

# JLX1602A-4 使用说明书

## 目 录

序号	内 容 标 题	页码
1	概述	2
2	字符型模块的特点	2
3	外形及接口引脚功能	3-6
4	基本原理	7-10
5	技术参数	10
6	时序特性	11-13
7	指令功能	13-末页

## 1. 概述

晶联讯电子专注于液晶屏及液晶模块的研发、制造。所生产 JLX1602A 型液晶模块由于使用方便、显示清晰，广泛应用于各种人机交流面板。

JLX1602A 可以显示 2 行，每行 16 个英文、数字、符号，并可选择日文、俄文、以色列文、希腊文等文字（注俄文、以色列文、希腊文需订制）。并可以自编字符（每显示一个界面最多可以达到 8 个 5\*8 点阵或 4 个 5\*11 自编字符）。

## 2. 字符型模块的性能

重量轻： $\leq 30\text{g}$ ；

体积小： $\leq 11\text{mm}$  厚；

功耗低： $10 - 100\text{mW}$ （不带背光  $10\text{ mW}$ ，带背光不大于  $100\text{ mW}$ ）；

显示内容： $192$  种字符（ $5 \times 8$  点字型）；

$32$  种字符（ $5 \times 11$  点字型）；

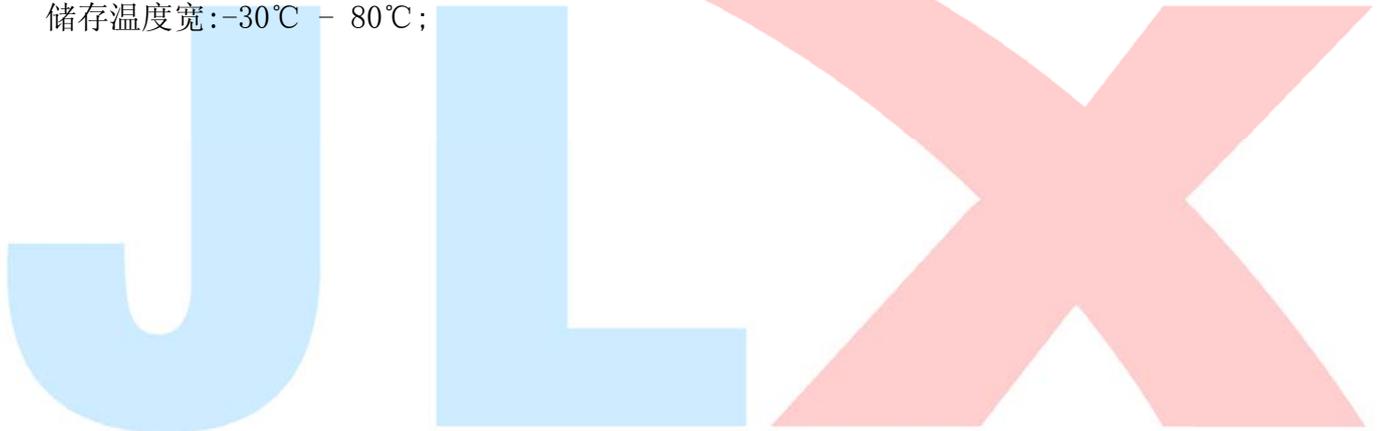
可自编 8 种（ $5 \times 8$ ）或 4 种（ $5 \times 11$ ）种字符，（注每显示一个界面最多可以达到 8 个自编字符，但更换显示界面后可再编）；

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；

接口简单方便：可选择 4 位并行接口、8 位并行接口、4 线 SPI 串行接口、IIC 接口（ $I^2C$  接口）。

工作温度宽： $-20^\circ\text{C} - 70^\circ\text{C}$ ；

储存温度宽： $-30^\circ\text{C} - 80^\circ\text{C}$ ；



3. 外形尺寸及接口引脚功能

Dimensions (mm):

- 80±0.3 (PCB)
- 75.4±0.3 (HOLE)
- 72±0.3 (BEZEL)
- 64±0.3 (V.A)
- 55.7±0.3 (A.A)
- P2.54\*15=38.1
- 16
- 12.15±0.3
- 8±0.3
- 4±0.3
- 2.3±0.3
- 2.5±0.2
- 16±0.3 (V.A)
- 11±0.3 (A.A)
- 9.65±0.3
- 5±0.3
- 25.3±0.3 (BEZEL)
- 31±0.3 (HOLE)
- 36±0.3 (PCB)
- 10.3±0.3
- 1.6±0.3
- 6.0±0.3
- 0.60
- 0.05
- 0.62
- 0.05
- 0.54
- 2.90
- 5.15

Pin No | Symbol | Level | Function

1	VSS	—	GND.
2	VDD	—	Power Supply(+5V).
3	V0	—	Operating Voltage for LCD.
4	RS	H/L	H:data; L: Instruction code
5	R/W	H/L	H:Read; L:Write(parallel)/data(Serial).
6	E	H/L	Upper Screen: Enable; Signal(parallel) serial Clock.
7-14	DB0-DB7	H/L	Data Bus.
15	LED+	—	Power supply for LED backlight(+5v).
16	LED-	—	Power Supply for LED backlight(0V).

ABSOLUTE MAXIMUM RATING

Item	Symbol	Min	Type	Max	Unit
Operating Voltage	VDD	4.8	5.0	5.2	v
Operating Current	IDD	4.8	5.2	5.5	mA
LED Voltage	VLED	4.8	5.0	5.2	v
LED Current (Side LED)	ILED	72	5.2	80	mA

1.DISPLAY TYPE: STN/TRANS YELLOW GREEN(BLUE)  
 2.VIEWING DIRECTION: 60°  
 3.DISPLAY MODE: TRANS/BLUE  
 4.DRIVING METHOD: 1/16DUTY 1/5BIAS  
 5.OPERATING VOLTAGE(VDD-V0): 1.1V  
 6.OPERATING TEMPERATURE: -20° - 70°C  
 7.STORAGE TEMPERATURE: -30°-80°C  
 8.CONNECTOR: ZEBRA  
 9.BACKLIGHT LED: WHITE

External VOP adjust circuit

1)V<sub>R</sub>: 10K-20K  
 2)V<sub>DD</sub>=5V, T<sub>amb</sub>=25°C

Block Diagram:

REVISION RECORD

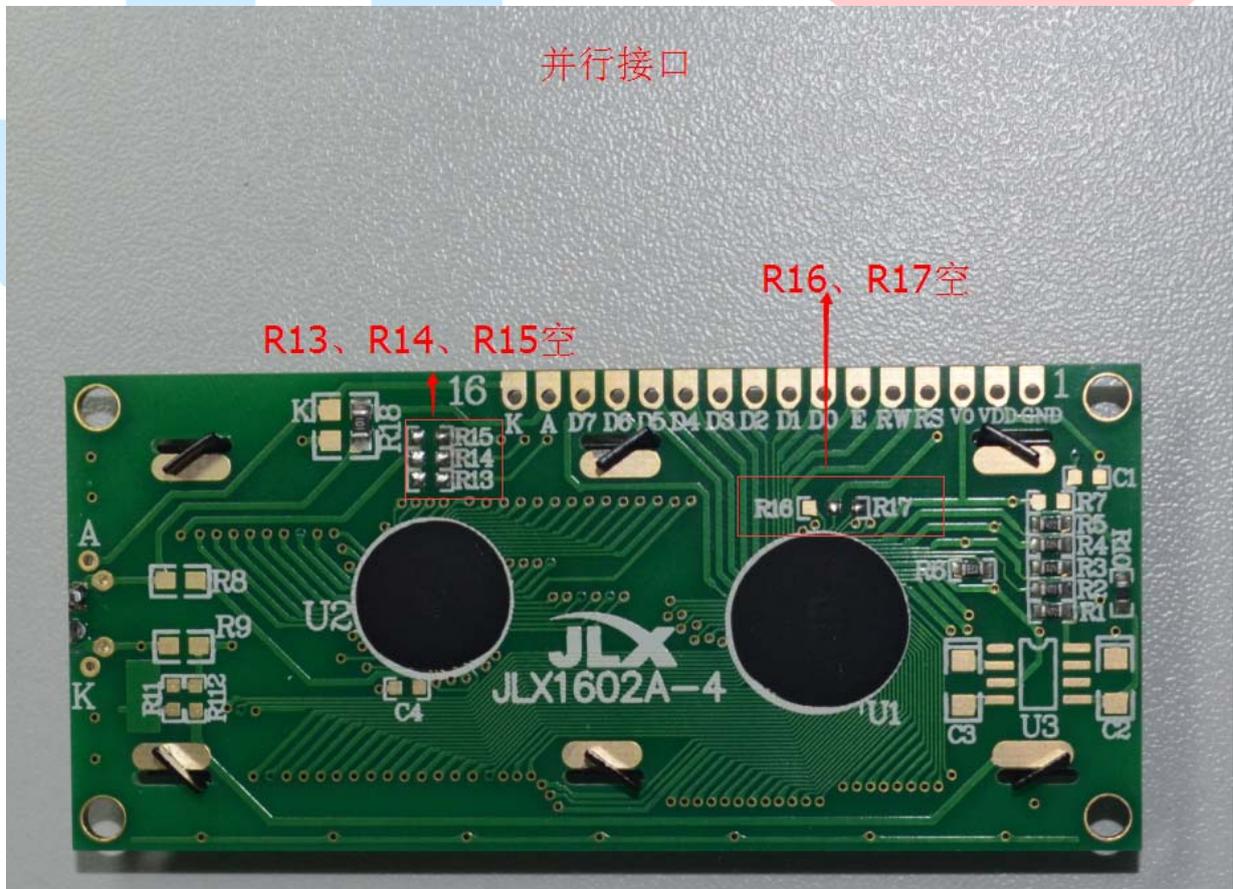
DATE	DATE	DATE	DATE	DATE	DATE
APPROVED	APPROVED	APPROVED	APPROVED	APPROVED	APPROVED
Model No: JLX1602A-4					
Part No: LCM					
DATE: 2015.04.16					
CHECKED	CHECKED	CHECKED	CHECKED	CHECKED	CHECKED
UNIT: mm					
SCALE: 1/1					

