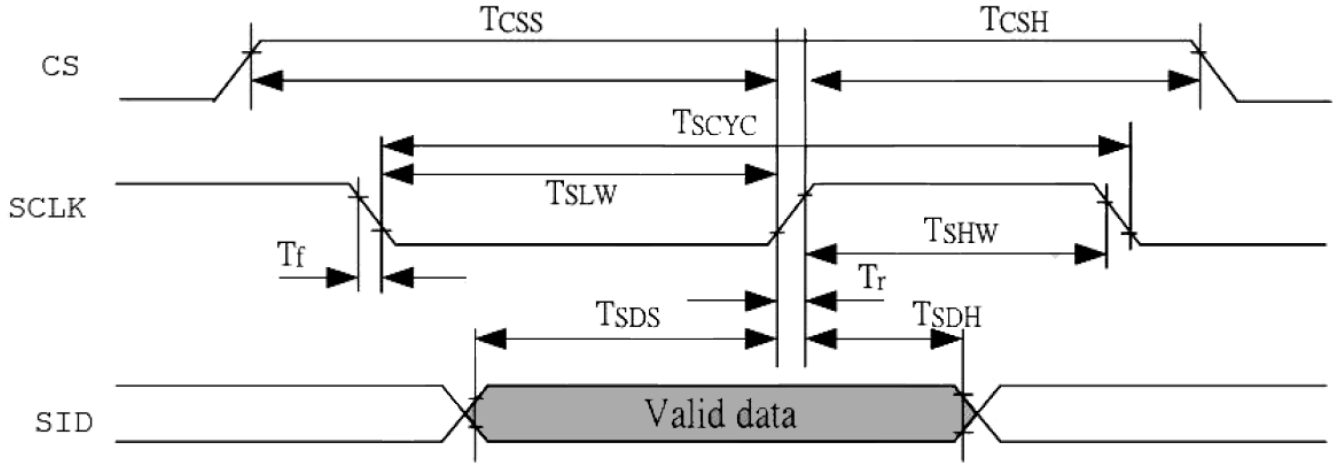
5.3.3 交流参数1(Ta=25°C,Vdd=4.5V~5.5V)串行接口模式

参数	符号	测试条件	最小	典型	最大	单位
SCLK 高脉宽	T_{SHW}	E	200	-	-	ns
SCLK 低脉宽	T_{SLW}	E	200	-	-	ns
SID 数据建立时间	T_{SDS}	RW	40	-	-	ns
SID 数据保持时间	T_{SDH}	RW	40	-	-	ns
CS 建立时间	T_{CSS}	RS	60	-	-	ns
CS 保持时间	T_{CSH}	RS	60	-	-	ns

5.3.4 交流参数2(Ta=25°C,Vdd=2.7V~4.5V)串行接口模式

参数	符号	测试条件	最小	典型	最大	单位
SCLK 高脉宽	T_{SHW}	E	300	-	-	ns
SCLK 低脉宽	T_{SLW}	E	300	-	-	ns
SID 数据建立时间	T_{SDS}	RW	40	-	-	ns
SID 数据保持时间	T_{SDH}	RW	40	-	-	ns
CS 建立时间	T_{CSS}	RS	60	-	-	ns
CS 保持时间	T_{CSH}	RS	60	-	-	ns

串口模式测试波形图



串行接口模式

6.接口说明

脚号	符号	功能	备注
1	VSS	供电电源负极	
2	VDD	供电电源正极	
3	V0	LCD偏压供电	可调节对比度
4	RS(CS)	数据/指令选择(H:数据 L: 指令)(串行芯片使能)	
5	R/W(SID)	读/写选择(H:读 L:写)(串行数据传输)	
6	E(SCLK)	使能信号(串行时钟传输)	
7	D0	数据位 0	
8	D1	数据位1	
9	D2	数据位2	
10	D3	数据位3	
11	D4	数据位4	
12	D5	数据位5	
13	D6	数据位6	
14	D7	数据位7	
15	PSB	接口模式选择（H:并口， L:串口）	与模块背面短接点方向必须一致
16	NC	空脚	
17	RST	系统复位脚	
18	Vout	LCD倍压输出	
19	LEDK	LED 背光负	
20	LEDA	LED 背光正	

通过Vout连接一个电阻到V0可调节对比度

模块背面的P-S短接点为串并口模式选择，注意两个短接点不能同时短接，否则会引起模块短路

7.指令说明

模块具有4位/8位并行及串行通讯模式，经由模块背面的P-S短接点进行选定，P为4位/8位并行通讯模式，S为串行通讯模式。

在读写操作时，使用到2个8位寄存器，一个是数据寄存器DR，另一个是指令寄存器IR。数据寄存器DR作为写入和读出DDRAM/CGRAM/GDRAM以及IRAM的数据，在存取目标RAM的地址时，通过指令来选择，每次对数据寄存器（DR）存取动作都将自动以上次选择的目标RAM地址当主体来写入或读取。

通过设置RS/RW位的各种操作:

RS	RW	操作
L	L	写指令操作（MPU写指令代码至IR）
L	H	读忙标志（DB7）和地址计数器（DB0~DB6）
H	L	写数据操作（MCU写数据至DR）
H	H	读数据操作（MCU从DR读出数据）

7.1 忙标志（BF）

BF为高，表示内部操作正在进行，所以在这个时间内，下一条指令将不能被执行。当RS="0"且R/W="1"（读指令操作时），BF的值可以从DB7口读出，在执行下一条指令时，必须确认BF不为"1"。

7.2 地址计数器（AC）

地址计数器（AC）用来储存DDRAM/CGRAM/GDRAM之一的地址，它可以由设定指令寄存器（IR）来改变，之后只要写入或读出DDRAM/CGRAM/GDRAM的值时，地址计数器（AC）的值就会自动加一，当RS="0"且R/W="1"时，地址计数器中的值可以从DB0~DB6读出。

7.3 显示数据RAM（DDRAM）

模块的DDRAM提供64x2个8BIT的空间，最多可以控制4行16字（64个字）的中文字形显示，当写入显示数据RAM时，可以分别显示CGROM，HCGROM与CGRAM的字形，模块可以显示三种字形，分别是半宽的HCGROM字形、CGRAM字形及中文CGROM字形，三种字形的选 择由在DDRAM中写入的编码选择，在0000H~0006H的编码中选择CGRAM的自定义字形，02H~7FH的编码选择半宽英数字的字形，至于A1以上的编码将自动结合下一个8BIT字节组成两个BYTE编码形成中文字形的编码BIG5（A140H~D75FH）或GB（A1A0H~F7FFH），详细的各种字形编码如下:

1.显示半宽字形：将8BIT数据写入DDRAM中，范围为02H~7FH的编码。

2.显示CGRAM字形：将16BIT数据写入到DDRAM中，总共有0000H，0002H，0004H，0006H四种编码。

3.显示中文字形：将16位数据写入DDRAM中

范围BIG5（A140H~D75FH）或GB（A1A0H~F7FFH），将16位数据通过连续写入两个8位的方式写入DDRAM，先写入高8位，再写入低8位。

7.4 中文字形生成ROM（CGROM）及半宽字形ROM（HCGROM）

字形生成ROM具有8192个16x16点的中文字形图像以及126个16x8点的字母数字符号图像，它使用两个BYTE来提供字形编码选择，配合DDRAM将要显示的字形码写入到DDRAM上，硬件将自动根据编码从CGROM中将要显示的字形显示在屏幕上。

7.5 字形生成RAM（CGRAM）

字形生成RAM具有自定义字形（造字）功能，可以提供四组16x16点的 自定义字形空间，用户可以将内部字形没有提供的字形图像自定义到CGRAM中，然后CGRAM中的自定义字形

便可以通过DDRAM显示在屏幕上。

CGRAM字形与中文字形的编码只可出现在每一地址计数器的起始位置，如下表：

80		81		82		83		84		85		86		87		88		89		8A		8B		8C		8D		8E		8F	
H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L
L	C	M	6	4	3	2																									
中		文		字		库		L	C	D		模		组																	

