

# DM12232F 说明书

## 一、概述

DM12232F 是一种图形点阵液晶显示器,它主要由行驱动器/列驱动器及 122×32 全点阵液晶显示器组成。可完成 16×2 个(16×8 点阵)ASCII 码显示,也可以显示 7.5×2 个(16×16 点阵)汉字。与外部 CPU 接口可采用并行方式控制。

主要技术参数和性能:

1. 电源:VDD:+2.7~+5V。
2. 显示内容:122(列)×32(行)点。
3. 全屏幕点阵。
4. 2M ROM(CGROM)总共提供 8192 个汉字(16×16 点阵)。
5. 16K ROM(HCGROM)总共提供 128 个字符(16×8 点阵)。
6. 2MHZ 频率。
7. 工作温度: -15℃ ~ +60℃, 存储温度: -20℃ ~ +70℃

## 二、外形尺寸图

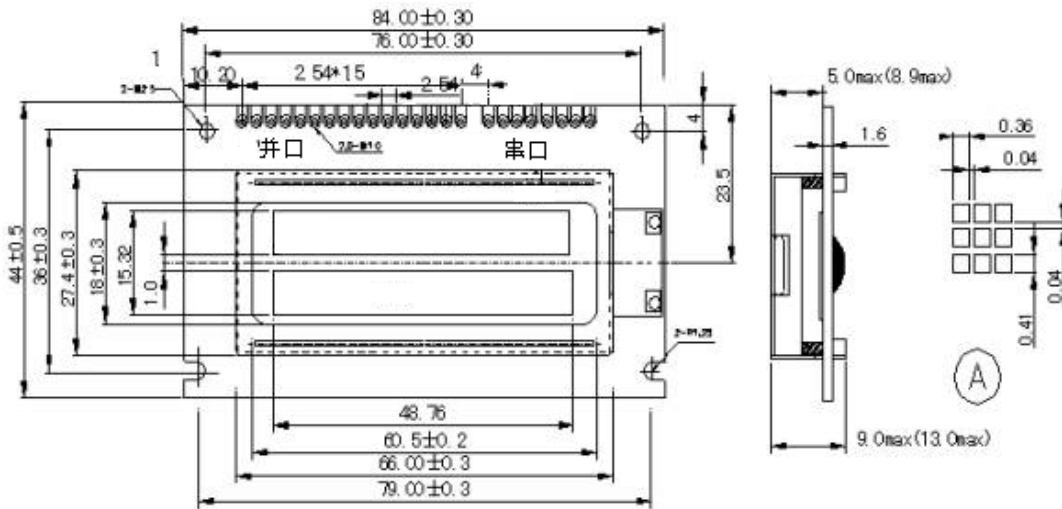


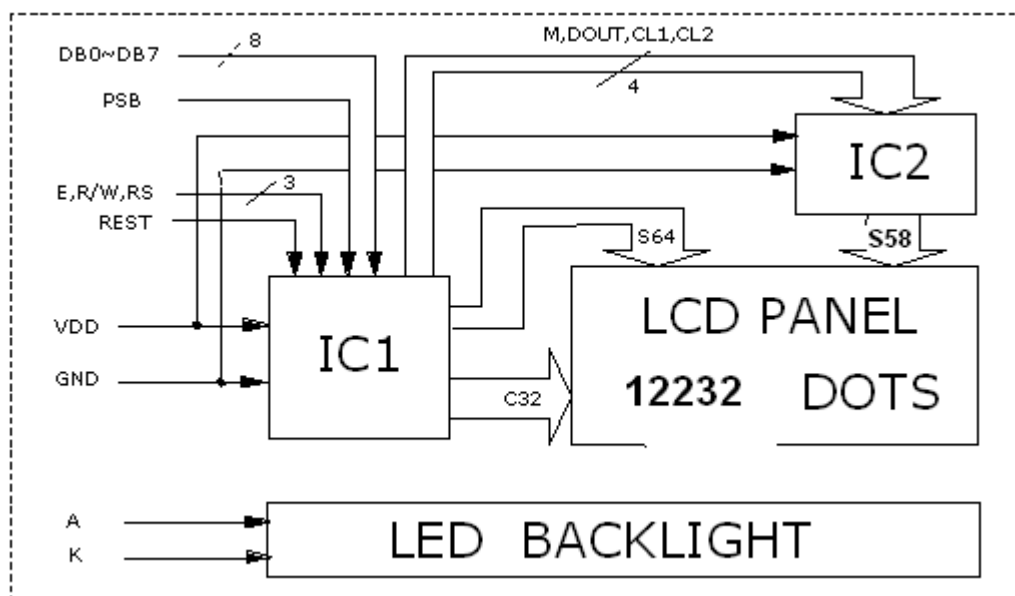
图 1

## 2. 外形尺寸图

表 1

项目	正常尺寸	单位
模块体积	84×44×13	mm
视域	60.5×18.0	mm
行列点阵数	122×32	DOTS
点距离	0.36×0.41	mm
点大小	0.40×0.45	mm

### 三、模块主要硬件构成说明



RS, R/W 的配合选择决定控制界面的 4 种模式:

RS	R/W	功能说明
L	L	MPU 写指令到指令暂存器 (IR)
L	H	读出忙标志 (BF) 及地址计数器 (AC) 的状态
H	L	MPU 写入数据到数据暂存器 (DR)
H	H	MPU 从数据暂存器 (DR) 中读出数据

- **忙标志:BF**

BF 标志提供内部工作情况. BF=1 表示模块在进行内部操作, 此时模块不接受外部指令和数据. BF=0 时, 模块为准备状态, 随时可接受外部指令和数据.

利用 STATUS R D 指令, 可以将 BF 读到 DB7 总线, 从而检验模块之工作状态.