GEN.TOL:±0.2  
 PEG(3)  
 VER: A  
 SHEET: 1/1

JLX 深圳市晶联讯电子有限公司  
 Http://www.jlxlcd.cn

在并行接口时的引脚功能：

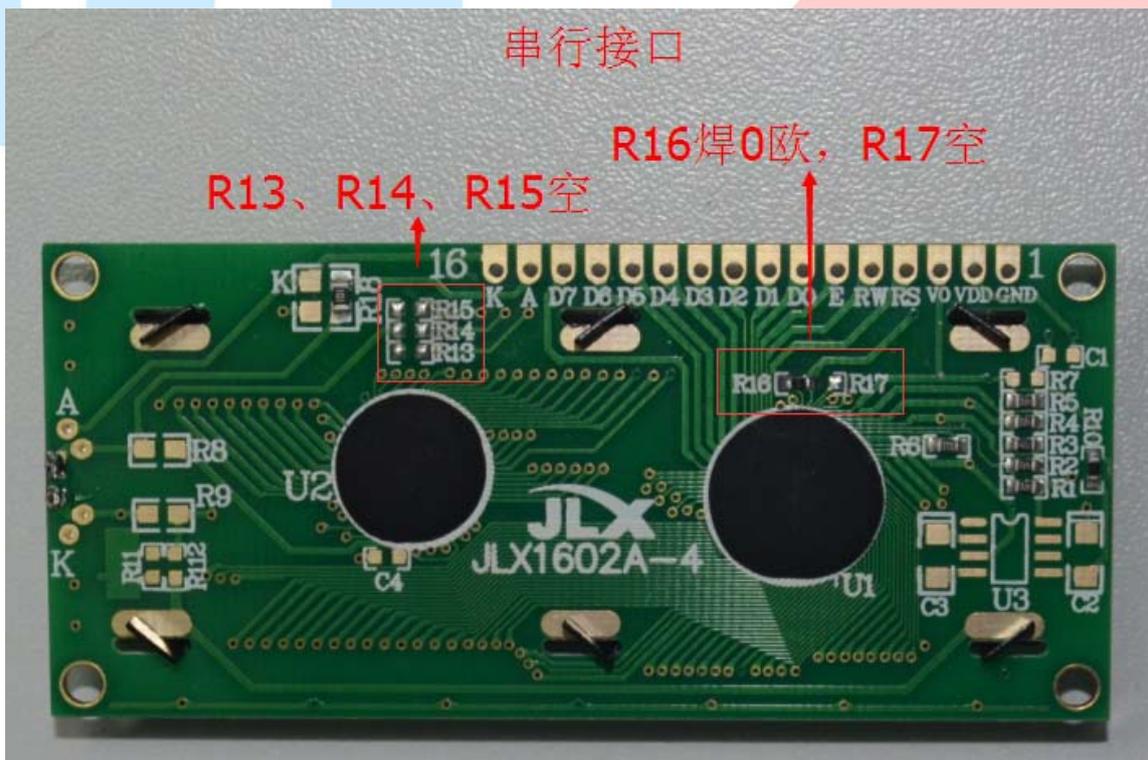
引脚号	符号	功能
1	Vss	接地：0V
2	VDD	供电电源：5V
3	V0	液晶驱动电压： $VDD-V0=1.1V\pm 0.3V$
4	RS	寄存器选择信号：H:数据寄存器， L:指令寄存器
5	R/W	读/写信号：H:读， L:写
6	E	使能信号：下降沿触发，锁存数据
7	DB0	当8位并行接口时：数据线低4位：DB0-DB3 当4位并行接口时：空
8	DB1	
9	DB2	
10	DB3	
11	DB4	当8位并行接口时：数据线高4位：DB4-DB7 当4位并行接口时：数据线高、低4位：DB4-DB7
12	DB5	
13	DB6	
14	DB7	
15	A	背光电源：正极
16	K	背光电源：负极



并行时接电阻方法：R16 和 R17：空；R13\R14\R15 全都空

在串行接口时的引脚功能：

引脚号	符号	功能
1	Vss	接地：0V
2	VDD	供电电源：5V
3	V0	液晶驱动电压：VDD-V0=4.5V±0.3V
4	RS	寄存器选择信号：H:数据寄存器 L:指令寄存器
5	NC	空
6	NC	空
7	NC	空
8	NC	空
9	NC	空
10	NC	空
11	NC	空
12	CS	片选
13	SCLK	串行时钟输入
14	SID	串行数据输入
15	A	背光电源：正极
16	K	背光电源：负极



4 线 SPI 串行时接电阻方法 (R16 接 0 欧 R17 空; R13\R14\R14 全都空)

在 IIC 接口时的引脚功能：

引脚号	符号	功能
1	Vss	接地：0V
2	VDD	供电电源：5V
3	V0	液晶驱动电压： $VDD-V0=4.5V \pm 0.3V$
4	NC	空
5	NC	空
6	NC	空
7	NC	空
8	NC	空
9	NC	空
10	NC	空
11	NC	空
12	NC	空
13	SDA	串行数据输入
14	SCL	串行时钟输入
15	A	背光电源：正极
16	K	背光电源：负极

表 1：模块的接口引脚功能

I2C总线时：



IIC 总线时 PCB 的电阻接线方法（R16 空 R17 接 0 欧；R13\R14\R15 全都接 0 欧）

4. 基本原理

4.1 液晶屏 (LCD)

在液晶板上排列着若干5×7或5×10点阵的字符显示位, 每个显示位可显示1个字符, 本产品每行16个显示位, 共两行。若要每行8、20、24、40位, 1行、2行或4行请选用本厂的: 0802, 1601, 1604, 2002, 2004, 4004等产品。

4.2 工作电图:

图1是 JLX1602A 字符型模块的电路框图, 它由 KS0066, KS0065 及几个电阻电容组成。KS0065 是扩展显示字符用的(例如: 16 字符×1 行模块就不用 KS0065, 16 字符×2 行模块就要用 1 片 KS0065)。