DDRAM/CGRAM地址对应图：

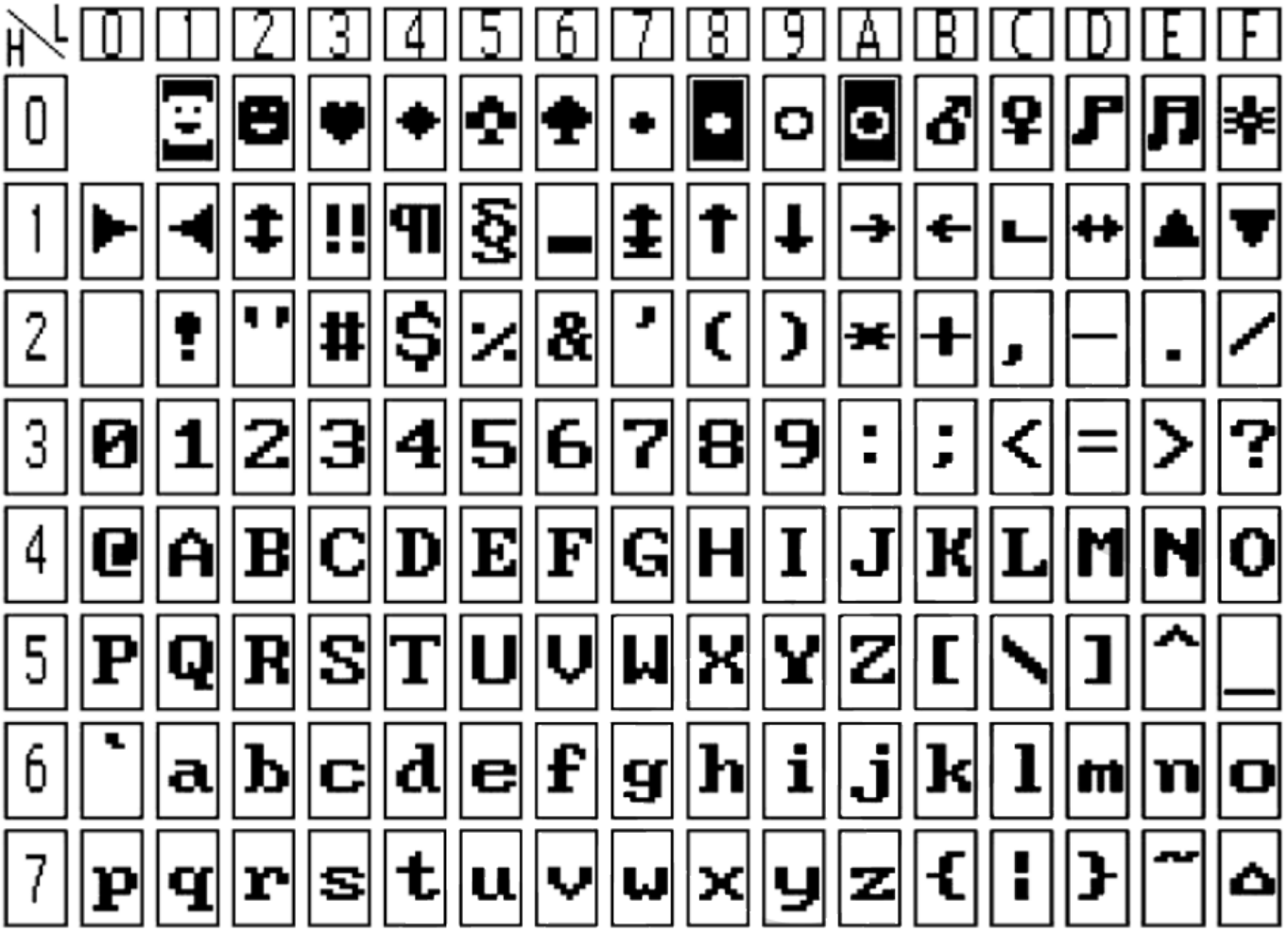
DDRAM 数据					CGRAM 地址		CGRAM 数据高 8 位										CGRAM 数据低 8 位							
B15～B4	B3	B2	B1	B0	B5	B4	B3	B2	B1	B0	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0	X	00	X	00			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
							0	0	0	1	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
							0	0	1	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0
							0	0	1	1	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0
							0	1	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0
							0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
							0	1	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0
							0	1	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0
							1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0
							1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
							1	0	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
							1	0	1	1	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0
							1	1	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0
							1	1	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
							1	1	1	0	0	0	1	0	0	0	0	1	1	0	1	1	0	0	0	0	0
							1	1	1	1	0	1	0	0	1	1	1	0	0	0	0	0	1	1	0	0
0	X	01	X	01			0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0
							0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0
							0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
							0	0	1	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0
							0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0
							0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
							0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
							0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
							1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0
							1	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0	0	0
							1	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0	0	0
							1	0	1	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0
							1	1	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	1	0	0
							1	1	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
							1	1	1	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
							1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

注：1.DDRAM数据（字形代码）1~2位和CGRAM地址的4~5位同步吻合（2位：4组图像）。

2.CGRAM地址0~3位指定字形图像的列地址，总共指令16行（4位），第16行是光标的显示区域，光标显示和第16行的数据采用逻辑OR的方式产生显示结果。

3.显示图像的行列图素对应到CGRAM数据0~15位（15位在最左边）。

4.选择到CGRAM的图像数据，DDRAM数据4~15位须设为0，至于0位及3位财可为任意值



16x8半宽字形符号表

7.6 绘图RAM（GDRAM）

绘图显示RAM提供128x64个8位存储空间（由扩展指令设定绘图RAM地址），最多可以控制256x64点的二维绘图空间，在更改绘图RAM时，由扩展指令设定GDRAM地址，先设垂直地址再设水平地址（连续写两个8位数据完成垂直和水平坐示地址设定），再写入两个8位数据到绘图RAM，而地址计数器AC会自动加一，整个写入绘图RAM的步骤如下:

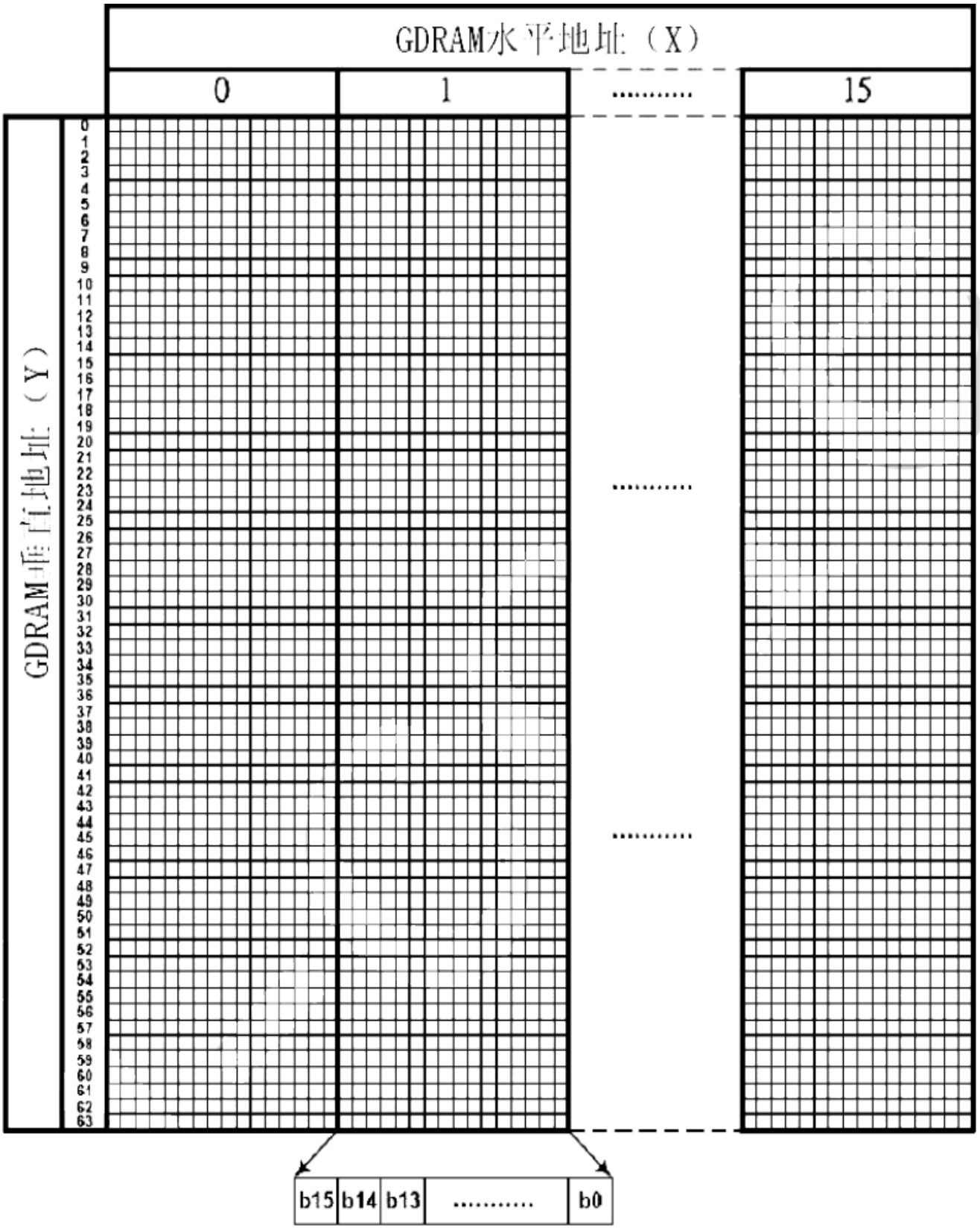
1.先将垂直的8位坐示（Y）写入绘图RAM地址

2.再将水平坐示（X）写入绘图RAM地址

3.将D15~D8写入到RAM中（写入第一个数据）

4.将D7~D0写入到RAM中（写入第二个数据）

GDRAM坐标地址与数据排列顺序对照表


GDRAM水平地址（X）
0
1
.....
15
GDRAM垂直地址（Y）
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
.....
.....
b15
b14
b13
.....
b0

7.6 指令描述

模块提供两套控制指令，基本指令和扩展指令如下：

指令表1（RE=0：基本指令集）

指令	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	执行时间	描述
清除显示	0	0	0	0	0	0	0	0	0	1	1.6ms	将20H写入DDRAM，将地址计数器地址设为00H
地址归零	0	0	0	0	0	0	0	0	1	-	72us	将地址计数器地址设为00H，并将光标恢复至初始位置，DDRAM内容保持不变
输入模式设置	0	0	0	0	0	0	0	1	I/D	S	72us	在数据读取与写入时，设置光标移动方向及指定显示移动
显示开/关	0	0	0	0	0	0	1	D	C	B	72us	设置显示开关、光标开关，光标的位置及反白控制位
移位控制	0	0	0	0	0	1	S/C	R/L	-	-	72us	设置光标移动、显示移动方向的控制位，DDRAM数据保持不变
功能设置	0	0	0	0	1	DL	-	0 RE	-	-	72us	DL=1：8位传输DL=0：4位传输 RE=1：扩展指令RE=0：基本指令
设置CGRAM地址	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	72us	在地址计数器内设置CGRAM地址需确认扩展指令中SR=0(卷动地址或RAM地址选择)
设置DDRAM地址	0	0	1	0 AC6	AC5	AC4	AC3	AC2	AC1	AC0	72us	在地址计数器内设置DDRAM地址，AC6固定为0
读忙标志&地址	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	0us	读取忙标志位BF，地址计数器中的内容同时被读出
写数据	1	0	D7	D6	D5	D4	D3	D2	D1	D0	72us	写数据至内部RAM（DDRAM/CGRAM/GDRAM）
读数据	1	1	D7	D6	D5	D4	D3	D2	D1	D0	72us	从内部RAM(DDRAM/CGRAM/GDRAM)中读取数据