- **字型产生 ROM (CGROM)**

字型产生 ROM (CGROM) 提供 8192 个此触发器是用于模块屏幕显示开和关的控制. DFF=1 为开显示 (DISPLAY ON), DDRAM 的内容就显示在屏幕上, DFF=0 为关显示 (DISPLAY OFF)。

DFF 的状态是指令 DISPLAY ON/OFF 和 RST 信号控制的。

- **显示数据 RAM (DDRAM)**

显示数据 RAM 提供 64×2 个位元组的空间, 最多可控制 4 行 16 字 (64 个字) 的中文字型显示, 当写入显示数据 RAM 时, 可分别显示 CGROM 与 CGRAM 的字型; 此模块可显示三种字型, 分别是瘦长的英数字型、CGRAM 字型及 CGROM 的中文字型, 三种字型的选择, 由在 DDRAM 中写入的编码选择, 在 00~0F 的编码中将选择 CGRAM 的字定义字型, 10~7F 的编码中将选择瘦长英数字的字型, 至于 A0 以上的编码将自动的结合下一个位元组, 组成两个位元组的编码形成中

文字型的编码 (A140~D75F)。

- **字型产生 RAM (CGRAM)**

字型产生 RAM 提供图象定义(造字)功能, 可以提供四组 16×16 点的自定义图象空间, 使用者可以将内部字型没有提供的图象字型自行定义到 CGRAM 中, 便可和 CGRAM 中的定义一般的通过 DDRAM 显示在荧屏中。

- **地址计数器 AC**

地址计数器是用来贮存 DDRAM/CGRAM 之一的地址, 它可由设定指令暂存器来改变, 之后只要读取或是写入 DDRAM/CGRAM 的值时, 地址计数器的值就会自动加一, 当 RS 为“0”时而 R/W 为“1”时, 地址计数器的值会被读取到 DB6~DB0 中。

- **ICON RAM (IRAM)**

IC1 提供 256 点的 ICON 显示, 它分别由 16 组的 IRAM 地址来组成, 每一组 IRAM 地址由 16 个位元构成, 每次写入一组 IRAM 时, 必须通过连续写入两个位元组的资料来完成, 先写入高位元组再写入低位元组。

- **LCD 驱动电路**

LCD 驱动电路提供 33 COMMON 以及 64 SEGMENT 信号来驱动 LCD 棉板, SEGMENT 数据从 CGRAM/CGROM 转换储存到 64 位元的 SEGMENT 串列锁存, 当 33 个 COMMON 中的一个 COMMON 输出时, 相对应的 SEGMENT 数据将从 64 位元的串列锁存输出到 SEGMENT 驱动电路。

- **游标/闪烁控制电路**

此模块提供硬体游标及闪烁控制电路, 由地址计数器的值来指定 DDRAM 中的游标或闪烁位置。

### **.绘图 RAM (GDRAM)**

绘图显示 RAM 提供 128×8 个字节的记忆空间, 在更改绘图 RAM 时, 先连续写入水平与垂直的坐标值, 再写入两个字节的的数据到绘图 RAM, 而地址计数器 (AC) 会自动加一; 在写入绘图 RAM 的期间, 绘图显示必须关闭, 整个写入绘图 RAM 的步骤如下:

- 1、关闭绘图显示功能。
  - 2、先将水平的位元组坐标 (X) 写入绘图 RAM 地址;  
再将垂直的坐标 (Y) 写入绘图 RAM 地址;  
将 D15——D8 写入到 RAM 中;  
将 D7——D0 写入到 RAM 中;  
打开绘图显示功能。
- 绘图显示的缓冲区对应分布请参考“显示坐标”

## **四·模块的外部接口**

外部接口信号如下表 2、3 所示 (并行接口):

表 2

管脚号	管脚名称	LEVER	管脚功能描述
1	VSS	0V	电源地
2	VCC	3.0+5V	电源正
3	VEE	-	对比度调整
4	RS (CS)	H/L	RS=“H”, 表示 DB7~DB0 为显示数据 RS=“L”, 表示 DB7~DB0 为显示指令数据
5	R/W (SID)	H/L	R/W=“H”, E=“H”, 数据被读到 DB7~DB0 R/W=“L”, E=“H→L”, DB7~DB0 的数据被写到 IR 或 DR
6	E (CLK)	H/L	使能信号
7	DB0	H/L	数据线
8	DB1	H/L	数据线
9	DB2	H/L	数据线

10	DB3	H/L	数据线
11	DB4	H/L	数据线
12	DB5	H/L	数据线
13	DB6	H/L	数据线
14	DB7	H/L	数据线
15	BL+	VDD	背光源电压+4.2V—+5V
16	BL-	Vss	背光源公共端

### 串口接口管脚信号

表 3

管脚号	名称	LEVER	功能
1	VSS	0V	电源地
2	VDD	+5V	电源正 (3.0V~5.5V)
3	VEE	-	对比度调整
4	CLK	H/L	串行同步时钟：上升沿时读取 SID 数据
5	SID	H/L	串行数据输入端
6	CS	H/L	模组片选端，高电平有效
7	BL+	VDD	背光源电压+4.2V—+5V
8	BL-	VSS	背光源公共端

注：出厂默认为串口方式已 PSB 接地；V0 可通过外部电阻调节。

## 五、指令说明

IC1 提供两套控制命令，基本指令和扩充指令如下：

指令表 1：(RE=0：基本指令)

指令	指令码										功能
	RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0	
清除显示	0	0	0	0	0	0	0	0	0	1	将 DDRAM 填满“20H”，并且设定 DDRAM 的地址计数器 (AC) 到“00H”
地址归位	0	0	0	0	0	0	0	0	1	X	设定 DDRAM 的地址计数器 (AC) 到“00H”，并且将游标移到开头原点位置；这个指令不改变 DDRAM 的内容
显示状态开/关	0	0	0	0	0	0	1	D	C	B	D=1：整体显示 ON C=1：游标 ON