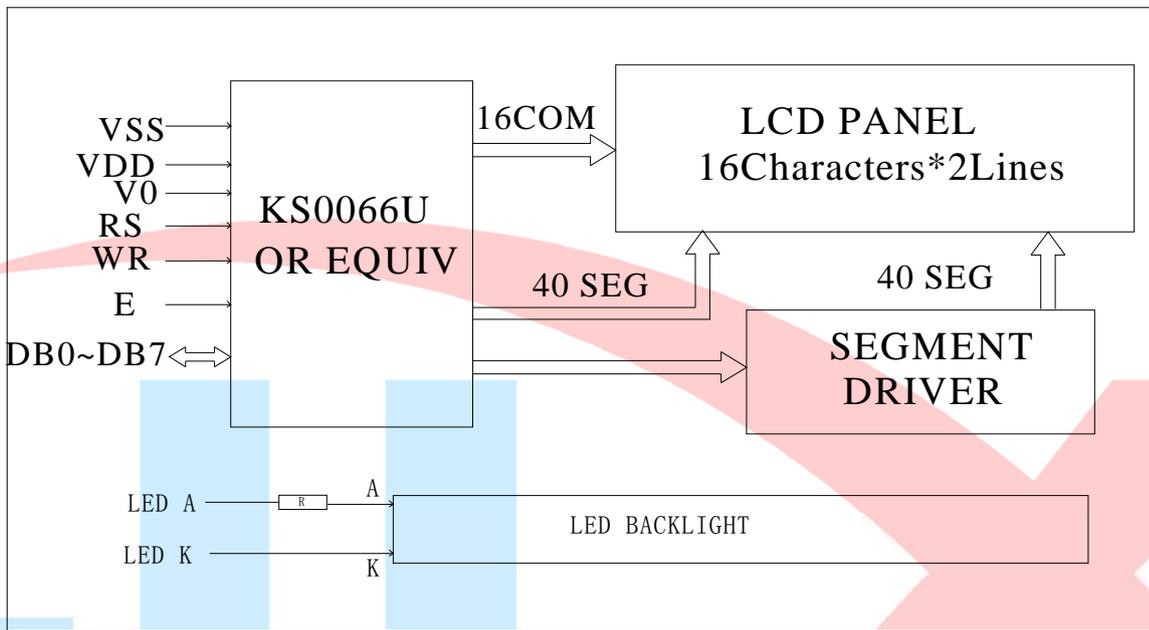


图 1: JLX1602A 字符型模块并行电路框图

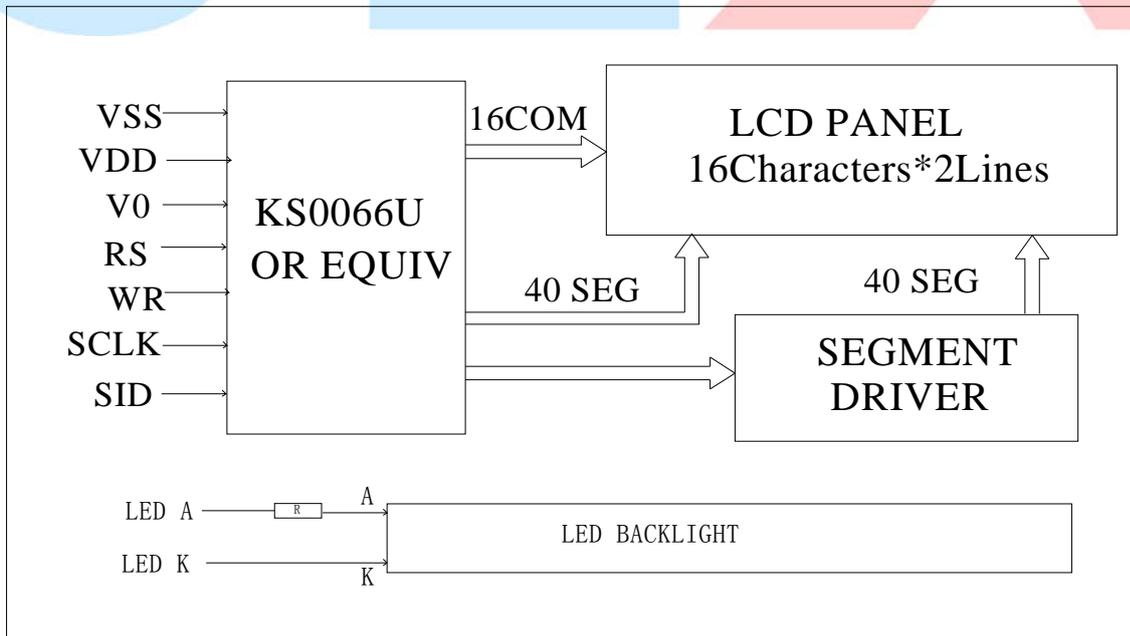


图 1: JLX1602A 字符型模块串行电路框图

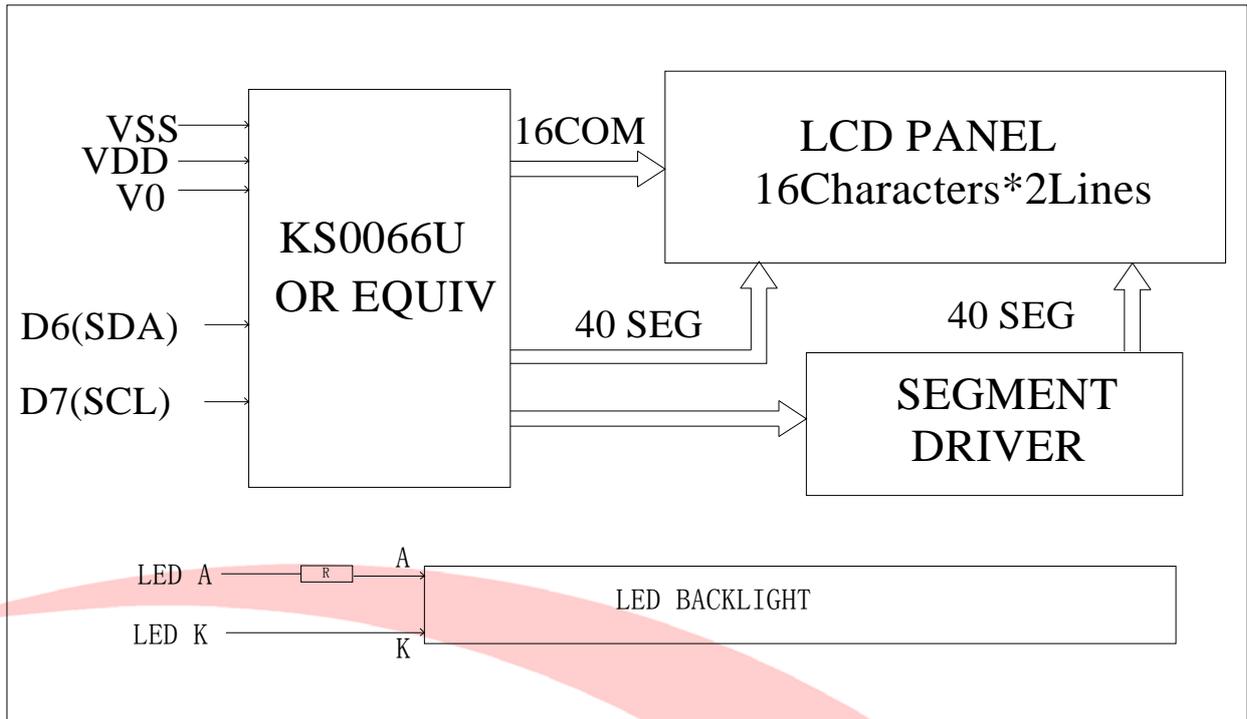


图 1: JLX1602A 字符型模块 IIC 电路框图

接口方面，并行有 8 条数据线，三条控制线；串行有 4 条控制\数据线；IIC 有 2 条控制\数据线；可与微处理器或微控制器相连，通过送入数据和指令，就可使模块正常工作。

#### 4. 3 LCD 驱动器和控制器(LCD driver and controller): KS0066i

见图 2，KS0066i 是用低功耗 CMOS 技术制造的大规模点阵 LCD 控制器(兼带驱动器)，和 4Bit/8Bit 微处理器相连，它能使点阵 LCD 显示大小英文字母，数字和符号。应用 KS0066，用户能用少量元件可组成一个完整点阵 LCD 系统。

特性：

- a. 容易和 4Bit/8Bit MPU 相连；
- b. 可选择 5×7 或 5×10 点阵字符；
- c. 显示数据 RAM 容量:80×8Bit (80 字符)；
- d. 字符发生器 ROM 能提供用户所需字符库或标准库；  
字库容量:192 个字符(5×7 点字型)；  
32 个字符(5×10 点字型)；
- e. DDRAM 和 CGRAM 都能从 MPU 读取数据；(DDRAM 为显示缓冲区；CGRAM 为可自编数据区)
- f. 输出信号:16 个行扫描信号(common signa0)，  
40 个列扫描信号(segment signa0)，本产品通过增加 KS0065 扩展至 80 个列扫描

数量。

- g. 电源复位电路；
- h. 显示占空比：

1/16duty(2 0ine, 5×7dots + Cursor);

- i. 振荡电路;
- j. 指令:11种;