注:“-”为不考虑

指令表2（RE=1：扩展指令集）

指令	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	执行时间	描述
待机模式	0	0	0	0	0	0	0	0	0	1	72ms	进入待机模式，执行任何其它指令都可终止待机模式
卷动地址或RAM地址选择	0	0	0	0	0	0	0	0	1	SR	72us	SR=1：允许输入垂直卷动址 SR=0：允许设定CGRAM地址（基本指令）
反白选择	0	0	0	0	0	0	0	1	R1	R0	72us	选择4行中任一行反白显示，R1、R0初始值00，执行第一次反白显示，再执行一次正常显示
扩展功能设置	0	0	0	0	1	DL	-	1 RE	G	0	72us	DL=1：8位传输DL=0：4位传输 RE=1：扩展指令RE=0：基本指令 G=1:绘图显示开 G=0:绘图显示关
设置IRAM地址或卷动地址	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	72us	SR=1：AC5~AC0为垂直卷动地址
设置GDRAM地址	0	0	1	0 0	AC5 0	AC4 0	AC3 AC3	AC2 AC2	AC1 AC1	AC0 AC0	72us	在地址计数器内设置GDRAM地址，先设垂直地址再设水平地址 垂直地址范围AC5~AC0 水平地址范围AC3~AC0

注： 1.当模块接受指令前，必须先确认模块处于非忙状态，即BF需为0，方可接受新指令。

2.RE为基本指令和扩展指令选择，当变更RE后，之后的指令以最后的状态为准，除非再次变更。

7.6.1 清除显示

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0

0	0	0	0	0	0	0	0	0	1

通过写入20H（空格代码）至所有的DDRAM地址和设置地址计数器为00H,可以清除显示数据，将光标放在初始状态位置，设置输入模式为递增（I/D为高）。

7.6.2 地址归零

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	1	-

“-”为不考虑

指令是将光标回到起始位置，将DDRAM地址设置为00H写入地址计数器，并将显示改为初始状态，DDRAM中的数据保持不变。

7.6.3 输入模式

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	1	I/D	S

设置光标和显示的移动方向

I/D：DDRAM地址（光标或闪烁）的递增或递减

当I/D为1时，光标闪烁向右移，DDRAM地址为递增; 当I/D为0时，光标闪烁向左移，DDRAM地址递减。

S：显示移位

当S为0时，对DDRAM或CGRAM读写操作时，整个显示不会位移。当S为1时，对DDRAM写操作是，整个显示的移位将根据I/D设定的方向位移。

7.6.4 显示开关控制

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	1	D	C	B

显示/光标及闪烁控制

D：显示开关控制位

当D为1时，显示开启；当D为0时，显示关闭，但DDRAM中显示数据保持不变。

C：光标开关控制位

当C为1时，光标开；当C为0时，光标消失，但I/D寄存器保存它的数据。

B：光标位置反白控制位

当B为1时，光标位置反白开，将光标所在的数据反白显示；当B为0时，光标位置反白关。

7.6.5移位控制

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	1	S/C	R/L	-	-

设定光标的移动与显示的移位控制位，这个指令不改变DDRAM内容

S/C	R/L	操作
0	0	光标向左移，地址计数器递减1
0	1	光标向右移，地址计数器递增1
1	0	所以显示向左移，光标跟随显示移位，地址计数器不变
1	1	所有显示向右移，光标跟随显示移位，地址计数器不变

7.6.6 功能设置

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	DL	-	RE	-	-

DL：接口数据宽度控制位

当DL为1时，表示8位总线连接至MPU

当DL为0时，表示4位总线连接至MPU，当为4位总线模式时，8位数据需要通过传输4位数据2次完成。

RE：指令集选择控制位

当RE为1时，为扩展指令集；当RE为0时，为基本指令集。

7.6.7 设置CGRAM地址

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0

将CGRAM地址置入地址计数器。

AC范围为00H~3FH

需确认扩展指令中SR=0（卷动地址和RAM地址选择）

7.6.8 设置DDRAM地址

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0

将DDRAM地址置入地址计数器。

第一行AC范围为80H~8FH

第二行AC范围为90H~9FH

第三行AC范围为A0H~AFH

第四行AC范围为B0H~BFH

注：四行中只显示两行。

7.6.9 读忙标志和地址

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0

该指令显示模块是否处于内部工作中，如果BF为1，内部工作在进行中，需要等待直到BF被置0时，下条指令才能被执行，在这条指令中，同样可以读到地址计数器内的值。

7.6.10 写数据到RAM

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	0	D7	D6	D5	D4	D3	D2	D1	D0

写入8位数据至RAM，当写入后会使（AC）改变，每个RAM地址（CGRAM，DDRAM…）都可连续写入两个8位数据，在写入第二个数据操作后地址计数器自动加一。

7.6.11 从RAM中读数据

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	1	D7	D6	D5	D4	D3	D2	D1	D0

从RAM中读取数据，当读取后会使（AC）改变。

当下设定地址指令后（CGRAM，DDRAM…），若要读取数据时需先DUMMY READ一次才会读出正确数据，第二次读取时则不需要DUMMY READ，除非又下了设定地址指令才需要再次DUMMY READ。

扩展指令集说明

7.6.12 待机模式

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	0	1

进入待机模式，执行任何其它指令都可终止待机模式；这个指令不改变RAM内容。

7.6.13 卷动地址或RAM地址选择

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	1	SR

当SR=1，允许输入垂直卷动地址

当SR=0，允许设定CGRAM地址（基本指令）

7.6.14 反白选择

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	1	R1	R0

先择4行中的任一行作反白显示，并可决定反白与否。

R1，R0初值为00，当第一次设定时为反白显示，再一次设定时为正常显示。

R1	R0	说明
L	L	第一行反白或正常显示
L	H	第二行反白或正常显示
H	L	第三行反白或正常显示
H	H	第四行反白或正常显示

注：四行中只显示两行。

7.6.15 扩展功能设置

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	DL	-	RE	G	-

DL：接口宽度控制位

当DL=1，为8位MPU控制接口

当DL=0，为4位MPU控制接口

RE：指令集选择控制位

当RE=1，为扩展指令集

当RE=0，为基本指令集

G：绘图显示控制位

当G=1，绘图显示开

当G=0，绘图显示关

同一指令的动作不可同时改变RE及DL、G，需先改变DL或G后再改变RE才可确保正确设定。

7.6.16 设定卷动地址

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0

SR=1：AC5~AC1为垂直卷动地址

7.6.17 设定绘图GDRAM地址

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	0	AC5	AC4	AC3	AC2	AC1	AC0

设定GDRAM地址到地址计数器（AC）

先设垂直地址再设水平地址（连续写入两个8位数据以完成垂直与水平的坐标地址）

垂直地址范围AC5~AC0　　水平地址范围AC3~AC0

绘图RAM 的地址计数器只会对水平地址自动加一，当水平地址等于0FH,会被设为00H，垂直地址不会自动加一，所以当连续写数据时，需要程序判断垂直地址是否需要重新设定。

