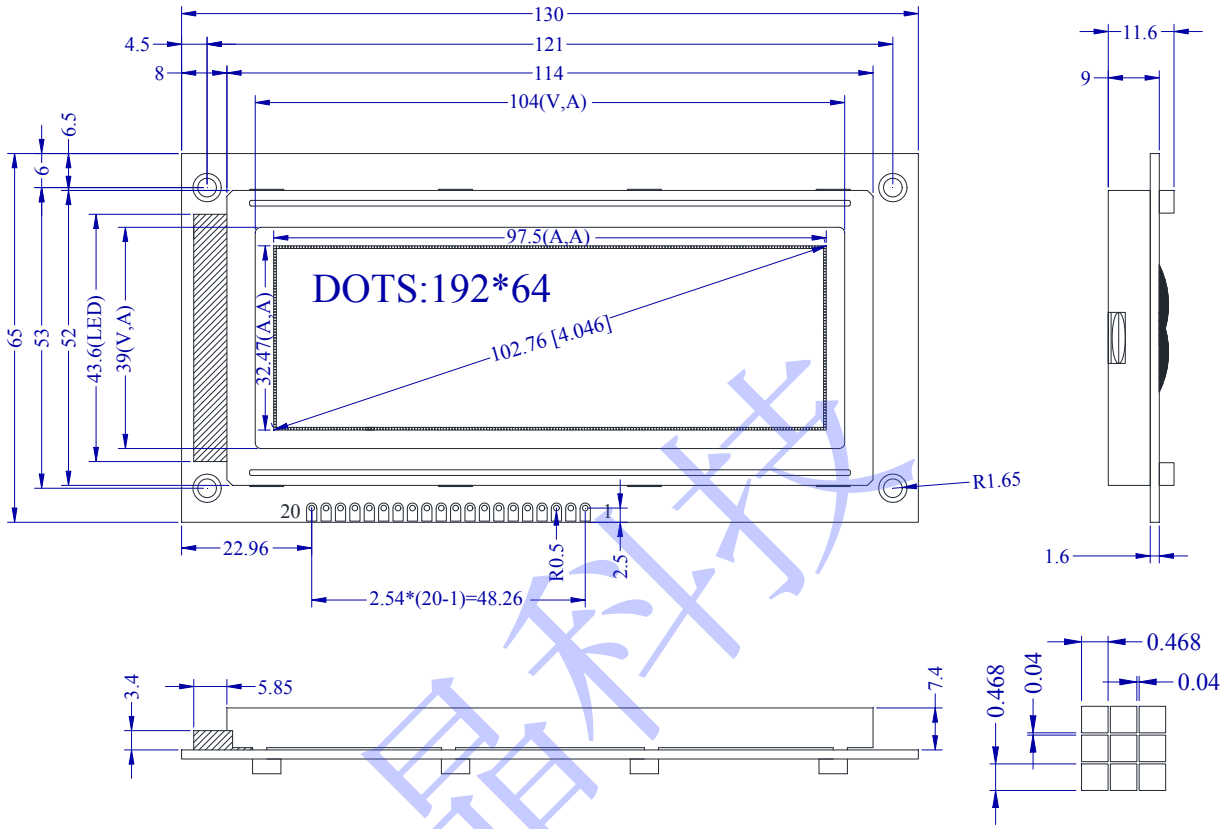
一、模块尺寸



项目	参考值
LCM 尺寸（长×宽×厚）	130.0×65.0×11.6
可视区域（长×宽）	104.0×39.0
点间距（长×宽）	0.468×0.468
点尺寸（长×宽）	0.508×0.508

二、功能介绍

- ✧ 工作电压 3.3V 或者 5.0V（默认出厂为 5.0V，背光 5V 供电）
- ✧ 提供三种与 MPU 通讯方式：4 位、8 位并行，串行通讯（默认出厂未固定，用户自己控制 PSB 来选择）
- ✧ 48x16bit 的字符显示 RAM（LCD 最多为 4 行 8 字）
- ✧ 64x128bit 的绘图 RAM（GDRAM）
- ✧ 2M bit 的中文字库 ROM（CGROM），总共有 8192 个中文字型
- ✧ 16K bit 的半宽字型 ROW（HCGROM），总共有 126 个字母符号字形
- ✧ 64x16 bit 的自定义字符 RAM（CGRAM）
- ✧ 自动上电复位功能
- ✧ 外置复位端（XRESET）
- ✧ 低功率省电设计(除背光 45MA)
  - ■ 正常模式（450uA*2 typ VDD=5V）
  - ■ 待机模式（30uA*2 max VDD=5V）
- ✧ 显示驱动电压 VLCD（V0~VSS）：最大 7V
- ✧ 绘图以及文字画面混合显示功能
- ✧ 多功能指令：
  - ■ 画面清除（display clear） 显示移位（display shift）
  - ■ 光标归零（display clear） 垂直画面旋转（vertical line scroll）
  - ■ 显示开/关（display on/off） 反白显示（by_line reverse display）
  - ■ 光标显示/隐藏（cursor on/off） 待机模式（standby mode）
  - ■ 显示字闪烁（display character blink）
  - ■ 光标移位（cursor shift）
- ✧ 占空比 1/64 偏压比 1/9
- ✧ 最佳可视角为 6 点钟，可以更改
- ✧ 显示颜色效果选黄绿，蓝白，灰白，也可以定制
- ✧ 工作温度，宽温:-20 度到+70 度