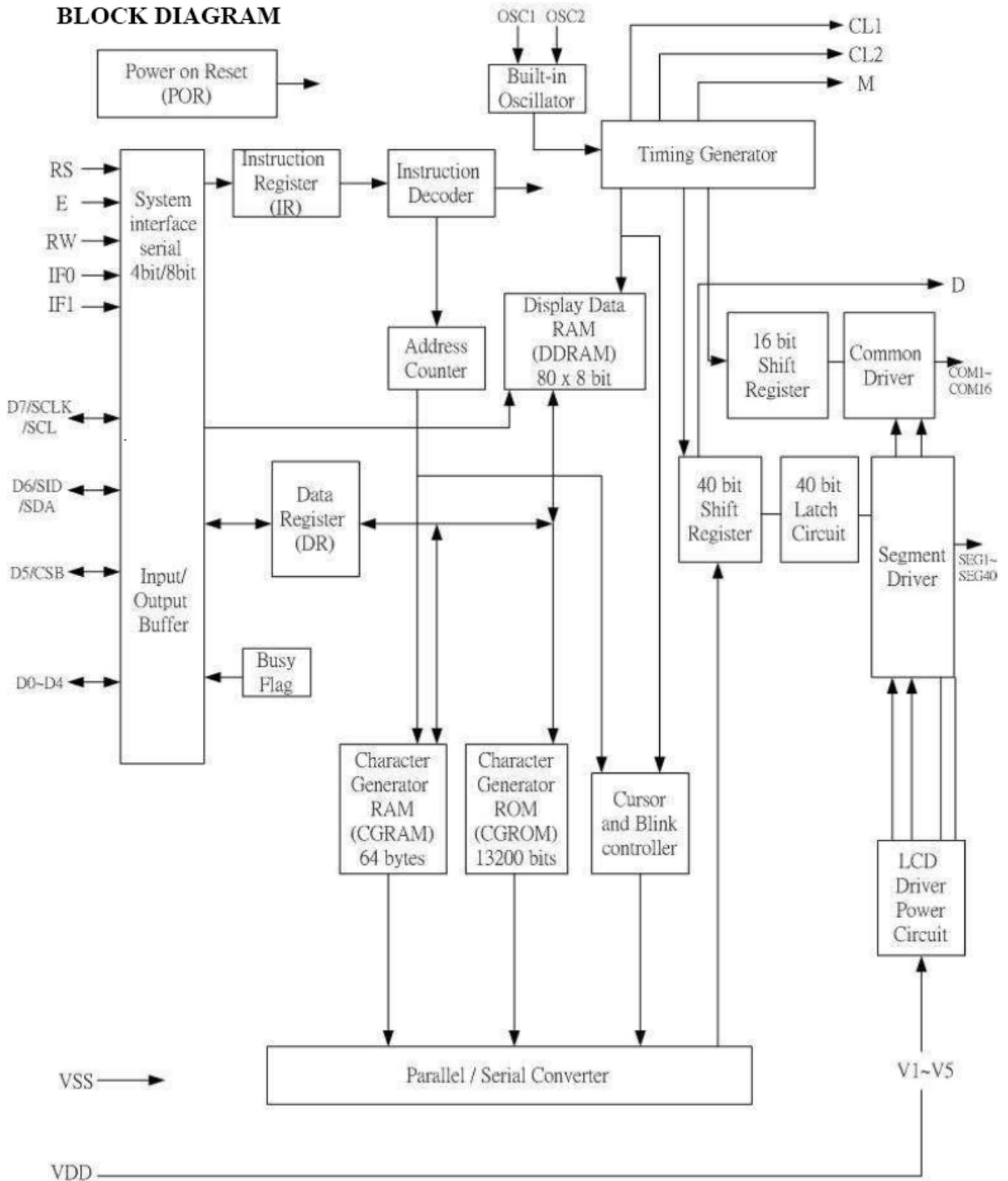


图 2: IC KS0066i 内部电路框图

## 4.4 背光参数

字符模块通常带 LED 背光板。它的性能参数如下：

工作温度： $-20\sim+70^{\circ}\text{C}$ ；

存储温度： $-30\sim+80^{\circ}\text{C}$ ；

背光板可显示绿色, 黄绿色, 兰色和白色。背光一般为绿色，也可为客户设计为其他颜色，但价格较绿色贵一点。

正常工作电流为： $10\sim 20\text{mA}$ ；

工作电压： $5.0\text{V}$ （LED 的工作电压是  $3.0\text{V}$ ，电流 $\leq 20\text{mA}$ ，因为在 PCB 上有限流电阻，所以可以用  $5.0\text{V}$  驱动）；

## 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		-20		+70	$^{\circ}\text{C}$
储存温度		-30		+80	$^{\circ}\text{C}$

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	
输入高电平	VIH	-	2.2		VDD	V
输入低电平	VIO	-	-0.3		0.6	V
输出高电平	VOH	IOH = 0.2mA	2.4		-	V
输出低电平	VOO	IOO = 1.2mA	-		0.4	V
工作电流	IDD	VDD = 5.0V		2.0		mA
液晶驱动电压	VDD - V0	Ta = 0 $^{\circ}\text{C}$		4.8		V
		Ta = 25 $^{\circ}\text{C}$		4.5		
		Ta = 50 $^{\circ}\text{C}$		4.2		

表 3：直流（DC）参数

## 6. 读写时序特性

### 6.1 从 CPU 写到 KS0066i (Writing Data from CPU to KS0066i)

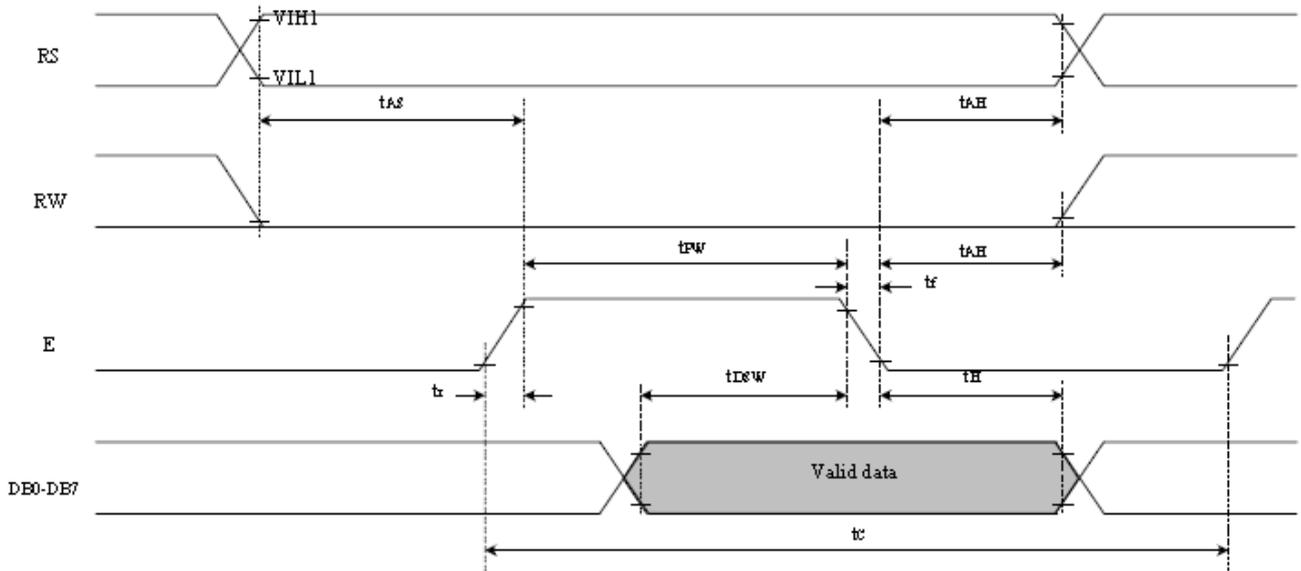


图 3. 从 CPU 写到 KS0066i (Writing Data from CPU to KS0066i)

### 6.2 从 KS0066i 读到 CPU (Reading Data from KS0066i to CPU)

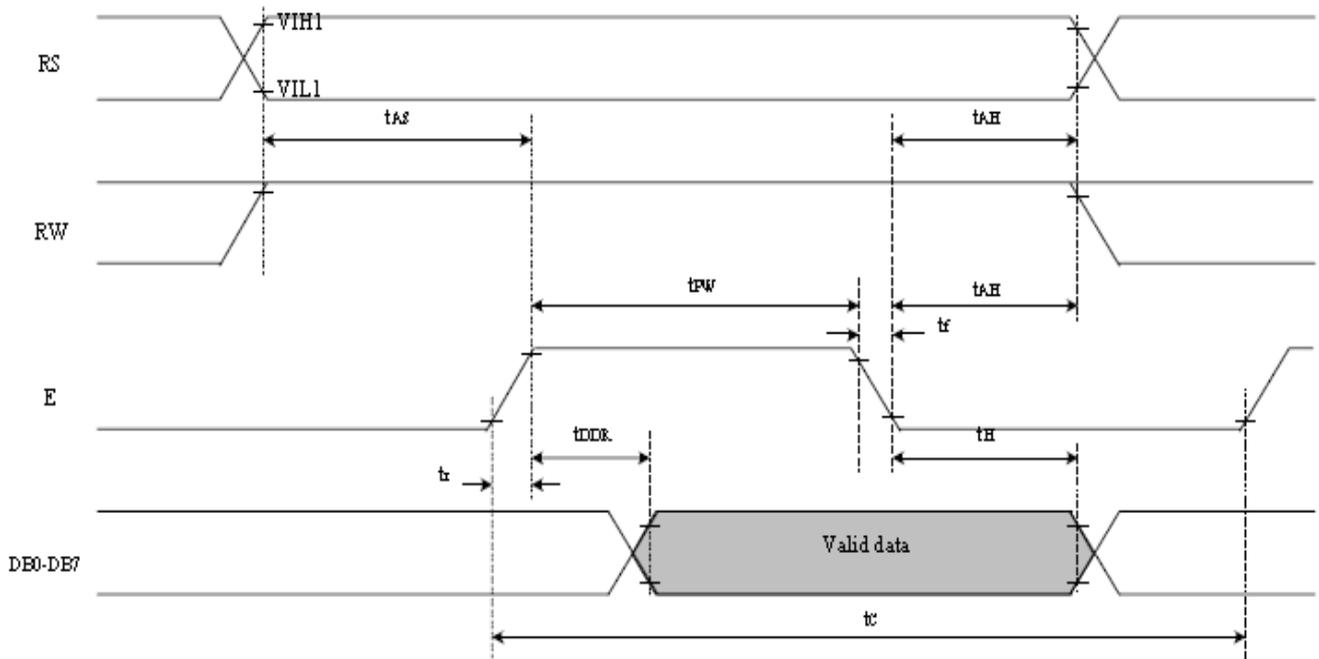


图 4: 从 KS0066i6 读到 CPU (Reading Data from KS0066i to CPU)

## 6.3 时序要求 (AC 参数):

## 写数据到 KS0066i 的时序要求:

表 4.