8.操作时序

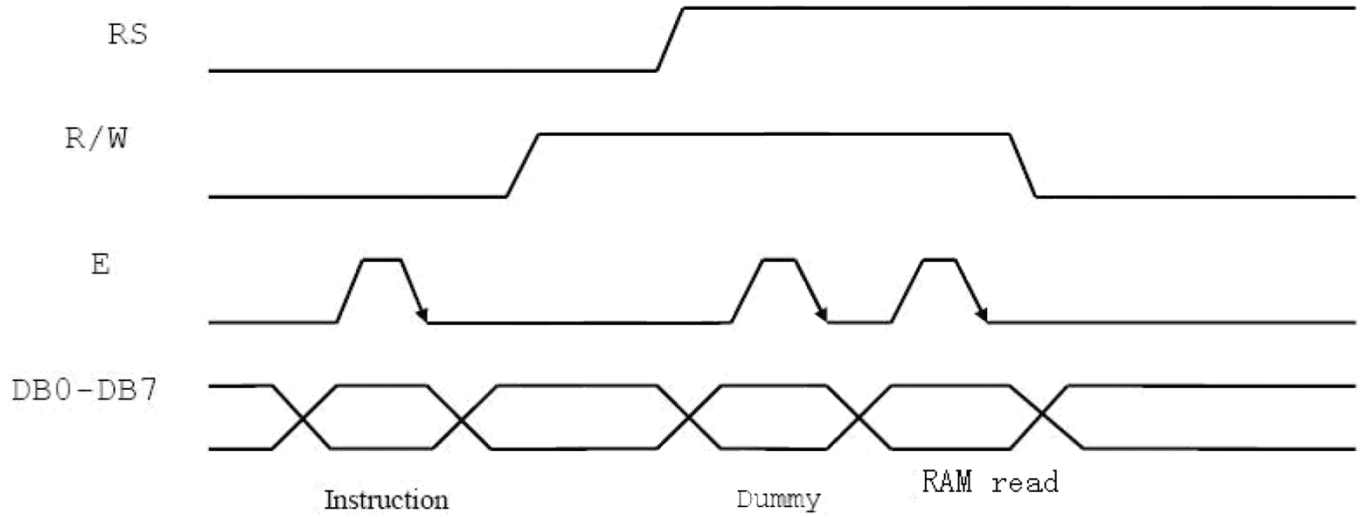
8.1 并行数据传输接口

当接口选择并行传输模式时，可由指令改变DL来选择8位或4位并行接口，主控制系统将由（RS、RW、E、DB0~DB7）来完成数据传输。

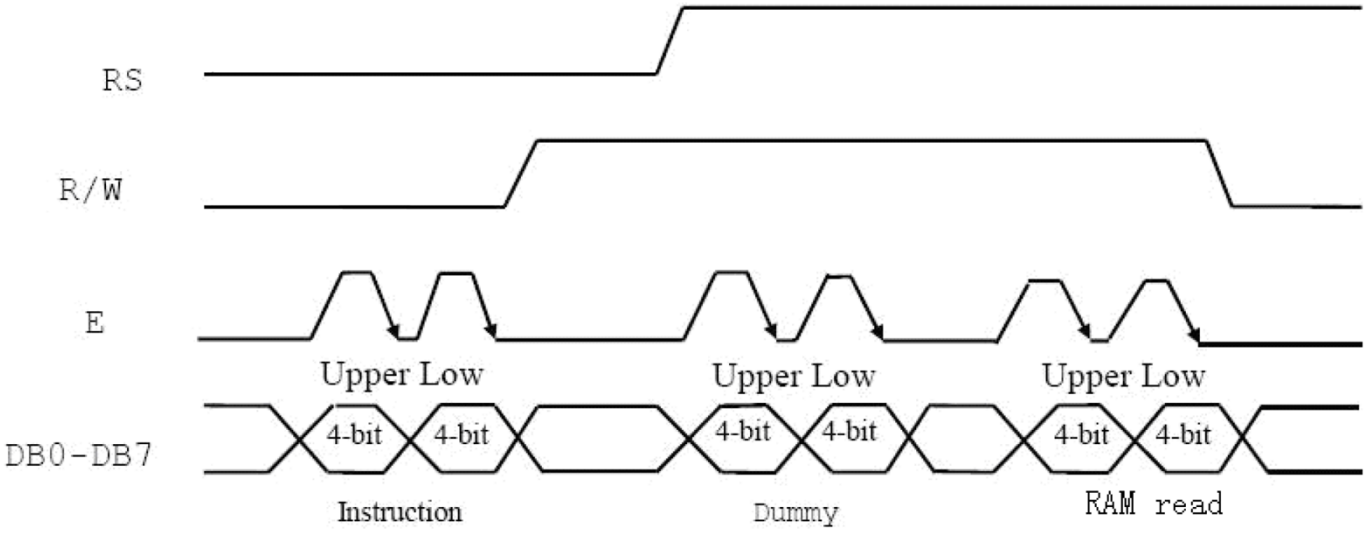
坐一个完整的流程来看，当下设定地址指令后（CGRAM、DGRAM、IRAM…），若要读取数据时需先DUMMY READ一次才会读到正确数据，第二次读取时则不需DUMMY READ，除非重新设定地址指令才需再次DUMMY READ。

在4位并行传输模式中，每一个8位指令或数据都将被分成两个8位数据，高4位被放在第一个8位数据的高4位，低4位被放在第二个8位数据的高4位，两个8位数据的低4位不做考虑。

时序如下图所示:



8位并行总线模式时序图



4位并行总线模式时序图

8.2 串行数据传输接口

当接口选择为串行传输模式时，将使用两条数据传输线作为串行数据的传送，主控制系统将配合传输同步时钟端（SCLK）与串行数据接收端（SID）来完成串行传输的动作。

当主控系统连接多个模块时，片选脚（CS）将要被配合使用，在片选脚（CS）设为高电平时，串行传输数据才会被接收，当片选脚（CS）为低电平时模块内部的串行传输计数与串行数据将会被重置，也就是说在此状态下，传输中的数据将被终止清除，并且将待传输的串行数据计数重设回第一位，在一个系统只用到一个模块的情况下，只需要使用同步时钟和串行数据传输脚，片选脚固定接到高电平即可。

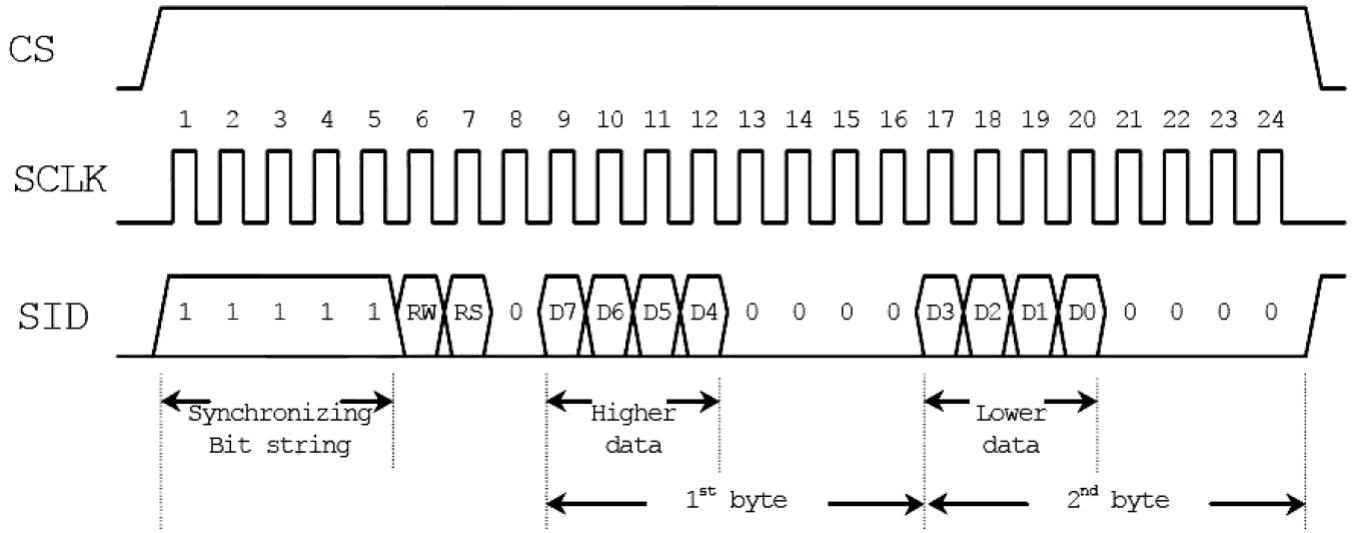
串行传输数据时需考虑指令执行时间，必须确实等到前一个指令完全执行完成才能传送下一

条指令，因为模块内部没有串行发送/接收缓冲区。

从一个完整的串行传输流程来看，一开始先传输8位起始位，模块需接收到5个1（同步码），此时传输计数将被重置并且串行传输将被同步，再跟随的两位分别设定为传输方向位（RW）及寄存器选择位（RS），最后第8位为0.

在接收到同步码及RW和RS数据的8位起始码后，每一个8位指令将被分为两个8位数据传输，高4位被放在第一个8位的LSB部分，低4位被放在第二个8位的LSB部分，其它4位则都为0.