											B=1: 游标位置 ON		
进入点 设定	0	0	0	0	0	0	0	0	1	I/D	S	指定在数据的读取与写入时, 设定游标的移动方向及指定显示的移位 I/D=1: 游标向右移, DDRAM 地址计数器 (AC) 加 1 I/D=0: 游标向左移, DDRAM 地址计数器 (AC) 减 1 S: 显示画面整体位移	
游标或 显示移 位控制	0	0	0	0	0	1	S/C	R/L	X	X		设定游标的移动与显示的移位控制位; 这个指令不改变 DDRAM 的内容 S/C=0, R/L=0: 游标向左移动 S/C=0, R/L=1: 游标向右移动	
功能 设定	0	0	0	0	1	DL	X	0	RE	X	X	DL=1 (必须设为 1) RE=1: 扩充指令操作 RE=0: 基本指令操作	
设定 CGRAM 地址	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0		设定 CGRAM 地址到地址计数器	
设定 DDRAM 地址	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0		设定 DDRAM 地址到地址计数器	
读取忙 标志和 地址	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0		读取忙标志 (BF) 可以确认内部动作是否完成, 同时可以读出地址计数器 (AC) 的值	
写数据 到 RAM	1	0	数据										将数据 D7~D0 写入到内部的 RAM (DDRAM/CGRAM/IRAM/GRAM)
读出 RAM 的值	1	1	数据										从内部 RAM 读取数据 D7~D0 (DDRAM/CGRAM/IRAM/GRAM)

指令表 2: (RE=1: 扩充指令)

指令	指令码										功能	
	RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0		
待命 模式	0	0	0	0	0	0	0	0	0	0	1	进入待命模式, 执行其他指令都终止待命模式
卷动地址 开关开启	0	0	0	0	0	0	0	0	0	1	SR	SR=1: 允许输入卷动地址 SR=0: 允许输入 IRAM 地址
反白 选择	0	0	0	0	0	0	0	0	1	R1	R0	选择 4 行中的任一行作反白显示, 并可决定反白与否
睡眠 模式	0	0	0	0	0	0	0	1	SL	X	X	SL=0: 进入睡眠模式 SL=1: 脱离睡眠模式
点距 书面 移位 控制	0	0	0	0	0	1	OA	LR	L1	L0		OA=1: 选择单行移位 OA=0: 全部 4 行一起移位 LR=1: 点距右移 LR=0: 点距左移 L1, L0: 选择移位行
扩充 功能 设定	0	0	0	0	1	CL	X	1	RE	G	GP	CL=1 (必须设为 1) RE=1: 扩充指令操作 RE=0: 基本指令操作 G=0: (必须设为 0) GP=0: (必须设为 0)

设定 IRAM 地址或是卷动地址	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	SR=1: AC5~AC0 为卷动地址 SR=0: AC5~AC0 为 ICON RAM 地址
设定绘图 RAM 地址	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	本版本不提供此功能 0

备注:当 IC1 在接受指令前,微处理器必须先确认其内部处于非忙碌状态,即读取 BF 标志时,BF 需为零,方可接受新的指令;如果在送出一个指令前并不检查 BF 标志,那么在前一个指令和这个指令中间必须延长一段较长的时间,即是等待前一个指令确实执行完成。

## 具体指令介绍:

### 1、清除显示

CODE:      RS    RW    DB7    DB6    DB5    DB4    DB3    DB2    DB1    DB0

L	L	L	L	L	L	L	L	L	L	H
---	---	---	---	---	---	---	---	---	---	---

功能:清除显示屏幕,把 DDRAM 位址计数器调整为“00H”

### 2、位址归位

CODE:      RS    RW    DB7    DB6    DB5    DB4    DB3    DB2    DB1    DB0

L	L	L	L	L	L	L	L	L	H	X
---	---	---	---	---	---	---	---	---	---	---

功能:把 DDRAM 位址计数器调整为“00H”,游标回原点,该功能不影响显示 DDRAM

### 3、位址归位

CODE:      RS    RW    DB7    DB6    DB5    DB4    DB3    DB2    DB1    DB0

L	L	L	L	L	L	L	L	H	I/D	S
---	---	---	---	---	---	---	---	---	-----	---

功能:把 DDRAM 位址计数器调整为“00H”,游标回原点,该功能不影响显示 DDRAM 功能:执行该命令后,所设置的行将显示在屏幕的第一行。显示起始行是由 Z 地址计数器控制的,该命令自动将 A0-A5 位地址送入 Z 地址计数器,起始地址可以是 0-63 范围内任意一行。Z 地址计数器具有循环计数功能,用于显示行扫描同步,当扫描完一行后自动加一。

### 4、显示状态 开/关

CODE:      RS    RW    DB7    DB6    DB5    DB4    DB3    DB2    DB1    DB0

L	L	L	L	L	L	L	H	D	C	B
---	---	---	---	---	---	---	---	---	---	---

功能: D=1; 整体显示 ON      C=1; 游标 ON      B=1; 游标位置 ON

### 5、游标或显示移位控制

CODE:      RS    RW    DB7    DB6    DB5    DB4    DB3    DB2    DB1    DB0

L	L	L	L	L	H	S/C	R/L	X	X
---	---	---	---	---	---	-----	-----	---	---

功能:设定游标的移动与显示的移位控制位:这个指令并不改变 DDRAM 的内容

### 6、功能设定

CODE:      RS    RW    DB7    DB6    DB5    DB4    DB3    DB2    DB1    DB0

L	L	L	L	H	DL	X	0 RE	X	X
---	---	---	---	---	----	---	------	---	---

功能: DL=1 (必须设为 1)      RE=1; 扩充指令集动作      RE=0: 基本指令集动作

### 7、设定 CGRAM 位址

CODE:      RS    RW    DB7    DB6    DB5    DB4    DB3    DB2    DB1    DB0

L	L	L	H	AC5	AC4	AC3	AC2	AC1	AC0
---	---	---	---	-----	-----	-----	-----	-----	-----

功能：设定 CGRAM 位址到位址计数器（AC）

#### 8、设定 DDRAM 位址

CODE:

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
L	L	H	AC6	AC5	AC4	AC3	AC2	AC1	AC0

功能：设定 DDRAM 位址到位址计数器（AC）

#### 9、读取忙碌状态（BF）和位址

CODE:

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
L	H	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0

功能：读取忙碌状态（BF）可以确认内部动作是否完成，同时可以读出位址计数器（AC）的值

#### 10、写资料到 RAM

CODE:

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
H	L	D7	D6	D5	D4	D3	D2	D1	D0

功能：写入资料到内部的 RAM（DDRAM/CGRAM/TRAM/GDRAM）

#### 11、读出 RAM 的值

CODE:

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
H	H	D7	D6	D5	D4	D3	D2	D1	D0

功能：从内部 RAM 读取资料（DDRAM/CGRAM/TRAM/GDRAM）

#### 12、待命模式（12H）

CODE:

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
L	L	L	L	L	L	L	L	L	H

功能：进入待命模式，执行其他命令都可终止待命模式

#### 13、卷动位址或 IRAM 位址选择（13H）

CODE:

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
L	L	L	L	L	L	L	L	H	SR

功能：SR=1；允许输入卷动位址      SR=0；允许输入 IRAM 位址

#### 14、反白选择（14H）

CODE:

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
L	L	L	L	L	L	L	H	R1	R0

功能：选择 4 行中的任一行作反白显示，并可决定反白的与否

#### 15、睡眠模式（015H）

CODE:

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
L	L	L	L	L	L	H	SL	X	X

功能：SL=1；脱离睡眠模式      SL=0；进入睡眠模式

## 16、扩充功能设定 (016H)

CODE:	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	L	L	L	L	H	H	X	1 RE	G	L

功能: RE=1; 扩充指令集动作 RE=0; 基本指令集动作 G=1; 绘图显示 ON G=0; 绘图显示 OFF

## 17、设定 IRAM 位址或卷动位址 (017H)

CODE:	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	L	L	L	H	AC5	AC4	AC3	AC2	AC1	AC0

功能: SR=1; AC5~AC0 为垂直卷动位址 SR=0; AC3~AC0 写 ICONRAM 位址

## 18、设定绘图 RAM 位址 (018H)

CODE:	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	L	L	H	AC6	AC5	AC4	AC3	AC2	AC1	AC0

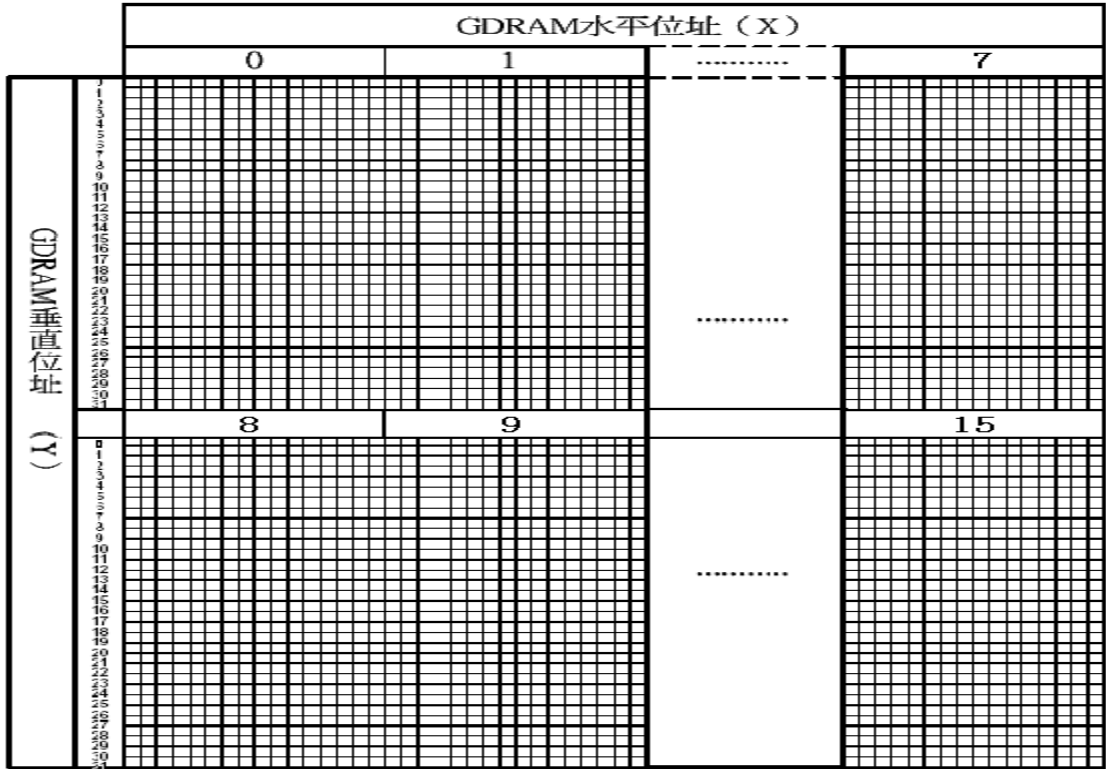
功能: 设定 GDRAM 位址到位址计数器 (AC)