项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
E上升和下降时间	$T_R, T_F$	引脚: E	--	--	25	ns
E信号周期	$T_C$	引脚: E	500	--	--	ns
E脉冲宽度	$T_{PW}$	引脚: E	40	--	--	ns
地址建立时间	$T_{AS}$	引脚: E、RS, RW	0	--	--	ns
地址保持时间	$T_{AH}$	引脚: E、RS, RW	10	--	--	ns
数据建立时间	$T_{DSW}$	引脚: DBO-DB7	20	--	--	ns
数据保持时间	$T_H$	引脚: DBO-DB7	10	--	--	ns

$V_{DD} = 5.0V \pm 5\%$ ,  $T_a = 25^\circ C$

## 读数据到 KS0066i 的时序要求:

表 5.

项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
E上升和下降时间	$T_R, T_F$	引脚: E	--	--	25	ns
E信号周期	$T_C$	引脚: E	1200	--	--	ns
E脉冲宽度	$T_{PW}$	引脚: E	140	--	--	ns
地址建立时间	$T_{AS}$	引脚: E、RS, RW	0	--	--	ns
地址保持时间	$T_{AH}$	引脚: E、RS, RW	10	--	--	ns
数据建立时间	$T_{DSW}$	引脚: DBO-DB7	--	--	100	ns
数据保持时间	$T_H$	引脚: DBO-DB7	10	--	--	ns

$V_{CC} = 5.0V \pm 5\%$ ,  $T_a = 25^\circ C$

6.4 电源启动时序要求 (POWER SUPPLY CONDITION):

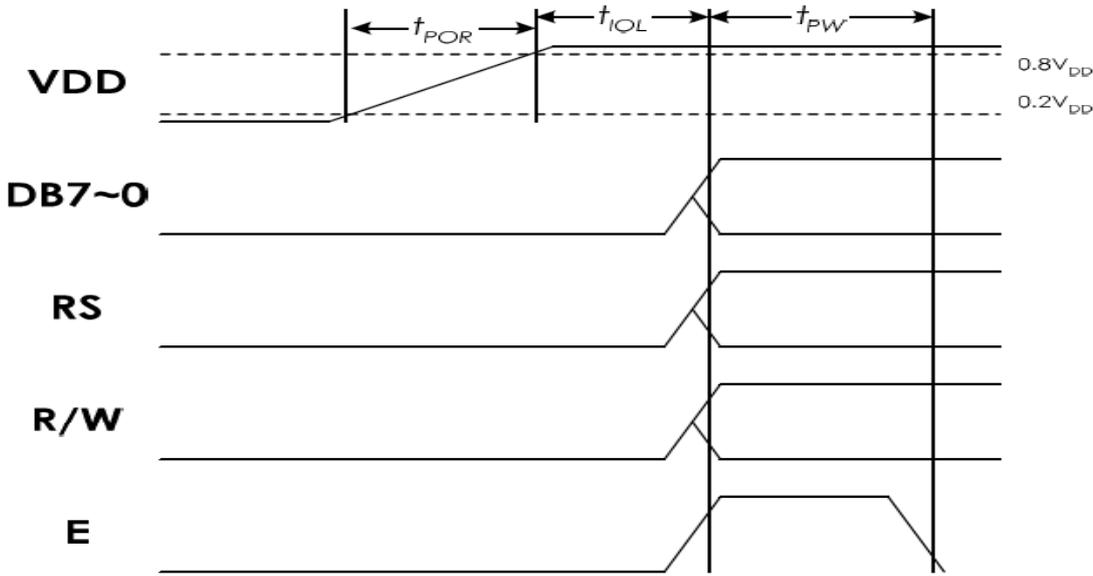


图 5: 电源启动时序

表 6: 电源启动时序要求

项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
电源启动上升时间	$t_{POR}$	引脚: VDD	0.1	--	100	ms
I/O 口保持低电平时间	$t_{IOL}$	I/O 口保持低电平的时间	40	--	--	ms
使能信号时间	$t_{PW}$	请参考上述 AC 参数				

7. 指令功能:

7.1 寄存器选择功能

表 7.

RS	R/W	操 作
0	0	指令寄存器 (IR) 写入
0	1	忙标志和地址计数器读出
1	0	数据寄存器 (DR) 写入
1	1	数据寄存器读出

备注: 忙标志为“1”时, 表明正在进行内部操作, 此时不能输入指令或数据, 要等内部操作结束, 忙标志为“0”时才能进行内部操作。

7.2 指令表

格式:

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

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	执行周期 FCP=270KHZ	
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0			
清 屏	0	0	0	0	0	0	0	0	0	0	1	清除屏幕,置AC为零	0. 76ms
返 回	0	0	0	0	0	0	0	0	0	1	X	设 DDRAM 地址为零,显示回原位,DDRAM 内容不变	0. 76ms
输入方式设置	0	0	0	0	0	0	0	0	1	I/D	S	设光标移动方向并指定整体显示是否移动	18. 5us
显示开关控制	0	0	0	0	0	0	0	1	D	C	B	设整体显示开关(D),光标开关(C),及光标位的字符闪耀(B)	18. 5us
移 位	0	0	0	0	0	0	1	S/C	R/0	X	X	移动光标或整体显示,同时不改变 DDRAM 内容	18. 5us
功能设置	0	0	0	0	0	0	D0	N	F	X	X	设接口数据位数(D0),显示行数(O),及字型(F)	18. 5us
CGRAM 地址设置	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0		设 CGRAM 地址,设置后 CGRAM 数据被发送和接收	18. 5us
DDRAM 地址设置	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0		设 DDRAM 地址,设置后 DDRAM	18. 5us
读忙信号(BF)及地址计数器	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0		读忙信号位(BF)判断内部操作正在执行并读地址计数器内容	0us
写数据 CG/DD RAM	Write data										写数据到 CG 或 DDRAM	18. 5us	
读数据由 CG/DD RAM	Read data										读数据由 CG 或 DDRAM	18. 5us	

I/D	1:增量方式, 0:减量方式	DDRAM: 显示数据 RAM	执行周期随主频率改变而改变 例如:当 Fosc 或 fcp 为 250KHZ 37us*270K/250K= 40 us
S	1:移位	CGRAM: 字符生成 RAM	
S/C	1:显示移位, 0:光标移位	AC0~AC6:用于 DD 和 CGRAM 地址的地址计数器	
R/0	1:右移, 0:左移		
D0	1:8 位, 0:4 位		
N	1:2 行, 0:1 行		
F	1:5×10, 0: 5×7		
BF	1:内部操作, 0:接收指令		
RS	:寄存器选择		
R/W	:读/写		

### 7.3 字符库及对应关系

#### 7.3.1 显示位与 DD RAM 地址的对应关系

表 9.

显示位序号		DD RAM 地址					
DD RAM	第一行	00	01	02	03	04	..... 15
地址(HEX)	第二行	40	41	42	43	44	..... 55

#### 7.3.2 标准字符库

下表所示的是字符库的内容, 字符码和字形的对应关系。例如“A”的字符码为 41 (HEX), “B”的字符码为 42 (HEX)。



b7~4 b3~0	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	CG RAM [00]			0	1	2	3	4	5	6	7	8	9	A	B	C
0001	CG RAM [01]		!	l	l	Q	a	a			.	7	7	6	6	9
0010	CG RAM [02]		"	2	B	R	b	r			7	7	7	7	7	7
0011	CG RAM [03]		#	3	O	S	s	s			7	7	7	7	7	7
0100	CG RAM [04]		*	4	D	T	d	t			7	7	7	7	7	7
0101	CG RAM [05]		%	5	E	U	e	u			.	7	7	7	7	7
0110	CG RAM [06]		&	6	F	V	f	v			7	7	7	7	7	7
0111	CG RAM [07]		'	7	G	W	g	w			7	7	7	7	7	7
1000	CG RAM [00]		(	8	H	X	h	x			7	7	7	7	7	7
1001	CG RAM [01]		)	9	I	Y	i	y			7	7	7	7	7	7
1010	CG RAM [02]		*	0	J	Z	j	z			7	7	7	7	7	7
1011	CG RAM [03]		+	1	K	k	k	k			7	7	7	7	7	7
1100	CG RAM [04]		,	<	L	7	l	l			7	7	7	7	7	7
1101	CG RAM [05]		-	=	M	m	m	m			7	7	7	7	7	7
1110	CG RAM [06]		.	>	N	n	n	n			7	7	7	7	7	7
1111	CG RAM [07]		/	?	O	o	o	e			7	7	7	7	7	7