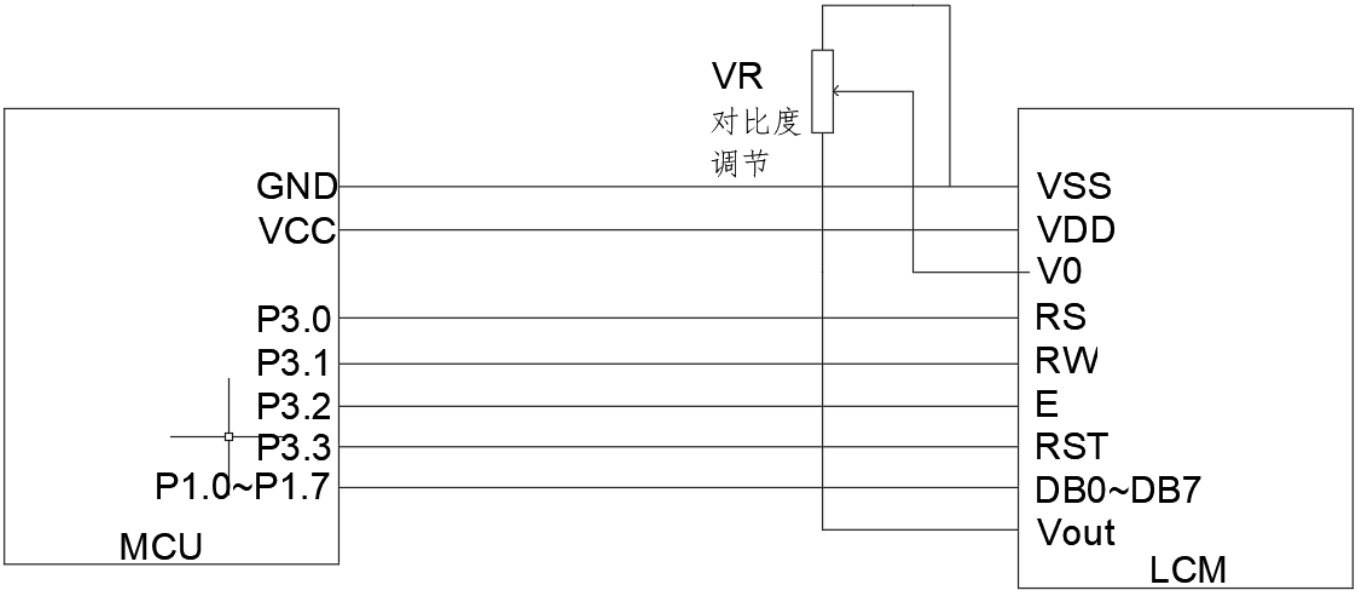
时序如下图所示：



串行模式数据传输时序图

9.应用例程

9.1 并行传输接线图



并行传输接线方法

9.2 并行C51例程

```
#include <STC15.H>
#include <string.h>
#include <INTRINS.H>
#define uchar unsigned char
#define uint  unsigned int
#define db  P1
sbit rs=P3^0;
sbit rw=P3^1;
sbit e=P3^2;
sbit rst=P3^3;
 uchar code IC_DAT1[]={
"0123456789:;<=>?"
"PQRSTUVWXYZ[/]^_"                    //0x80
"@ABCDEFGHIJKLMNO"
"'abcdefghijklmno"   };
uchar code IC_DAT2[]={
"湖南旭阳显示科技"                    //0x88
"深圳市耀兴阳实业"                    //0x98
"有限公司湖南芷江"
"有限公司专业模组"};
//////////////////////////////////////////////////////////////////////
// Bitmap点阵数据表                                                 //
// 图片: D:\..    12864.bmp,横向取模左高位,数据排列:从左到右从上到下   //
// 图片尺寸: 128 * 64                                               //
//////////////////////////////////////////////////////////////////////
unsigned char code nBitmapDot[] =              // 数据表
{
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7F,0xFB,0xC0,0xCF,0xF3,0xE0,0xDF,0xFE,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7F,0xFB,0xC0,0xCF,0xF3,0xE0,0xDF,0xFE,
    0x00,0x01,0x20,0x84,0x6C,0x08,0x40,0x00,0x60,0x1B,0xE1,0xF7,0x8F,0x07,0xD8,0x06,
    0x5F,0xF1,0x24,0x84,0xB4,0x44,0x47,0xBF,0x6F,0xDB,0xFF,0xF3,0x8F,0x0F,0xDB,0xF6,
    0x32,0x91,0x24,0x95,0x6C,0x24,0x84,0xA1,0x6F,0xD9,0xFE,0x3D,0x8C,0x04,0x1B,0xF6,
    0x04,0x43,0xA4,0x8E,0xB4,0x20,0x84,0xA1,0x6F,0xDB,0xFE,0xBD,0xC8,0xBE,0x1B,0xF6,
    0x09,0x21,0x24,0x84,0x48,0x01,0x05,0x21,0x6F,0xD8,0xFF,0xFF,0xE1,0xFF,0x1B,0xF6,
    0x41,0x01,0x24,0x9F,0x7C,0x00,0x05,0x3F,0x6F,0xD8,0x71,0xE1,0x80,0x8E,0x1B,0xF6,
    0x2F,0xF1,0x24,0x8A,0xD0,0xFF,0xE4,0xA1,0x6F,0xD8,0x31,0xC1,0x80,0x04,0x1B,0xF6,
    0x01,0x01,0x24,0x8A,0x7C,0x00,0x04,0xA1,0x60,0x1B,0x1E,0x01,0xF0,0x7B,0xD8,0x06,
    0x03,0x81,0xA4,0x8A,0x50,0x11,0x04,0xA1,0x7F,0xFB,0x0E,0x49,0xFA,0x77,0xDF,0xFE,
    0x25,0x43,0x24,0x8A,0x7C,0x20,0x87,0x21,0x7F,0xFB,0x36,0x4D,0x93,0x64,0xDF,0xFE,
    0x59,0x30,0x44,0x8B,0x50,0x40,0x44,0x3F,0x00,0x03,0x9E,0xF2,0x6E,0x60,0xC0,0x00,
    0x01,0x00,0x80,0x92,0x7C,0x80,0x24,0x21,0x00,0x03,0x0E,0xF6,0x6F,0x60,0xC0,0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0xDB,0xD9,0xFD,0x80,0x7C,0x3E,0x3E,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x6F,0xD9,0xC9,0xFF,0x81,0x7F,0x7E,0xFE,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x60,0x60,0xC0,0x7E,0x33,0x99,0xDE,0xF6,
```

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0x19,0x99,0xE9,0x93,0x1F,0x1E,0x76,
0x02,0x00,0x44,0x01,0x00,0xF7,0xC0,0x88,0x0F,0x1B,0x09,0xCD,0xBB,0x1F,0x1C,0xFE,
0x7F,0xF0,0x44,0x01,0x00,0x94,0x40,0x88,0x7F,0x60,0xF6,0x00,0x4F,0x07,0xDF,0xCE,
0x40,0x10,0x44,0x1F,0xFC,0x94,0x40,0x88,0x7F,0x61,0xE6,0x10,0xEF,0x1F,0xDF,0xFE,
0x09,0x00,0x44,0x02,0x00,0xA7,0xC1,0x04,0x73,0x9B,0x01,0xB1,0xF0,0xDC,0xF8,0x38,
0x05,0x02,0x44,0x87,0xF8,0xA4,0x41,0x24,0x03,0xF1,0x87,0x3D,0xF1,0x9C,0x74,0x30,
0x11,0x01,0x45,0x0C,0x08,0x94,0x42,0x22,0x83,0xE3,0xC6,0x08,0x81,0x9C,0x26,0xF0,
0x09,0x00,0xC6,0x17,0xF8,0x97,0xC4,0x41,0x6F,0xDF,0xC9,0x80,0x00,0x9F,0x1B,0xCE,
0x7F,0xF0,0x44,0x04,0x08,0x95,0x20,0x40,0x6F,0x9F,0xD9,0x25,0x10,0xDF,0x19,0xDE,
0x02,0x00,0x44,0x07,0xF8,0x95,0x40,0x88,0x70,0x00,0xC0,0x12,0x96,0xE4,0x78,0x08,
0x04,0xC0,0x44,0x04,0x08,0xE4,0x81,0x04,0x50,0x30,0xC6,0x00,0x00,0x7F,0x39,0x3E,
0x18,0x20,0x44,0x04,0x08,0x86,0x43,0xFE,0x00,0x78,0xC6,0x35,0x60,0x7F,0x7B,0x3E,
0x60,0x13,0xFF,0x84,0x18,0x84,0x20,0x02,0x1F,0xC4,0x06,0x08,0x10,0xC0,0xE6,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x3F,0x8E,0x06,0x00,0x0C,0x80,0xF6,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x6C,0xDE,0x20,0x00,0x80,0xDF,0x18,0x1E,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x63,0x63,0xF0,0x00,0xC0,0x7D,0x01,0x0E,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x63,0x23,0xCC,0x09,0x40,0x60,0xC3,0x86,
0x00,0x00,0x10,0x1F,0xFC,0x08,0x00,0x1E,0x78,0x78,0xDF,0x00,0x80,0xE6,0x7F,0xF8,
0x3F,0xE0,0x10,0x01,0x40,0x04,0x03,0xE0,0x3C,0x78,0xDF,0x05,0xE2,0xF4,0x7F,0xFC,
0x00,0x20,0x10,0x01,0x40,0xFF,0xE0,0x40,0x04,0xE0,0xC3,0x01,0xF2,0x78,0x26,0xFE,
0x7F,0xA1,0xFF,0x0F,0xF8,0x20,0x80,0x84,0x0C,0xF0,0x08,0x03,0xF6,0x7F,0x74,0x7E,
0x00,0x21,0x11,0x09,0x48,0x20,0x81,0xF8,0x01,0xF8,0x18,0x03,0xEE,0xFF,0x38,0x3E,
0x3F,0x21,0x11,0x09,0x48,0x11,0x00,0x30,0x7C,0xE3,0x3E,0x00,0x21,0xE0,0x1E,0x1C,
0x21,0x21,0x11,0x09,0x48,0x11,0x00,0xC2,0x7C,0x83,0x3E,0x46,0xF7,0xE0,0x1E,0x0C,
0x21,0x21,0xFF,0x0A,0x38,0x0A,0x03,0xFF,0x10,0x1B,0x38,0x0F,0xFF,0x6E,0x78,0x36,
0x21,0x21,0x11,0x0C,0x08,0x04,0x00,0x21,0x7F,0x3B,0xF9,0x3F,0xFE,0x65,0x58,0x26,
0x3F,0x20,0x10,0x08,0x08,0x0A,0x01,0x24,0x67,0x61,0xF9,0xB3,0xCF,0x63,0xDE,0x06,
0x20,0x20,0x10,0x0F,0xF8,0x31,0x82,0x22,0x60,0x9B,0x36,0x06,0x60,0x0F,0x18,0x0E,
0x01,0xC0,0x10,0x08,0x08,0xC0,0x64,0x61,0x61,0xD9,0x76,0x06,0x60,0x07,0x18,0x1E,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x63,0x03,0xC0,0xC8,0x73,0x8F,0xE1,0x88,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x62,0x0B,0x80,0xFD,0xFF,0x87,0xFF,0x8C,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x60,0x1F,0x00,0xF7,0xCD,0x98,0xFF,0x18,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x07,0xF8,0x30,0x70,0xF8,0xC3,0x8C,
0x00,0x10,0x10,0x00,0x78,0x04,0x00,0x20,0x00,0x01,0xF8,0x30,0x79,0xF8,0xC3,0x18,
0x7E,0x10,0x28,0x0F,0x80,0x02,0x00,0x50,0x7F,0xF8,0x37,0x0D,0xCF,0x67,0xDB,0xCE,
0x10,0x90,0x44,0x01,0x00,0xFF,0xE0,0x88,0x7F,0xF8,0x37,0xB7,0xEE,0xEF,0xDB,0xCE,
0x10,0x90,0x82,0x02,0x10,0x20,0x81,0x04,0x60,0x18,0x61,0xF3,0xF0,0xE4,0xC3,0x00,
0x1E,0x93,0x01,0x87,0xE0,0x11,0x06,0x03,0x6F,0xDB,0x19,0xFE,0x6F,0xFD,0xFF,0x38,
0x22,0x90,0xFE,0x00,0xC0,0x0E,0x01,0xFC,0x6F,0xDB,0x19,0xFE,0x6F,0xFB,0x7F,0xBC,
0x52,0x90,0x10,0x03,0x08,0x31,0x80,0x20,0x6F,0xD9,0x8E,0xC1,0x9C,0x93,0x1E,0xC0,
0x0A,0x90,0x10,0x0F,0xFC,0xD1,0x60,0x20,0x6F,0xDB,0x1E,0x49,0x99,0x9B,0x1E,0x42,
0x04,0x91,0xFF,0x00,0x84,0x11,0x03,0xFE,0x6F,0xDB,0x3E,0xCC,0x00,0x7C,0x66,0x36,
0x08,0x10,0x10,0x04,0x90,0x11,0x00,0x20,0x6F,0xD9,0x35,0xBB,0x9D,0xEF,0xE6,0x26,
0x10,0x10,0x10,0x08,0x88,0x21,0x00,0x20,0x60,0x18,0x01,0xB7,0xBF,0xE7,0xF8,0x06,
0x20,0x73,0xFF,0x91,0x84,0x41,0x07,0xFF,0x7F,0xF8,0x30,0xC2,0x0F,0x9F,0x1E,0xFE,

```
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7F,0xF8,0x20,0x46,0x0F,0x9F,0x0E,0xFE,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
};
void   delay(unsigned int m)          //延时程序
            {
              unsigned int i,j;
              for(i=0;i<m;i++)
                for(j=0;j<20;j++);
            }

void   delayms(unsigned int n)          //延时10×n毫秒程序
            {
              unsigned int i,j;
              for(i=0;i<n;i++)
                {
                 for(j=0;j<1000;j++);
                }
            }

/*********************************************************************
* 名称 : TransferData ()
* 功能 : 写指令或数据
*********************************************************************/
void TransferData(uchar data1,bit DI)
{
        rw=0;
        rs=DI;
        //delay(1);
        db=data1;
        e=1;
        delay(1);
        e=0;
            delay(1);
}
void initinal(void)         //LCD字库初始化程序
{
           rst=0;
           delay(10);
           rst=1;
        delayms(5);           //大于40MS的延时程序
      TransferData(0x30,0);  //Extended Function Set :8BIT设置,RE=0: basic //instruction set, G=0
                                :graphic display OFF
        delay(10);          //大于100uS的延时程序
        TransferData(0x30,0);  //Function Set
```