# 六、显示坐标关系

## 1、图形显示坐标

水平方向 X—以字节单位

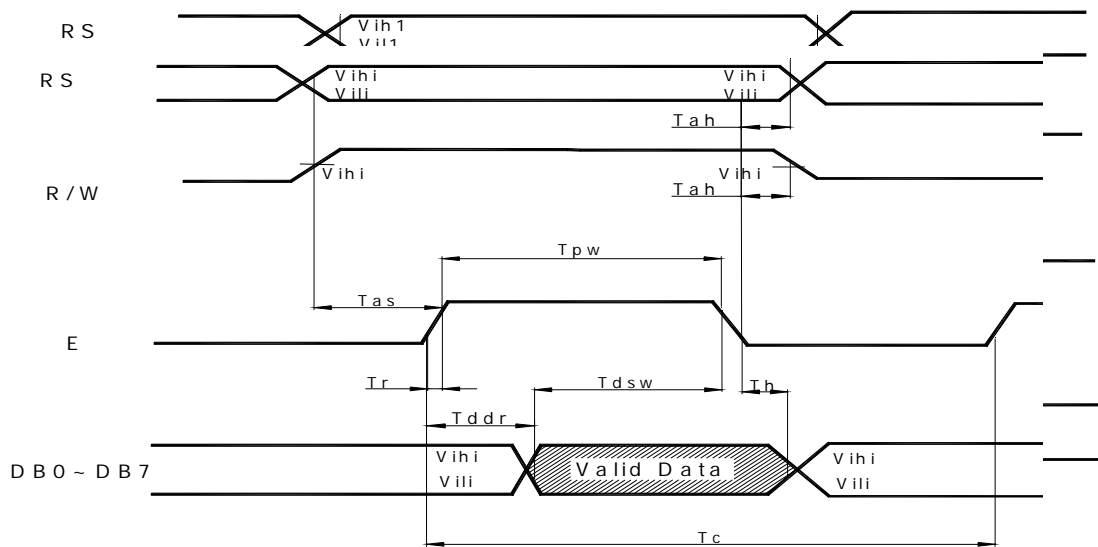
垂直方向 Y—以位为单位



2、汉字显示坐标

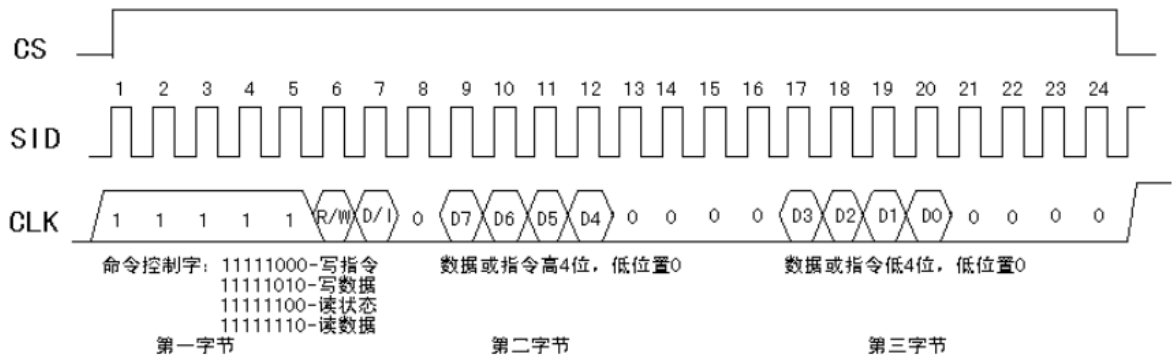
	X 坐标							
Line1	80H	81H	82H	83H	84H	85H	86H	87H
Line2	90H	91H	92H	93H	94H	95H	96H	97H

七、时序图



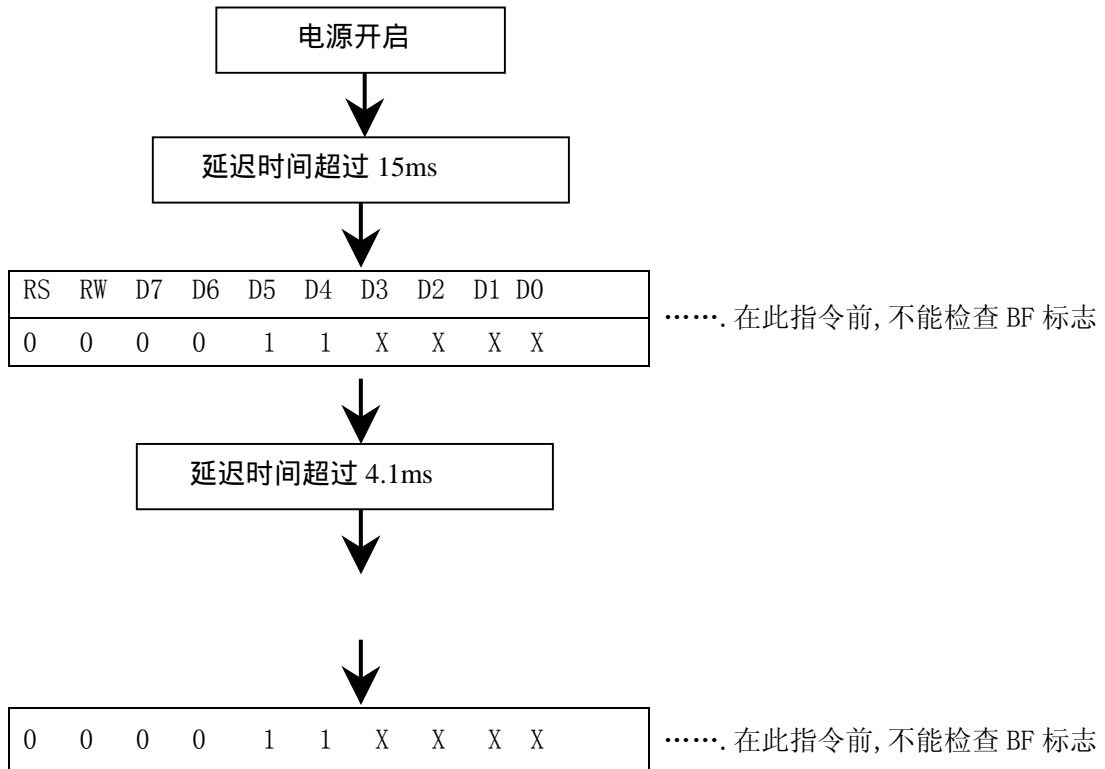
MPU 写数据  
MPU 读数据  
八位元界面时序图

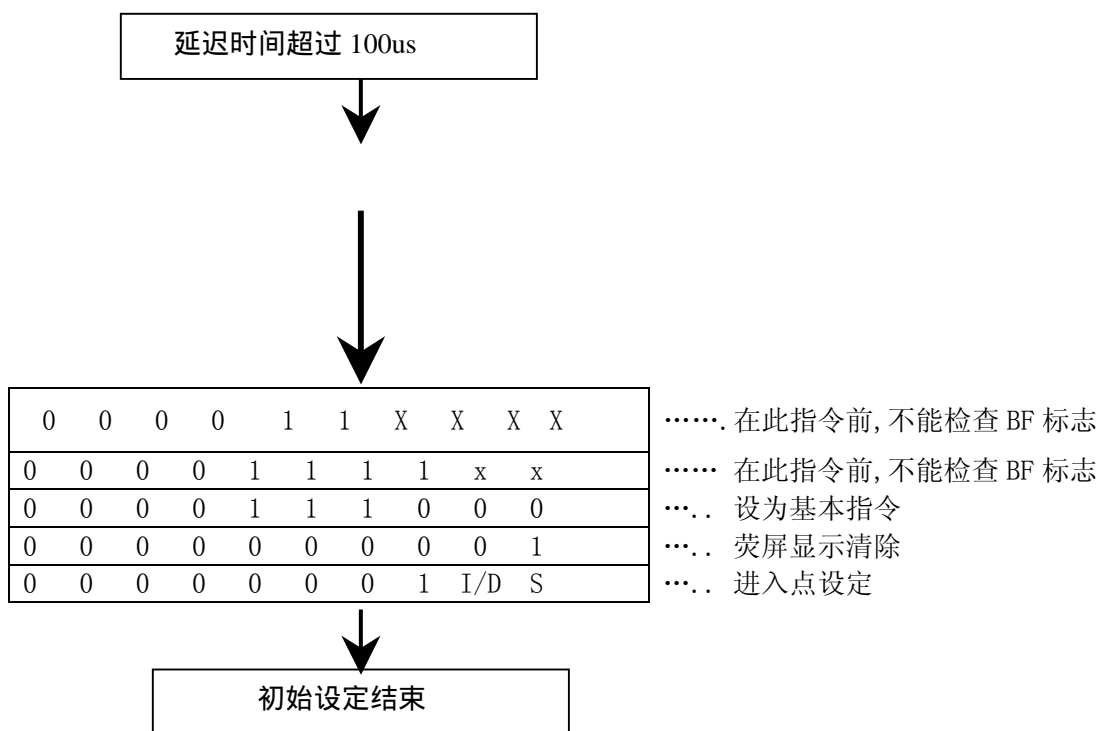
串口读写时序:



## 八、软件初始化:

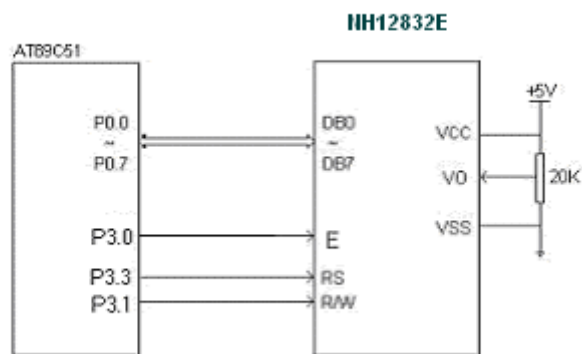
(8 位并行接口)





九. 应用举例:

12832B 与单片机 8031 的一种接口如图 5. 所示



八、附录部分

附录 1: ASCII 码表

☒	☒	☒	♥	♦	♣	♣	•	◐	◑	♂	♀	♫	♫	✳
▶	◀	‡	!!	¶	§	—	‡	†	↓	→	←	⊥	↕	▼
	!	"	#	\$	%	&	'	(	)	*	+	,	-	.
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>
Q	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^
'	a	b	c	d	e	f	g	h	i	j	k	l	m	n
p	q	r	s	t	u	v	w	x	y	z	{		}	~
														△

16x8 半寬字型符號表

附录 2: 汉字码址表





CEA0 魏伪微危韦违桅围唯惟为滩维苇萎委  
CEB0 伟卫伟微危韦违桅围唯惟为滩维苇萎委  
CEC0 卫卫微危韦违桅围唯惟为滩维苇萎委  
CED0 伪伪微危韦违桅围唯惟为滩维苇萎委  
CEE0 微微危韦违桅围唯惟为滩维苇萎委  
CEF0 伪伪微危韦违桅围唯惟为滩维苇萎委  
CFA0 稀稀危韦违桅围唯惟为滩维苇萎委  
CFB0 习习微危韦违桅围唯惟为滩维苇萎委  
CFC0 侠侠微危韦违桅围唯惟为滩维苇萎委  
CFD0 闲闲微危韦违桅围唯惟为滩维苇萎委  
CFE0 相相微危韦违桅围唯惟为滩维苇萎委  
CFF0 橡橡微危韦违桅围唯惟为滩维苇萎委  
DOA0 邪邪微危韦违桅围唯惟为滩维苇萎委  
DOC0 欣欣微危韦违桅围唯惟为滩维苇萎委  
DOD0 行行微危韦违桅围唯惟为滩维苇萎委  
DOE0 朽朽微危韦违桅围唯惟为滩维苇萎委  
DOF0 叙叙微危韦违桅围唯惟为滩维苇萎委  
D1A0 寻寻微危韦违桅围唯惟为滩维苇萎委  
D1C0 牙牙微危韦违桅围唯惟为滩维苇萎委  
D1D0 研研微危韦违桅围唯惟为滩维苇萎委  
D1E0 燕燕微危韦违桅围唯惟为滩维苇萎委  
D1F0 伴伴微危韦违桅围唯惟为滩维苇萎委  
D2A0 野野微危韦违桅围唯惟为滩维苇萎委  
D2B0 依依微危韦违桅围唯惟为滩维苇萎委  
D2C0 倚倚微危韦违桅围唯惟为滩维苇萎委  
D2D0 亦亦微危韦违桅围唯惟为滩维苇萎委  
D2E0 菌菌微危韦违桅围唯惟为滩维苇萎委  
D2F0 茵茵微危韦违桅围唯惟为滩维苇萎委  
D3A0 影影微危韦违桅围唯惟为滩维苇萎委  
D3B0 永永微危韦违桅围唯惟为滩维苇萎委  
D3C0 有余微危韦违桅围唯惟为滩维苇萎委  
D3E0 羽羽微危韦违桅围唯惟为滩维苇萎委  
D3F0 浴浴微危韦违桅围唯惟为滩维苇萎委  
D4A0 园园微危韦违桅围唯惟为滩维苇萎委  
D4B0 岳岳微危韦违桅围唯惟为滩维苇萎委  
D4C0 肚肚微危韦违桅围唯惟为滩维苇萎委  
D4D0 孕孕微危韦违桅围唯惟为滩维苇萎委  
D4E0 贵贵微危韦违桅围唯惟为滩维苇萎委  
D4F0 责责微危韦违桅围唯惟为滩维苇萎委  
D5A0 瞻瞻微危韦违桅围唯惟为滩维苇萎委  
D5B0 统统微危韦违桅围唯惟为滩维苇萎委  
D5C0 招招微危韦违桅围唯惟为滩维苇萎委  
D5D0 昭昭微危韦违桅围唯惟为滩维苇萎委  
D5E0 震震微危韦违桅围唯惟为滩维苇萎委  
D5F0 振振微危韦违桅围唯惟为滩维苇萎委  
D6A0 职职微危韦违桅围唯惟为滩维苇萎委  
D6B0 擲擲微危韦违桅围唯惟为滩维苇萎委  
D6C0 中中微危韦违桅围唯惟为滩维苇萎委  
D6D0 粥粥微危韦违桅围唯惟为滩维苇萎委  
D6E0 逐逐微危韦违桅围唯惟为滩维苇萎委  
D6F0 竹竹微危韦违桅围唯惟为滩维苇萎委  
D7A0 装装微危韦违桅围唯惟为滩维苇萎委  
D7B0 桌桌微危韦违桅围唯惟为滩维苇萎委  
D7C0 仔仔微危韦违桅围唯惟为滩维苇萎委  
D7D0 仔仔微危韦违桅围唯惟为滩维苇萎委  
D7E0 奏奏微危韦违桅围唯惟为滩维苇萎委