表 10. KS0066i-0A 字库表

7.5. 3 自编字库 (CGRAM)

字符码 (DDRAM DATA), CGRAM ADDRESS 与自编字形 (CGRAM DATA) 之间关系如下面 2 个表 所示:

表 12. 5 × 7 点阵字符模式 (注明的“SST”)

DD RAM Data (字符代码)		CG RAM (地址)		CG RAM Data (字符模式)
6 5 4 3 2 1 0 MSB OSB		4 3 2 1 0 MSB OSB		6 5 4 3 2 1 0 MSB OSB
0 0 0 0 X 0 0 0		0 0 0 0 0 1 0 1 0 0 0 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1		X X X 0 1 1 1 1 1 0 0 0 0 1 0 0 0 0 ↓ 0 1 1 1 0 0 0 0 0 1 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0
0 0 0 0 X 0 0 1		0 0 0 0 0 1 0 1 0 0 0 1 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1		X X X 0 1 1 1 1 1 0 0 0 0 1 0 0 0 0 ↓ 0 1 1 1 0 0 0 0 0 1 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0
0 0 0 0 X 0 1 0		0 0 0 0 0 1 0 1 0 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1		X X X 1 1 1 1 1 0 0 1 0 0 0 0 1 0 0 ↓ 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0
:		:		:
:		:		:
:		:		:
0 0 0 0 X 1 1 1		0 0 0 0 0 1 0 1 0 1 1 1 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1		X X X ↓

X:表示不用理会

备注:1. 字符码的高4位为0000时, 它的低3位对应于第1-8个(000 - 111)自编字形;

2. 字符码的0 - 2位对应于CGRAM地址的3 - 5位;

3. 自编字形的列位置对应于CGRAM DATA的0 - 4位, 行位置对应于CGRAM ADDRESS的0 - 2位;

4. X代表无效位;

5. H代表显示位;

表 13. 5x10 点字符模式(注明” K11” )

DD RAM Data (字符代码)		CG RAM (地址)		CG RAM Data (字符模式)	
6 5 4 3 2 1 0		4 3 2 1 0		7 6 5 4 3 2 1 0	
MSB		MSB	OSB	MSB	OSB
OSB					
0 0 0 0 X 0 0 X		0 0 0 0		X X X 0 0 0 0 0	
		0 0 0 H		0 0 0 0 0	
		0 0 H 0		H 0 0 0 H	
		0 0 0 0 H H		↓ H 0 0 H 0	
		0 H 0 0		H 0 H 0 0	
		0 H 0 H		H H 0 0 H	
		0 H H 0		H 0 H 0 0	
		0 H H H		0 0 0 H 0	
		H 0 0 0		0 0 0 0 H	
		H 0 0 H		0 0 0 0 0	
	H 0 H 0		0 0 0 0 0		
	0 0	H 0 H H		X X X X X	
	H H 0 0				
	H H 0 H				
	H H H 0				
	H H H H				
0 0 0 0 X 0 H 0		0 0 0 0		X X X 0 0 0 0 0	
		0 0 0 H		H H H H H	
		0 0 H 0		0 H 0 H 0	
		0 H 0 0 H H		↓ 0 H 0 H 0	
		0 H 0 0		0 H 0 H 0	
		0 H 0 H		0 H 0 H 0	
		0 H H 0		0 H 0 H 0	
		0 H H H		0 H 0 H 0	
		H 0 0 0		H H H H H	
		H 0 0 H		0 0 0 0 0	
	H 0 H 0		0 0 0 0 0		
	0 H	H 0 H H		X X X X X	
	H H 0 0				
	H H 0 H				
	H H H 0				
	H H H H				
:		:		:	
:		:		:	
:		:		:	

0 0 0 0 X H H X		0 0 0 0		X X X X X X X X
		0 0 0 H		
		0 0 H 0		
	H H	0 0 H H		
		0 H 0 0		
		0 H 0 H		
		0 H H 0		
		0 H H H		
		H 0 0 0		
		H 0 0 H		
	H 0 H 0			
	H H	H 0 H H		
		H H 0 0		
		H H 0 H		
		H H H 0		
		H H H H		

X:表示不用理会

### 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, 下面介绍两种初始化方法;

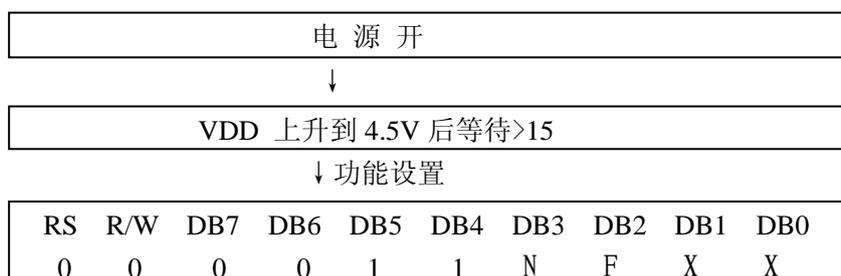
#### 7.4.1 用内部复位电路进行初始化

- 如果电路电源能满足图所示的条件的话, 就可实行初始化, 下面指令是在初始化过程中执行的。
- 清屏 (DISPOAY COEAR);
- 功能设置 (FUNCTION SET);
- DO = 1: 8Bit 接口数据;
- N = 0: 1行显示; F = 0:5×7dot 字形;
- 显示开/关控制 (DISPOAY ON/OFF CONTROO)
- D = 0: 显示关; C = 0: 光标关; B = 0: 消隐关
- 输入方式设置 (ENTRY MODE SET )
- I/D = 1: (增量); S = 0: 无移位;

#### 7.4.2 软件复位

如果电路电源不能满足复位电路的要求的话, 那么初始化就要用软件来实现, 过程如下:

8 位接口初始化流程图



↓ 等待>100uS (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

↓ 等待>100uS (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

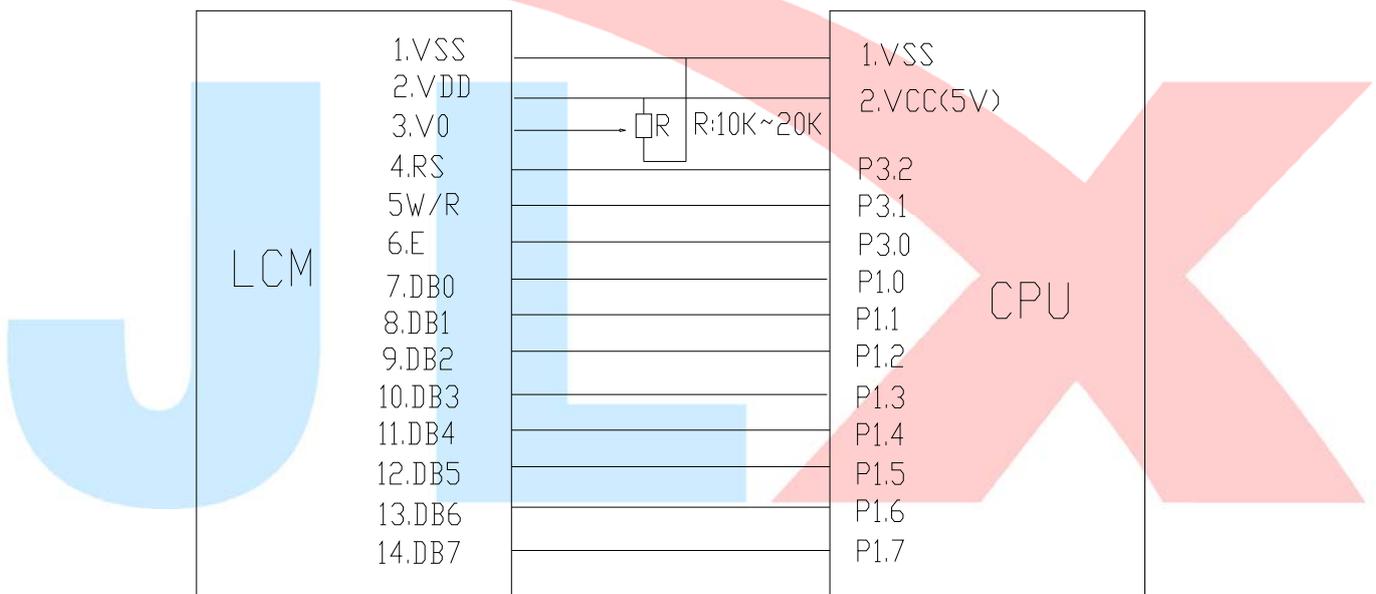
↓ 等待>10uS (进入模式设置)

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

↓  
初始化结束

### 7.4.3 程序举例:

液晶模块与 MPU(以 8051 系列单片机为例)接口图如下:



并行接口图

```

/*=====*/
/* JLX1602A 并行测试程序 */
/* LCD 驱动 IC: KS0066i 或兼容的 IC */
/* 编程者: 叶建人 */
/* 晶联讯电子: 公司网址: http://www.jlxlcd.cn; */
/*=====*/
#include <reg51.h>
sbit rs=P3^2;
    
```

```
sbit wr=P3^1;
sbit e=P3^0;
sbit busy_flag=P1^7;

/*=====*/
/*长一点的延时*/
void delay(int i)
{
    int j,k;
    for(j=0;j<i;j++)
        for(k=0;k<990;k++);
}

/*短一点的延时*/
void delay1(int i)
{
    int j,k;
    for(j=0;j<i;j++)
        for(k=0;k<10;k++);
}

/*等待一个按键 (P2.0 口与 GND 之间) */
void waitkey()
{
    repeat:
        if (P2&0x01) goto repeat;
        else delay(5);
        if (P2&0x01) goto repeat;
        else;
        delay(40);
}

/*检查忙标志位 DB7*/
void check_busy()
{
    rs=0;
    busy_flag=1;
    wr=1;
    if(busy_flag=1)
    {
        e=1;
        e=0;
    }
    else;
}

/*=====写指令=====*/
```

```
void transfer_command(int data1)
{
    check_busy();
    rs=0;
    delay1(10);
    wr=0;
    delay1(10);
    P1=data1;
    e=1;
    delay1(10);
    e=0;
}
```

```
/*-----写数据-----*/
```

```
void transfer_data(int data1)
{
    check_busy();
    rs=1;
    delay1(10);
    wr=0;
    delay1(10);
    P1=data1;
    e=1;
    delay1(10);
    e=0;
}
```

```
/*初始化 LCD MODULE*/
```

```
void initial_lcd()
{
    transfer_command(0x38);    /*function select*/
    transfer_command(0x01);    /*clear screen*/
    delay(5);
    transfer_command(0x06);    /*setdisplay mode*/
    delay(5);
    transfer_command(0x0c);    /*turn on display*/
    delay(5);
}
```

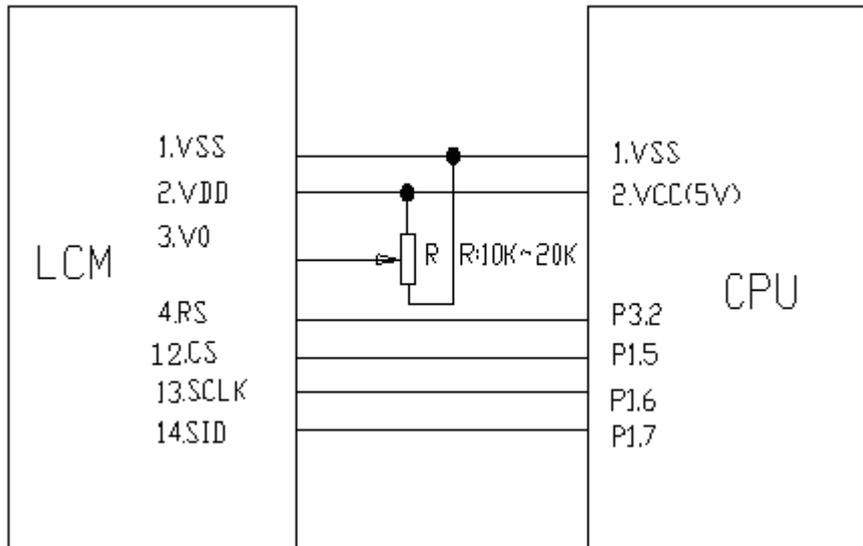
```
/*在指定行和列位置显示指定的字母、数字（5*7 点阵的）*/
```

```
void disp_char(int line,int column,char code *dp)
{
```

```
int i;
transfer_command(0x80+(line-1)*0x40+(column-1)); //set position
for(i=0;i<16;i++)
{
    transfer_data(*dp);
    dp=dp+1;
}

/*主程序*/
void main(void)
{
    e=0;
    initial_lcd();
    while(1)
    {
        disp_char(1,1,"*16*2 LCM no BL*"); /*在第1行,第1列,显示字符....*/
        disp_char(2,1,"**JLX1602I LCM**"); /*在第2行,第1列,显示字符....*/
        waitkey();
        disp_char(1,1,"16X2 characters:"); /*在第1行,第1列,显示字符....*/
        disp_char(2,1,"*standard ascii*"); /*在第2行,第1列,显示字符....*/
        waitkey();
    }
}
```

→



串行接口图

```

/* 液晶演示程序 JLX1602A-4-SPI
   驱动 IC 是:KS0066i, 4 线 SPI 串行接口
   编程者: 叶建人, 4 月 13 日, 2012
   晶联讯电子: 网址 http://www.jlxlcd.cn;

```

```

*/
#include <reg52.h>
#include <intrins.h>
#include <ctype.h>

```

```

sbit key=P2^0;

sbit rs=P3^2;
sbit cs=P1^5;
sbit sclk=P1^6;
sbit sid=P1^7;

```

```

#define uchar unsigned char
#define uint unsigned int
#define ulong unsigned long

```

```

char code CGRAM_data[]={
0x08, 0x1F, 0x02, 0x0F, 0x0A, 0x1F, 0x02, 0x02, //“年”
0X55, 0X55, 0X55, 0X55, 0X55, 0X55, 0X55, 0X55, //偶竖
0XAA, 0XAA, 0XAA, 0XAA, 0XAA, 0XAA, 0XAA, 0XAA, //奇竖
0XFF, 0X00, 0XFF, 0X00, 0XFF, 0X00, 0XFF, 0X00, //奇横
0X00, 0XFF, 0X00, 0XFF, 0X00, 0XFF, 0X00, 0XFF, //偶横
0XFF, 0X11, 0X11, 0X11, 0X11, 0X11, 0X11, 0XFF, //方框
0XFF, 0X11, 0X11, 0X11, 0X11, 0X11, 0X11, 0XFF, //方框
0XFF, 0X11, 0X11, 0X11, 0X11, 0X11, 0X11, 0XFF, //方框

```

```
};  
char code CGRAM_data_nian[]={  
0x08, 0x1F, 0x02, 0x0F, 0x0A, 0x1F, 0x02, 0x02,  
};
```

```
/*延时*/
```

```
void delay(int i)  
{  
    int j,k;  
    for(j=0;j<i;j++)  
        for(k=0;k<50;k++);  
}
```

```
void waitkey()
```

```
{  
repeat:  if(key==1)  
        goto repeat;  
        else  
        delay(400);  
}
```

```
/*写指令到 LCD 模块*/
```

```
void transfer_command(int data1)  
{  
    char i;  
    cs=0;  
    rs=0;  
    for(i=0;i<8;i++)  
    {  
        sclk=0;  
        if(data1&0x80) sid=1;  
        else sid=0;  
        delay(5);  
        sclk=1;  
        delay(5);  
        data1=data1<<=1;  
    }  
}
```

```
/*写数据到 LCD 模块*/
```

```
void transfer_data(int data1)  
{  
    char i;  
    cs=0;
```

```
rs=1;
for(i=0;i<8;i++)
{
    sclk=0;
    if(data1&0x80) sid=1;
    else sid=0;
    delay(1);
    sclk=1;
    delay(1);
    data1=data1<<=1;
}
}

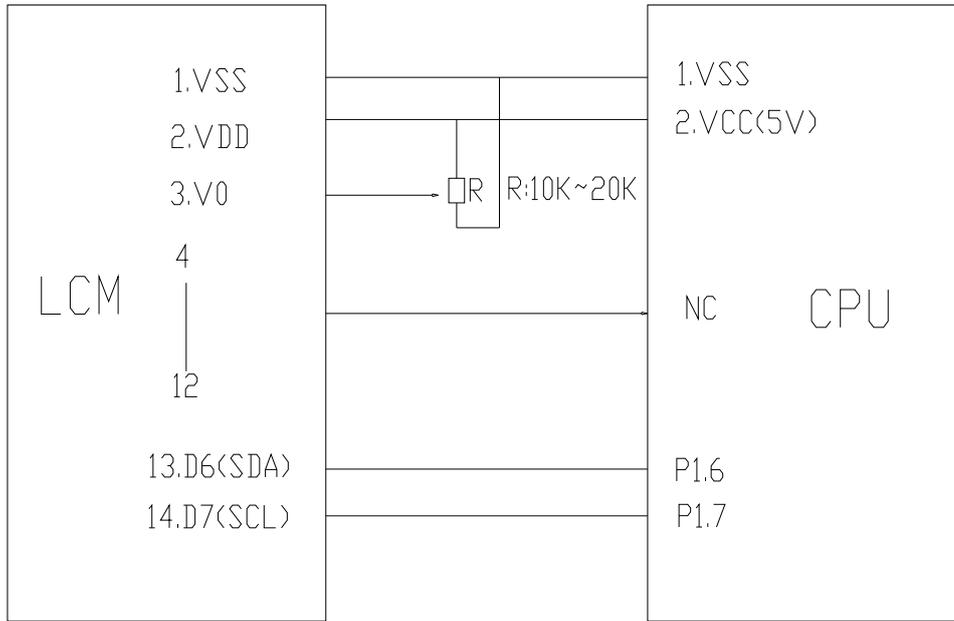
/*LCD 模块初始化*/
void initial_lcd()
{
    cs=0;
    delay(20);
    transfer_command(0x38); /**/
    transfer_command(0x0C); /**/
    transfer_command(0x01); /**/
    transfer_command(0x06); /**/
}

/*自编字符并存储起来，存至字库列表的 0x00 到 0x07 单元*/
void write_CGRAM()
{
    int i;
    transfer_command(0x40); //设置 XGRAM ADDRESS: 第几个 CGRAM, 0x40 表示第 0 个。
    for(i=0;i<64;i++)
    {
        transfer_data(CGRAM_data[i]);
    }
}

/*显示自编的字符*/
void disp_CGRAM()
{
    int i, j;
    for(j=0;j<6;j++)
    {
        transfer_command(0x80); //设置 DDRAM ADDRESS: 第几行, 第几列
        for(i=0;i<16;i++)
        {
            transfer_data(j);
        }
        transfer_command(0xc0); //设置 DDRAM ADDRESS: 第几行, 第几列
        for(i=0;i<16;i++)
```

```
        {
            transfer_data(j);
        }
        waitkey();
    }
}
/*在指定行和列位置显示指定的字母、数字（5*7点阵的）*/
void disp_char(int line,int column,char code *dp)
{
    int i;
    transfer_command(0x80+(line-1)*0x40+(column-1)); //设置 DDRAM ADDRESS: 第几行, 第几列
    for(i=0;i<16;i++)
    {
        transfer_data(*dp);
        dp=dp+1;
    }
}
void main()
{
    delay(100);
    initial_lcd();
    write_CGRAM();
    while(1)
    {
        disp_char(1,1,"16*2 LCD MODULE:"); /*在第1行, 第1列, 显示字符.... */
        disp_char(2,1,"JLX1602A-1-SPI**"); /*在第2行, 第1列, 显示字符.... */
        waitkey();
        disp_char(1,1,"16X2 characters:"); /*在第1行, 第1列, 显示字符.... */
        disp_char(2,1,"*standard ascii*"); /*在第2行, 第1列, 显示字符.... */
        waitkey();
        disp_char(1,1,"1234567890;=<>?"); /*在第1行, 第1列, 显示字符.... */
        disp_char(2,1,"ABCDEFGHabcdefgh"); /*在第2行, 第1列, 显示字符.... */
        waitkey();

        disp_CGRAM(); /*显示自编的 CGRAM 字符.... */
    }
}
}
```