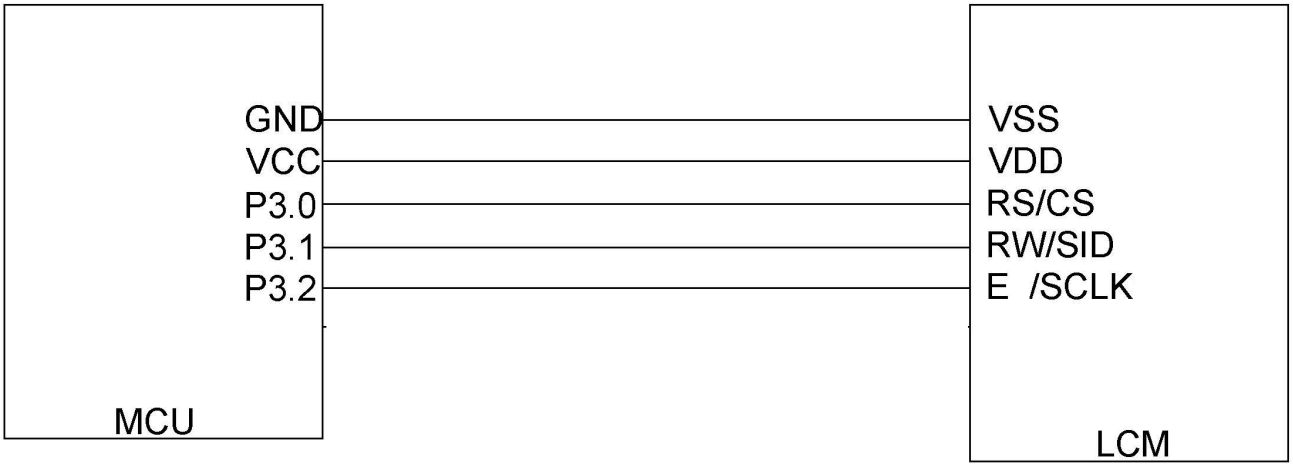
```
        delay(5);           ////大于37uS的延时程序
        TransferData(0x03,0);  //Display on Control
        delay(10);          //大于100uS的延时程序
        TransferData(0x0C,0);  //Display Control,D=1,显示开
        delay(10);          //大于100uS的延时程序
        TransferData(0x01,0);  //Display Clear
        delay(2);           //大于10mS的延时程序
        TransferData(0x06,0);  //Enry Mode Set,光标从右向左加1位移动
        delay(10);          //大于100uS的延时程序
       }
/*********************************************************************
* 名称 : disp_dat(数据1，数据2)
* 功能 : 显示图数据
* 输入 : 数据1，数据2
* 输出 : 无
*********************************************************************/
void disp_dat(unsigned char dat1,unsigned char dat2)
{
  int i,j;
 TransferData(0x36,0);
  for(i=0;i<16;i++)            //
   {
    TransferData((0x80 + (i*2)),0); //SET  垂直地址 VERTICAL ADD
    TransferData(0x80,0);       //SET  水平地址 HORIZONTAL ADD
    for(j=0;j<16;j++)
     {
      TransferData(dat1,1);
     }
    TransferData((0x80 + (i*2)+1),0); //SET  垂直地址 VERTICAL ADD
    TransferData(0x80,0);       //SET  水平地址 HORIZONTAL ADD
                for(j=0;j<16;j++)
     {
                        TransferData(dat2,1);
     }
   }
for(i=0;i<16;i++)            //
   {
    TransferData((0x80 + (i*2)),0); //SET  垂直地址 VERTICAL ADD
    TransferData(0x88,0);       //SET  水平地址 HORIZONTAL ADD
    for(j=0;j<16;j++)
     {
      TransferData(dat1,1);
     }
```

```
    TransferData((0x80 + (i*2)+1),0); //SET  垂直地址 VERTICAL ADD
    TransferData(0x88,0);       //SET  水平地址 HORIZONTAL ADD
                for(j=0;j<16;j++)
     {
                    TransferData(dat2,1);
     }
     }
}

void disp_bmp(uchar code *bmpadd)
{   uchar i,j;
    uint k=0;
  TransferData(0x36,0);
   for(i=0;i<32;i++)            //
    {
     TransferData((0x80 + i),0); //SET  垂直地址 VERTICAL ADD
     TransferData(0x80,0);       //SET  水平地址 HORIZONTAL ADD
     for(j=0;j<16;j++)
      {
       TransferData(bmpadd[k],1);
                    k++;
     }
               }
    for(i=0;i<32;i++)            //
    {
     TransferData((0x80 + i),0); //SET  垂直地址 VERTICAL ADD
     TransferData(0x88,0);       //SET  水平地址 HORIZONTAL ADD
     for(j=0;j<16;j++)
      {
       TransferData(bmpadd[k],1);
                    k++;
     }
               }


     }
/*****************************************************************
* 名称 : LCD_MESG()
* 功能 : 显示文字函数
* 输入 : 无
* 输出 : 无
*****************************************************************/
void   lcd_mesg(unsigned char code *adder1)
{
    unsigned char i;
```

```
    TransferData(0x30,0);
    TransferData(0x30,0);
     TransferData(0x80,0);  //Set Graphic Display RAM Address
    for(i=0;i<32;i++)
      {
            TransferData(*adder1,1);
            adder1++;
      }
    TransferData(0x30,0);
    TransferData(0x30,0);
     TransferData(0x90,0);  //Set Graphic Display RAM Address
    for(i=0;i<32;i++)
      {
            TransferData(*adder1,1);
            adder1++;
      }
}

//////////////////////////////////////////////////////
void bmpclear()
{
uchar i,j;
    TransferData(0x34,0);

    for(i=0;i<32;i++)
    {
    TransferData(0x80+i,0);
    TransferData(0x80,0);
      for(j=0;j<32;j++)
      TransferData(0,1);
    }
}

/*****************************************************************************
* 名称 : Main()
* 功能 : 主函数
* 输入 : 无
* 输出 : 无
*****************************************************************************/
void   main(void)
   {
    initinal();          //调用LCD显示图片(扩展)初始化程序
      while(1)
      {
```

```
        TransferData(0x30,0);
              TransferData(0x01,0);//txtclear();
                    delayms(2);
      disp_dat(0xff,0xff);
      delayms(1000);
      disp_dat(0xaa,0xaa); //Seperate Rows I
      delayms(1000);
      disp_dat(0xff,0x00); //Seperate Columns II
      delayms(1000);
      disp_dat(0xaa,0x55); //Seperate Dots I
      delayms(1000);
      disp_dat(0x55,0xaa); //Seperate Dots I
      delayms(1000);
      //txtclear();
      bmpclear();
  lcd_mesg(IC_DAT1);     //显示西文
  delayms(1200);
      TransferData(0x30,0);
              TransferData(0x01,0);//txtclear();
                    delayms(2);
  disp_bmp(bmp1);
  delayms(1200);
      bmpclear();
      TransferData(0x34,0);
  TransferData(0x04,0);
  lcd_mesg(IC_DAT2);    //显示中文汉字
  delayms(1200);

      TransferData(0x34,0);
  TransferData(0x04,0);
    }
  }
```