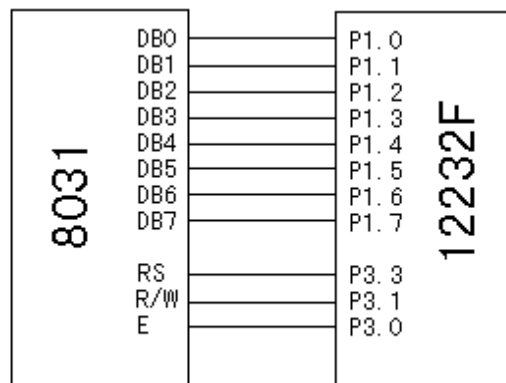
D7F0 尊尊  
D8A0 于于  
D8B0 元元  
D8C0 元元  
D8D0 元元  
D8E0 元元  
D8F0 元元  
D9A0 元元  
D9B0 元元  
D9C0 元元  
D9D0 元元  
D9E0 元元  
D9F0 元元  
DAA0 元元  
DAB0 元元  
DAC0 元元  
DAD0 元元  
DAE0 元元  
DAF0 元元  
DBA0 元元  
DBB0 元元  
DBC0 元元  
DBD0 元元  
DBE0 元元  
DBF0 元元  
DCA0 元元  
DCB0 元元  
DCC0 元元  
DCD0 元元  
DCE0 元元  
DCF0 元元  
DDA0 元元  
ddb0 元元  
DDC0 元元  
DDD0 元元  
DDE0 元元  
DDF0 元元  
DEA0 元元  
DEB0 元元  
DEC0 元元  
DED0 元元  
DEE0 元元  
DEF0 元元  
DFA0 元元  
DFB0 元元  
DFC0 元元  
DFD0 元元  
DFE0 元元  
DFF0 元元  
EOA0 元元  
EOB0 元元  
EOC0 元元  
EOD0 元元  
EOE0 元元  
EOF0 元元  
E1A0 元元  
E1B0 元元  
E1C0 元元  
E1D0 元元





八. 应用举例:

12232F 与 单片机 8031 的 一 种 接 口 如 图 5. 所 示



```
;This program is for 12232F
; RS-----P3.3
; R/W-----P3.1
; E-----P3.0
; DB0~7-----P1
```

```
DI EQU P3.3
RW EQU P3.1
E EQU P3.0
```

```
ORG 0000H
AJMP START
ORG 0003H
LCALL PAUSE
START:
MOV IE, #81H ;EXT. INTO PERMIT
MOV IP, #01H ;INTO IS FIRST INT. LEVEL
MOV TCON, #00H ;TIMER/COUNTER CONTROLER INIT.
mov SP, #67h
LCALL DELAY
LCALL DELAY
LCALL SETUP
LCALL DEF_CHAR
MOV A, #80H
LCALL WRITE_COM
MOV R3, #8
TEST11:
MOV DPTR, #CGRAM1 ;CGRAM TEST
LCALL WRITE_CGRAM
```

```

    DJNZ R3, TEST11
    MOV A, #90H
    LCALL WRITE_COM
    MOV R3, #8
TEST12:
    MOV DPTR, #CGRAM1
    LCALL WRITE_CGRAM
    DJNZ R3, TEST12
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    MOV A, #80H
    LCALL WRITE_COM
    MOV R3, #8
TEST21:
    MOV DPTR, #CGRAM2
    LCALL WRITE_CGRAM
    DJNZ R3, TEST21
    MOV A, #90H
    LCALL WRITE_COM
    MOV R3, #8
TEST22:
    MOV DPTR, #CGRAM2
    LCALL WRITE_CGRAM
    DJNZ R3, TEST22
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    MOV A, #80H
    LCALL WRITE_COM
    MOV R3, #8
TEST31:
    MOV DPTR, #CGRAM3
    LCALL WRITE_CGRAM
    DJNZ R3, TEST31
    MOV A, #90H
    LCALL WRITE_COM
    MOV R3, #8
TEST32:
    MOV DPTR, #CGRAM3
    LCALL WRITE_CGRAM
    DJNZ R3, TEST32
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    MOV A, #80H

```

```

    LCALL WRITE_COM
    MOV R3,#8
TEST41:
    MOV DPTR,#CGRAM4
    LCALL WRITE_CGRAM
    DJNZ R3,TEST41
    MOV A,#90H
    LCALL WRITE_COM
    MOV R3,#8
TEST42:
    MOV DPTR,#CGRAM4
    LCALL WRITE_CGRAM
    DJNZ R3,TEST42
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY

    MOV A#80H      ;WORD TEST
    LCALL WRITE_COM
    MOV DPTR,#CHINESE
    LCALL WRITE_HZ
    MOV A,#90H
    LCALL WRITE_COM
    MOV DPTR,#TABLE1
    LCALL WRITE_ASCII
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    MOV A#80H
    LCALL WRITE_COM
    MOV DPTR,#table1
    LCALL WRITE_ascii
    MOV A,#90H
    LCALL WRITE_COM
    MOV DPTR,#chinese
    LCALL WRITE_hz
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
AAA:    LJMP START

SETUP:
    LCALL DELAY
    LCALL DELAY
    LCALL DELAY
    MOV A,#01H      ;CLEAR DISPLAY