IIC 接口图

```

/*****
;      *      CONTROLLER:KS0066i, IIC 总线      *
;      *      FUNCTION:  TEST JLX1602A        *
;      *      PROGRAMED BY 叶建人            *
;      *      DATE:   2012. 3. 5              *
;      *      VOP=VDD-V0                      *
*****/
#include <reg51.h>
sbit scl=P1^7;
sbit sda=P1^6;

#define uchar unsigned char
#define uint unsigned int
#define ulong unsigned long

char code CGRAM_data[]={
0x08, 0x1F, 0x02, 0x0F, 0x0A, 0x1F, 0x02, 0x02, //年
0X55, 0X55, 0X55, 0X55, 0X55, 0X55, 0X55, 0X55, //偶竖
0XAA, 0XAA, 0XAA, 0XAA, 0XAA, 0XAA, 0XAA, 0XAA, //奇竖
0XFF, 0X00, 0XFF, 0X00, 0XFF, 0X00, 0XFF, 0X00, //奇横
0X00, 0XFF, 0X00, 0XFF, 0X00, 0XFF, 0X00, 0XFF, //偶横
0XFF, 0X11, 0X11, 0X11, 0X11, 0X11, 0X11, 0XFF, //方框
0XFF, 0X11, 0X11, 0X11, 0X11, 0X11, 0X11, 0XFF, //方框
0XFF, 0X11, 0X11, 0X11, 0X11, 0X11, 0X11, 0XFF, //方框
};

char code CGRAM_data_nian[]={
0x08, 0x1F, 0x02, 0x0F, 0x0A, 0x1F, 0x02, 0x02,

```

```
};  
//=====delay time=====  
void delay(int i)  
{  
    int j,k;  
    for(j=0;j<i;j++)  
        for(k=0;k<110;k++);  
}  
//-----wait a switch, jump out if P2.0 get a signal"0"-----  
void waitkey()  
{  
    repeat:  
        if (P2&0x01) goto repeat;  
        else;  
            delay(500);  
}  
void transfer(int data1)  
{  
    int i;  
    for(i=0;i<8;i++)  
    {  
        scl=0;  
        if(data1&0x80) sda=1;  
        else sda=0;  
        scl=1;  
        scl=0;  
        data1=data1<<1;  
    }  
    sda=0;  
    scl=1;  
    scl=0;  
}  
  
void start_flag()  
{  
    scl=1;    /*START FLAG*/  
    sda=1;    /*START FLAG*/  
    sda=0;    /*START FLAG*/  
}  
  
void stop_flag()  
{  
    scl=1;    /*STOP FLAG*/  
    sda=0;    /*STOP FLAG*/  
    sda=1;    /*STOP FLAG*/  
}
```

```

}
void initial_lcd()
{
    start_flag();      /*开始标志*/

    transfer(0x78);    /*选择 SLAVE ADDRESS*/
    transfer(0x00);    /*控制字节: Co=0, A0=0, 表示以下传输的 N 个字节是指令*/
    transfer(0x38);    //Function set
    transfer(0x0c);    //Display ON/OFF
    transfer(0x01);    //Clear display
    transfer(0x06);    //Entry mode set
    stop_flag();      /*结束标志*/
}
void write_CGRAM()
{
    int i;
    char *dp=CGRAM_data;
    start_flag();
    transfer(0x78);    /*选择 SLAVE ADDRESS*/
    transfer(0x80);    /*控制字节: Co=1, A0=0, 表示以下传输的有且只有 1 个字节是指令*/
    transfer(0x40);    //设置 XGRAM ADDRESS: 第几个 CGRAM, 0X40 表示第 0 个。
    transfer(0x40);    /*控制字节: Co=0, A0=1, 表示以下传输的 n 个字节是数据*/
    for(i=0;i<64;i++)
    {
        transfer(*dp);
        dp=dp+1;
    }
    stop_flag();
}
void disp_CGRAM()
{
    int i;
    start_flag();
    transfer(0x78);    /*选择 SLAVE ADDRESS*/
    transfer(0x80);    /*控制字节: Co=1, A0=0, 表示以下传输的 1 个字节是指令*/
    transfer(0x80);    //设置 DDRAM ADDRESS: 第几行, 第几列
    transfer(0x40);    /*控制字节: Co=0, A0=1, 表示以下传输的 n 个字节是数据*/
    for(i=0;i<16;i++)
    {
        transfer(0x01);
    }
    stop_flag();
    start_flag();
    transfer(0x78);    /*选择 SLAVE ADDRESS*/
    transfer(0x80);    /*控制字节: Co=1, A0=0, 表示以下传输的 1 个字节是指令*/
}

```

```

transfer(0xc0); //设置 DDRAM ADDRESS: 第几行, 第几列
transfer(0x40); /*控制字节: Co=0, A0=1, 表示以下传输的 n 个字节是数据*/
for(i=0;i<16;i++)
{
    transfer(0x01);
}
stop_flag();

}
/*在指定行和列位置显示指定的字母、数字 (5*7 点阵的) */
void disp_char(int line,int column,char code *dp)
{
    int i;
    start_flag();
    transfer(0x78); /*选择 SLAVE ADDRESS*/
    transfer(0x80); /*控制字节: Co=1, A0=0, 表示以下传输的 1 个字节是指令*/
    transfer(0x80+(line-1)*0x40+(column-1)); //设置 DDRAM ADDRESS: 第几行, 第几列
    transfer(0x40); /*控制字节: Co=0, A0=1, 表示以下传输的 n 个字节是数据*/
    for(i=0;i<16;i++)
    {
        transfer(*dp);
        dp=dp+1;
    }
    stop_flag();
}
void main()
{
    delay(100);
    initial_lcd();
    write_CGRAM();
    while(1)
    {
        disp_char(1,1,"*16*2 LCM no BL*"); /*在第 1 行, 第 1 列, 显示字符.... */
        disp_char(2,1,"**JLX1602I LCM**"); /*在第 2 行, 第 1 列, 显示字符.... */
        waitkey();
        disp_char(1,1,"16X2 characters:"); /*在第 1 行, 第 1 列, 显示字符.... */
        disp_char(2,1,"*standard ascii*"); /*在第 2 行, 第 1 列, 显示字符.... */
        waitkey();
        disp_CGRAM(); /*显示自编的 CGRAM 字符.... */
        waitkey();
    }
}
}

```

—END—