9.3 串口接线图



串口接线方法

9.4 串口C51例程

```
#include <STC15.H>
#include <string.h>
#include <INTRINS.H>
#define uchar unsigned char
#define uint  unsigned int
#define db  P1
sbit rs=P3^0;
sbit rw=P3^1;
sbit e=P3^2;
sbit rst=P3^3;
uchar code IC_DAT1[]={
"0123456789:;<=>?"
"PQRSTUVWXYZ[/]^_"                    //0x80
"@ABCDEFGHIJKLMNO"
"'abcdefghijklmno"   };
uchar code IC_DAT2[]={
"湖南旭阳显示科技"                    //0x88
"深圳市耀兴阳实业"                    //0x98
"有限公司湖南芷江"
"有限公司专业模组"};
////////////////////////////////////////////////////////////////////////////
```

```
// Bitmap点阵数据表                                    //
// 图片: D:\..   12864.bmp,横向取模左高位,数据排列:从左到右从上到下   //
// 图片尺寸: 128 * 64                                  //
//////////////////////////////////////////////////////////////////////
unsigned char code nBitmapDot[] =              // 数据表
{
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7F,0xFB,0xC0,0xCF,0xF3,0xE0,0xDF,0xFE,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7F,0xFB,0xC0,0xCF,0xF3,0xE0,0xDF,0xFE,
    0x00,0x01,0x20,0x84,0x6C,0x08,0x40,0x00,0x60,0x1B,0xE1,0xF7,0x8F,0x07,0xD8,0x06,
    0x5F,0xF1,0x24,0x84,0xB4,0x44,0x47,0xBF,0x6F,0xDB,0xFF,0xF3,0x8F,0x0F,0xDB,0xF6,
    0x32,0x91,0x24,0x95,0x6C,0x24,0x84,0xA1,0x6F,0xD9,0xFE,0x3D,0x8C,0x04,0x1B,0xF6,
    0x04,0x43,0xA4,0x8E,0xB4,0x20,0x84,0xA1,0x6F,0xDB,0xFE,0xBD,0xC8,0xBE,0x1B,0xF6,
    0x09,0x21,0x24,0x84,0x48,0x01,0x05,0x21,0x6F,0xD8,0xFF,0xFF,0xE1,0xFF,0x1B,0xF6,
    0x41,0x01,0x24,0x9F,0x7C,0x00,0x05,0x3F,0x6F,0xD8,0x71,0xE1,0x80,0x8E,0x1B,0xF6,
    0x2F,0xF1,0x24,0x8A,0xD0,0xFF,0xE4,0xA1,0x6F,0xD8,0x31,0xC1,0x80,0x04,0x1B,0xF6,
    0x01,0x01,0x24,0x8A,0x7C,0x00,0x04,0xA1,0x60,0x1B,0x1E,0x01,0xF0,0x7B,0xD8,0x06,
    0x03,0x81,0xA4,0x8A,0x50,0x11,0x04,0xA1,0x7F,0xFB,0x0E,0x49,0xFA,0x77,0xDF,0xFE,
    0x25,0x43,0x24,0x8A,0x7C,0x20,0x87,0x21,0x7F,0xFB,0x36,0x4D,0x93,0x64,0xDF,0xFE,
    0x59,0x30,0x44,0x8B,0x50,0x40,0x44,0x3F,0x00,0x03,0x9E,0xF2,0x6E,0x60,0xC0,0x00,
    0x01,0x00,0x80,0x92,0x7C,0x80,0x24,0x21,0x00,0x03,0x0E,0xF6,0x6F,0x60,0xC0,0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0xDB,0xD9,0xFD,0x80,0x7C,0x3E,0x3E,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x6F,0xD9,0xC9,0xFF,0x81,0x7F,0x7E,0xFE,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x60,0x60,0xC0,0x7E,0x33,0x99,0xDE,0xF6,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0x19,0x99,0xE9,0x93,0x1F,0x1E,0x76,
    0x02,0x00,0x44,0x01,0x00,0xF7,0xC0,0x88,0x0F,0x1B,0x09,0xCD,0xBB,0x1F,0x1C,0xFE,
    0x7F,0xF0,0x44,0x01,0x00,0x94,0x40,0x88,0x7F,0x60,0xF6,0x00,0x4F,0x07,0xDF,0xCE,
    0x40,0x10,0x44,0x1F,0xFC,0x94,0x40,0x88,0x7F,0x61,0xE6,0x10,0xEF,0x1F,0xDF,0xFE,
    0x09,0x00,0x44,0x02,0x00,0xA7,0xC1,0x04,0x73,0x9B,0x01,0xB1,0xF0,0xDC,0xF8,0x38,
    0x05,0x02,0x44,0x87,0xF8,0xA4,0x41,0x24,0x03,0xF1,0x87,0x3D,0xF1,0x9C,0x74,0x30,
    0x11,0x01,0x45,0x0C,0x08,0x94,0x42,0x22,0x83,0xE3,0xC6,0x08,0x81,0x9C,0x26,0xF0,
    0x09,0x00,0xC6,0x17,0xF8,0x97,0xC4,0x41,0x6F,0xDF,0xC9,0x80,0x00,0x9F,0x1B,0xCE,
    0x7F,0xF0,0x44,0x04,0x08,0x95,0x20,0x40,0x6F,0x9F,0xD9,0x25,0x10,0xDF,0x19,0xDE,
    0x02,0x00,0x44,0x07,0xF8,0x95,0x40,0x88,0x70,0x00,0xC0,0x12,0x96,0xE4,0x78,0x08,
    0x04,0xC0,0x44,0x04,0x08,0xE4,0x81,0x04,0x50,0x30,0xC6,0x00,0x00,0x7F,0x39,0x3E,
    0x18,0x20,0x44,0x04,0x08,0x86,0x43,0xFE,0x00,0x78,0xC6,0x35,0x60,0x7F,0x7B,0x3E,
    0x60,0x13,0xFF,0x84,0x18,0x84,0x20,0x02,0x1F,0xC4,0x06,0x08,0x10,0xC0,0xE6,0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x3F,0x8E,0x06,0x00,0x0C,0x80,0xF6,0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x6C,0xDE,0x20,0x00,0x80,0xDF,0x18,0x1E,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x63,0x63,0xF0,0x00,0xC0,0x7D,0x01,0x0E,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x63,0x23,0xCC,0x09,0x40,0x60,0xC3,0x86,
    0x00,0x00,0x10,0x1F,0xFC,0x08,0x00,0x1E,0x78,0x78,0xDF,0x00,0x80,0xE6,0x7F,0xF8,
    0x3F,0xE0,0x10,0x01,0x40,0x04,0x03,0xE0,0x3C,0x78,0xDF,0x05,0xE2,0xF4,0x7F,0xFC,
    0x00,0x20,0x10,0x01,0x40,0xFF,0xE0,0x40,0x04,0xE0,0xC3,0x01,0xF2,0x78,0x26,0xFE,
    0x7F,0xA1,0xFF,0x0F,0xF8,0x20,0x80,0x84,0x0C,0xF0,0x08,0x03,0xF6,0x7F,0x74,0x7E,
    0x00,0x21,0x11,0x09,0x48,0x20,0x81,0xF8,0x01,0xF8,0x18,0x03,0xEE,0xFF,0x38,0x3E,
```

```
    0x3F,0x21,0x11,0x09,0x48,0x11,0x00,0x30,0x7C,0xE3,0x3E,0x00,0x21,0xE0,0x1E,0x1C,
    0x21,0x21,0x11,0x09,0x48,0x11,0x00,0xC2,0x7C,0x83,0x3E,0x46,0xF7,0xE0,0x1E,0x0C,
    0x21,0x21,0xFF,0x0A,0x38,0x0A,0x03,0xFF,0x10,0x1B,0x38,0x0F,0xFF,0x6E,0x78,0x36,
    0x21,0x21,0x11,0x0C,0x08,0x04,0x00,0x21,0x7F,0x3B,0xF9,0x3F,0xFE,0x65,0x58,0x26,
    0x3F,0x20,0x10,0x08,0x08,0x0A,0x01,0x24,0x67,0x61,0xF9,0xB3,0xCF,0x63,0xDE,0x06,
    0x20,0x20,0x10,0x0F,0xF8,0x31,0x82,0x22,0x60,0x9B,0x36,0x06,0x60,0x0F,0x18,0x0E,
    0x01,0xC0,0x10,0x08,0x08,0xC0,0x64,0x61,0x61,0xD9,0x76,0x06,0x60,0x07,0x18,0x1E,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x63,0x03,0xC0,0xC8,0x73,0x8F,0xE1,0x88,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x62,0x0B,0x80,0xFD,0xFF,0x87,0xFF,0x8C,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x60,0x1F,0x00,0xF7,0xCD,0x98,0xFF,0x18,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x07,0xF8,0x30,0x70,0xF8,0xC3,0x8C,
    0x00,0x10,0x10,0x00,0x78,0x04,0x00,0x20,0x00,0x01,0xF8,0x30,0x79,0xF8,0xC3,0x18,
    0x7E,0x10,0x28,0x0F,0x80,0x02,0x00,0x50,0x7F,0xF8,0x37,0x0D,0xCF,0x67,0xDB,0xCE,
    0x10,0x90,0x44,0x01,0x00,0xFF,0xE0,0x88,0x7F,0xF8,0x37,0xB7,0xEE,0xEF,0xDB,0xCE,
    0x10,0x90,0x82,0x02,0x10,0x20,0x81,0x04,0x60,0x18,0x61,0xF3,0xF0,0xE4,0xC3,0x00,
    0x1E,0x93,0x01,0x87,0xE0,0x11,0x06,0x03,0x6F,0xDB,0x19,0xFE,0x6F,0xFD,0xFF,0x38,
    0x22,0x90,0xFE,0x00,0xC0,0x0E,0x01,0xFC,0x6F,0xDB,0x19,0xFE,0x6F,0xFB,0x7F,0xBC,
    0x52,0x90,0x10,0x03,0x08,0x31,0x80,0x20,0x6F,0xD9,0x8E,0xC1,0x9C,0x93,0x1E,0xC0,
    0x0A,0x90,0x10,0x0F,0xFC,0xD1,0x60,0x20,0x6F,0xDB,0x1E,0x49,0x99,0x9B,0x1E,0x42,
    0x04,0x91,0xFF,0x00,0x84,0x11,0x03,0xFE,0x6F,0xDB,0x3E,0xCC,0x00,0x7C,0x66,0x36,
    0x08,0x10,0x10,0x04,0x90,0x11,0x00,0x20,0x6F,0xD9,0x35,0xBB,0x9D,0xEF,0xE6,0x26,
    0x10,0x10,0x10,0x08,0x88,0x21,0x00,0x20,0x60,0x18,0x01,0xB7,0xBF,0xE7,0xF8,0x06,
    0x20,0x73,0xFF,0x91,0x84,0x41,0x07,0xFF,0x7F,0xF8,0x30,0xC2,0x0F,0x9F,0x1E,0xFE,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7F,0xF8,0x20,0x46,0x0F,0x9F,0x0E,0xFE,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
};
void   delay(unsigned int m)          //延时程序
            {
              unsigned int i,j;
              for(i=0;i<m;i++)
                for(j=0;j<20;j++);
            }
void   delayms(unsigned int n)          //延时10×n毫秒程序
            {
              unsigned int i,j;
              for(i=0;i<n;i++)
                                                                    {
                for(j=0;j<1000;j++);
                                                                    }
            }

/*****************************************************************
* 名称 : sendbyte()
* 功能 : 按照液晶的串口通信协议，发送数据
* 输入 : zdata
* 输出 : 无
*****************************************************************/
void sendbyte(unsigned char zdata)
{
```

```
        unsigned int i;
        e = 0;
        for(i=0; i<8; i++)
        {rw=(bit)(zdata & 0x80);
                zdata=zdata<<1;
                e = 0;
                e = 1;
        }
        }

/*********************************************************************
* 名称 : TransferData ()
* 功能 : 写串口指令或数据
* 输入 : data1,DI
* 输出 : 无
*********************************************************************/
void TransferData(uchar data1,bit DI)
{
        if(DI==0)sendbyte(0xf8);
        else sendbyte(0xfa);
        sendbyte(data1 & 0xf0);
        sendbyte((data1 << 4) & 0xf0);
}

void initinal(void)          //LCD字库初始化程序
{
                rst=0;
                delay(10);
                rst=1;
         delayms(5);            //大于40MS的延时程序
         TransferData(0x30,0);  //Extended Function Set :8BIT设置,
                                  //RE=0: basic instruction set, G=0 :graphic display OFF
         delay(10);           //大于100uS的延时程序
         TransferData(0x30,0);  //Function Set
         delay(5);            ////大于37uS的延时程序
         TransferData(0x03,0);  //Display on Control
         delay(10);           //大于100uS的延时程序
         TransferData(0x0C,0);  //Display Control,D=1,显示开
         delay(10);           //大于100uS的延时程序
         TransferData(0x01,0);  //Display Clear
         delay(2);            //大于10mS的延时程序
         TransferData(0x06,0);  //Enry Mode Set,光标从右向左加1位移动
         delay(10);           //大于100uS的延时程序
        }
/*********************************************************************
* 名称 : disp_dat(数据1，数据2)
* 功能 : 显示图数据
* 输入 : 数据1，数据2
* 输出 : 无
*********************************************************************/
void disp_dat(unsigned char dat1,unsigned char dat2)
```

```
{
  int i,j;
 TransferData(0x36,0);
  for(i=0;i<16;i++)            //
   {
    TransferData((0x80 + (i*2)),0); //SET  垂直地址 VERTICAL ADD
    TransferData(0x80,0);       //SET  水平地址 HORIZONTAL ADD
    for(j=0;j<16;j++)
     {
      TransferData(dat1,1);
     }
       TransferData((0x80 + (i*2)+1),0); //SET  垂直地址 VERTICAL ADD
    TransferData(0x80,0);       //SET  水平地址 HORIZONTAL ADD
                        for(j=0;j<16;j++)
     {
                                TransferData(dat2,1);
     }
   }
for(i=0;i<16;i++)            //
   {
    TransferData((0x80 + (i*2)),0); //SET  垂直地址 VERTICAL ADD
    TransferData(0x88,0);       //SET  水平地址 HORIZONTAL ADD
    for(j=0;j<16;j++)
     {
      TransferData(dat1,1);
     }
       TransferData((0x80 + (i*2)+1),0); //SET  垂直地址 VERTICAL ADD
    TransferData(0x88,0);       //SET  水平地址 HORIZONTAL ADD
                        for(j=0;j<16;j++)
     {
                                TransferData(dat2,1);
     }
      }
}
//////////////////////////////////////////////////////////
void disp_bmp(uchar code *bmpadd)
{   uchar i,j;
       uint k=0;

 TransferData(0x36,0);

  for(i=0;i<32;i++)            //
   {
    TransferData((0x80 + i),0); //SET  垂直地址 VERTICAL ADD
    TransferData(0x80,0);       //SET  水平地址 HORIZONTAL ADD
    for(j=0;j<16;j++)
     {
      TransferData(bmpadd[k],1);
                                k++;
     }
                        }
for(i=0;i<32;i++)            //
   {
```

```
    TransferData((0x80 + i),0); //SET  垂直地址 VERTICAL ADD
    TransferData(0x88,0);       //SET  水平地址 HORIZONTAL ADD
    for(j=0;j<16;j++)
     {
      TransferData(bmpadd[k],1);
                                k++;
     }
                        }

        }
/*********************************************************************
* 名称 : LCD_MESG()
* 功能 : 显示文字函数
*********************************************************************/
void   lcd_mesg(unsigned char code *adder1)
{
       unsigned char i;
       TransferData(0x30,0);
       TransferData(0x30,0);
      TransferData(0x80,0);  //Set Graphic Display RAM Address

       for(i=0;i<32;i++)
               {
                       TransferData(*adder1,1);
                       adder1++;
               }

       TransferData(0x30,0);
       TransferData(0x30,0);
      TransferData(0x90,0);  //Set Graphic Display RAM Address

       for(i=0;i<32;i++)
               {
                       TransferData(*adder1,1);
                       adder1++;
               }
}
//////////////////////////////////////////////////////
void bmpclear()
{
uchar i,j;
       TransferData(0x34,0);

       for(i=0;i<32;i++)
       {
       TransferData(0x80+i,0);
       TransferData(0x80,0);
               for(j=0;j<32;j++)
               TransferData(0,1);
       }
}

/*********************************************************************
```

```
* 名称 : Main()
* 功能 : 主函数
*********************************************************************/
void    main(void)
    {
     initinal();          //调用LCD显示图片(扩展)初始化程序
       while(1)
          {
       TransferData(0x30,0);
       TransferData(0x01,0);//txtclear();
       delayms(2);
       disp_dat(0xff,0xff);
       delayms(1000);
       disp_dat(0xaa,0xaa); //Seperate Rows I
       delayms(1000);
       disp_dat(0xff,0x00); //Seperate Columns II
       delayms(1000);
       disp_dat(0xaa,0x55); //Seperate Dots I
       delayms(1000);
       disp_dat(0x55,0xaa); //Seperate Dots I
       delayms(1000);
       //txtclear();
       bmpclear();
   lcd_mesg(IC_DAT1);     //显示西文
   delayms(1200);
       TransferData(0x30,0);
       TransferData(0x01,0);//txtclear();
       delayms(2);
   disp_bmp(bmp1);
   delayms(1200);
       bmpclear();
       TransferData(0x34,0);
   TransferData(0x04,0);
   lcd_mesg(IC_DAT2);    //显示中文汉字
   delayms(1200);
   TransferData(0x34,0);
   TransferData(0x04,0);
      }
   }
```