```

```

LCALL WRITE_COM
MOV A,#00110000B ;FUNCTION SETTING
LCALL WRITE_COM
MOV A,#00000010B ;DDRAM SET TO '00H'
LCALL WRITE_COM
MOV A,#00000100B ;
LCALL WRITE_COM
MOV A,#00001100B ;DISPLAY ON
LCALL WRITE_COM
MOV A,#00000001B ;CLEARING SCREEN
LCALL WRITE_COM
MOV A,#10000000B ;SET DDRAM ADDRESS
LCALL WRITE_COM
RET

WRITE_COM: ;WRIT///cv
;WRITE COMMANDS TO ST7920
LCALL DELAY1 ;INSTEAD OF CHECKING BF STATE
CLR RS
CLR RS
CLR RW
CLR RW
MOV P1,A
MOV P1,A
SETB E
SETB E
NOP
NOP
CLR E
CLR E
;LCALL DELAY1
RET

WRITE_DAT: ;WRITE DISPLAY DATAS TO ST79220
LCALL DELAY1
SETB RS
SETB RS
CLR RW
CLR RW
MOV P1,A
MOV P1,A
SETB E
SETB E
NOP
NOP
CLR E
CLR E
RET

DELAY1:
MOV R7,#010H
D11: MOV R6,#010H
DJNZ R6,$

```

```

    DJNZ R7, D11
    RET

DELAY:
    MOV R1, #00H
D2:   MOV R2, #00H
    DJNZ R2, $
    DJNZ R1, D2
    RET

DEF_CHAR:                ;WRITE TO CGRAM
    MOV A, #01000000B    ;SET CGRAM ADDRESS
    LCALL WRITE_COM
    MOV R3, #8
DEF1:
    MOV A, #000H
    LCALL WRITE_DAT
    LCALL WRITE_DAT
    MOV A, #0FFH
    LCALL WRITE_DAT
    LCALL WRITE_DAT
    DJNZ R3, DEF1
    MOV R3, #8
DEF2:
    MOV A, #0AAH
    LCALL WRITE_DAT
    LCALL WRITE_DAT
    MOV A, #0AAH
    LCALL WRITE_DAT
    LCALL WRITE_DAT
    DJNZ R3, DEF2
    MOV R3, #8
DEF3:
    MOV A, #055H
    LCALL WRITE_DAT
    LCALL WRITE_DAT
    MOV A, #0AAH
    LCALL WRITE_DAT
    LCALL WRITE_DAT
    DJNZ R3, DEF3
    mov R3, #8
DEF4:
    MOV A, #0FFH
    LCALL WRITE_DAT
    LCALL WRITE_DAT
    LCALL WRITE_DAT
    LCALL WRITE_DAT
    DJNZ R3, DEF4
    RET
WRITE_ASCII:
    MOV R4, #16
DDDD:  CLR A

```

```

        MOVC A, @A+DPTR
        LCALL WRITE_DAT
        INC DPTR
        DJNZ R4, DDDD
        RET
WRITE_HZ:          ;WRITE 8 CHINESE TO LCD
        MOV R4, #8
DD: CLR A
        MOVC A, @A+DPTR
        INC DPTR
        LCALL WRITE_DAT
        CLR A
        MOVC A, @A+DPTR
        INC DPTR
        LCALL WRITE_DAT
        DJNZ R4, DD
        RET

WRITE_CGRAM:      ;CGRAM TESTING
        CLR A
        MOVC A, @A+DPTR
        LCALL WRITE_DAT
        INC DPTR
        CLR A
        MOVC A, @A+DPTR
        LCALL WRITE_DAT
        RET
PAUSE: SETB P3.2          ;PAUSE KEY PROCESS
        SETB P3.2
        LCALL DELAY1
        MOV C, P3.2
        MOV C, P3.2
        JNC PAUSE        ;CHECK KEY WAS PRESSED
PAUSE1: MOV C, P3.2
        MOV C, P3.2
        LCALL DELAY1
        JC PAUSE1        ;CHECK KEY OPEN AFTER PRESSED
PAUSE2: SETB P3.2
        SETB P3.2
        LCALL DELAY1
        MOV C, P3.2
        MOV C, P3.2
        JNC PAUSE2      ;CHECK KEY WAS PRESSED AGAIN
        RETI

TABLE1:
; “这里是 16*8 点阵的字符代码”
CGRAM1: DB 000H, 000H          ;这里是自造字符地址表
CGRAM2: DB 000H, 002H
CGRAM3: DB 000H, 004H
CGRAM4: DB 000H, 006H
CHINESE:

```

; “这里是 16\*16 点阵的汉字代码表”  
END

以下为串口写指令和数据的子程序:

```
WRITE_COM:
    LCALL DELAY1          ; INSTEAD OF CHECKING BF STATE
    SETB CS
    PUSH ACC
    MOV R0, #8
    MOV A, #11111000B

COMM1:
    CLR C
    RLC A
    MOV SID, C
    CLR CLK
    SETB CLK
    DJNZ R0, COMM1
    POP ACC
    MOV R5, A
    ANL A, #0F0H
    MOV R0, #8

COMM2: CLR C
    RLC A
    MOV SID, C
    CLR CLK
    SETB CLK
    DJNZ R0, COMM2
    MOV A, R5
    SWAP A
    ANL A, #0F0H
    MOV R0, #8

COMM3: CLR C
    RLC A
    MOV SID, C
    CLR CLK
    SETB CLK
    DJNZ R0, COMM3
    CLR CS

    RET

WRITE_DAT:
    LCALL DELAY1
    SETB CS
    PUSH ACC
    MOV R0, #8
    MOV A, #11111010B

DATA1: CLR C
    RLC A
    MOV SID, C
    CLR CLK
    SETB CLK
    DJNZ R0, DATA1
    POP ACC
```

```
        MOV R5, A
        ANL A, #0F0H
        MOV R0, #8
DATA2:  CLR C
        RLC A
        MOV SID, C
        CLR CLK
        SETB CLK
        DJNZ R0, DATA2
        MOV A, R5
        SWAP A
        ANL A, #0F0H
        MOV R0, #8
DATA3:  CLR C
        RLC A
        MOV SID, C
        CLR CLK
        SETB CLK
        DJNZ R0, DATA3
        CLR CS
RET
```