10.注意事项

1.液晶显示器（LCD）

液晶显示器是由玻璃，有机密封胶，有机流体，和聚合物基偏光片。搬运时应注意以下事项：，

（1） .保持温度在使用和储存范围内。过高的温度和湿度会导致偏振退化、偏振器剥离或气泡。

（2） .不要用比HB铅笔芯更硬的东西接触暴露的偏光镜。清除显示器表面的灰尘，用棉花轻轻擦拭，鹿皮巾或其他软材料浸泡在清洁油中。

（3） 立即擦掉唾液或水滴。ITO与水接触时间过长，会导致液晶显示器表面变形或变色。

（4） 玻璃很容易因粗暴的操作而碎裂。尤其是在角落和边缘。

（5） .不要用直流电压驱动液晶显示器。

2.液晶显示模块

2.1机械注意事项

LCM的装配和调整具有高精度。避免过度震动，不要做任何改动或修改。应注意以下几点。

（1） .不要以任何方式改变金属框架上的凸耳。

（2） .请勿通过钻额外的孔、更改其轮廓、移动其组件或修改其图案来修改PCB。

（3） .请勿触摸弹性体连接器，尤其是插入背光面板（例如，EL）。

（4） .安装LCM时，请确保PCB板不受任何压力，如弯曲或扭曲。弹性体接触非常精细，任何元素的轻微错位都可能导致像素缺失。

（5） .避免压在金属挡板上，否则弹性体连接器可能会变形和失去接触，从而导致丢失像素。

2.2.静电

LCM包含CMOS LSI，对此类设备应采取相同的预防措施，即

（1） .当操作员与模块接触时，应将其接地。切勿用人体任何部位接触任何导电部件，如LSI焊盘、PCB上的铜导线和接口端子。

（2） .模块应保存在防静电袋或其他防静电容器中储存。

（3） .只能使用正确接地的烙铁。

（4） .如果使用电动螺丝刀，应良好接地，并防止换向器火花。

（5） .工作服和工作台应遵守正常的防静电措施；对于后者，建议使用导电（橡胶）垫。

（6） 。由于干燥空气会感应静电，建议相对湿度为50-60%。

2.3.焊接

（1） .仅焊接至I/O端子。

（2） .只能使用接地正确且无漏电的烙铁。

（3） .焊接温度：280℃±10℃

（4） .焊接时间：3到4秒。

（5） .使用树脂助焊剂填充的低温焊锡。

（6） .如果使用助焊剂，应覆盖LCD表面，以避免焊剂飞溅。助焊剂残留物应在防护后清除。

2.4.操作

（1） 观察角度可以通过改变LCD驱动电压V0来调节。

（2） 驱动电压应保持在规定的范围内，过高的电压会缩短显示器的寿命。（3） 响应时间随着温度的降低而增加。

（4） .在高于其工作范围的温度下，显示器可能会变成黑色或深蓝色；这（但是不要按压显示区域）可能会导致部分显示线段“断裂”。

（5） .操作过程中的机械损害（如按压显示区域）可能会导致线段出现“断裂”。

2.5.储存

如果有液体从损坏的玻璃电池中漏出，将任何接触的人体部分用肥皂和水冲洗干净。切勿吞下液体。毒性极低，但应始终小心。

2.6.有限保修

除非与客户另有约定，从装运日期起一年内，当根据验收标准检查时发现其电气和外观缺陷，将维修或修理其任何LCD和IC，该日期的确认应以货运单据为依据，保修责任仅限于根据上述条款进行的维修和/或更换。不会对任何后续或后果性事件负责。