三．接口定义

引脚	名称	方向	说明
1	VSS	--	电源负端(0V)
2	VDD	--	电源正端(+3.3V 或+5.0V，默认出厂时设定+5.0V)
3	V0	--	LCD 驱动电压(可调)
4	RST	I	复位脚（低电平有效）
5	RS(CS)	I	并口方式： ● RS=0： 当 MPU 进行读模块操作，指向地址计数器。 当 MPU 进行写模块操作，指向指令寄存器。 ● RS=1： 无论 MPU 读/写操作，均指向数据寄存器。 串口方式： CS：串行片选信号，高电平有效。
6	R/W(SID)	I	并口方式： ● R/W=0 写操作。 ● R/W=1 读操作。 串口方式： SID：串行数据输入端
7	E1(SCLK)	I	（上半屏）并口方式：使能信号，高电平有效。 串口方式：SCLK 串行时钟信号。
8	E2(SCLK)	I	（下半屏）并口方式：使能信号，高电平有效。 串口方式：SCLK 串行时钟信号。
9	PSB	I	串/并口控制选择端：（内部 P、S 可选）
10-17	DB0 ~ DB7	I/O	MPU 与模块之间并口的数据传送通道， 4 位总线模式下 D0 ~ D3 脚断开

18	A	--	空脚（接 VDD）/或者背光正极
19	K	--	空脚(接 VSS)/或者背光负极
20	VOUT	--	倍压输出脚。（VDD＝+3.3V 时有效）

四．电性参数(直流)

名称	符号	测试条件	参数范围			单位
			最小	标准	最大	
模块工作电压	VDD	-	4.5/3.1	5.0/3.3	5.5/3.5	V
玻璃电压	V0	V0-VDD	4.5	5.0	7.0	V
背光工作电压	VLED	-	4.5/3.1	5.0/3.3	5.5/3.5	V
IO 输入高电平	VIH	-	0.7VDD	-	VDD	V
IO 输入低电平	VIL	-	-	-	1.0	V
LCM 输出高电平	VOH	-	0.8VDD	-	VDD	V
LCM 输出低电平	VOL	-	-	-	0.1VDD	V
模块工作电流	IDD	=VDD	-	-	1.0	MA
模块待机电流	IDO	=VDD	-	-	20	uA
背光工作电流	ILED	=VLED	30	45	60	MA

五．显示原理

DDRAM 地址与 192*64 点阵显示屏的关系（可以显示 12*4=48 个汉字）一行最多 16 字

表 1

Y X		192 点																			
		H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L
E1	32点	80		81		82		83		84		85		86		87		88		89		8A		8B	
		90		91		92		93		94		95		96		97		98		99		9A		9B	
E	3	80		81		82		83		84		85		86		87		88		89		8A		8B	

2	2点	90	91	92	93	94	95	96	97	98	99	9A	9B

80 表示 DDRAM 地址，80 代表一个显示汉字的地址，占 16*16 点阵，只要在 80 地址内写入汉字内码就能显示你的汉字或者字符， 汉字内码有两个字节，高位（H）写在前，低位（L）写在后，注意地址会自动加 1。

现在的单片机编译软件都能引用汉字编译出汉字内码，19264 有一套标准的 GB2312 简体字库，收到汉字内码直接调取内部对应点阵显示在屏幕上。

HJ19264ZWA-1 分上半屏和下半屏，用 E1/E2 或者 CS 来选择上屏或者下屏

GDRAM 与 19264 点阵的关系（192*64 点阵的绘图像素）

表 2

Y点 X点		192点																					
		H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H	L
		0		1		2		3		4		5		6		7		8		9		10		11	
E1	0	0/0		1/0		2/0		3/0		4/0		5/0		6/0		7/0		8/0		9/0		10/0		11/0	
	1	1/1		1/1		2/1		3/1		4/1		5/1		5/1		7/1		8/1		9/1		10/1		11/1	
	2	0/2		1/2		2/2		3/2		4/2		5/2		6/2		7/2		8/2		9/2		10/2		11/2	
	``	```		```		```		```		```		```		```		```		```		```		```		```	
	31	0/31		1/31		2/31		3/31		4/31		5/31		6/31		7/31		8/31		9/31		10/31		11/31	
E2	0	0/0		1/0		2/0		3/0		4/0		5/0		6/0		7/0		8/0		9/0		10/0		11/0	
	1	1/1		1/1		2/1		3/1		4/1		5/1		5/1		7/1		8/1		9/1		10/1		11/1	
	2	0/2		1/2		2/2		3/2		4/2		5/2		6/2		7/2		8/2		9/2		10/2		11/2	
	``	```		```		```		```		```		```		```		```		```		```		```		```	
	31	0/31		1/31		2/31		3/31		4/31		5/31		6/31		7/31		8/31		9/31		10/31		11/31	

位=点	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
字节	H								L							
地址	80h															

GDRAM 与 19264 点阵的分布图，程序写入过程为，先写入垂直地址，例如 E1（0-32）再写水平 X 地址，例如；（0-11），一个水平地址有 1*16 位点阵（分两字节，先写高字节，再写低字节，高位在左边），水平地址可以自动加 1，

这里 0 地址为 0X80，在写地址的时候都要加上 0X80；举例，如果是在第 2 行的第 33 列，垂直地址为 0X80+1，水平地址为 0X+2，

要调一幅图片时，使用横向取模取出点阵数据， 你也可以自编图形， 在最后的程序例程中，编写了双点阵的边框供你参考

写显示数据流程

a) CGRAM

自编字符 CGRAM

模块预留了 1024b 自编字符空间（可以写汉字 4 个），编写一些特殊的字符或者符号.

写入地址范围为 0x40-0x7F, 每一个地址可以放两个字节，

自编字库的调用地址为 00-07H 范围；

自编地址，调用地址显示关系 如下

自编地址	调用地址
0x40；	0x00,0x00；
0x50；	0x00,0x02；
0x60；	0x00,0x04；
0x70	0x00,0x06；

假设要自编一个汉字，

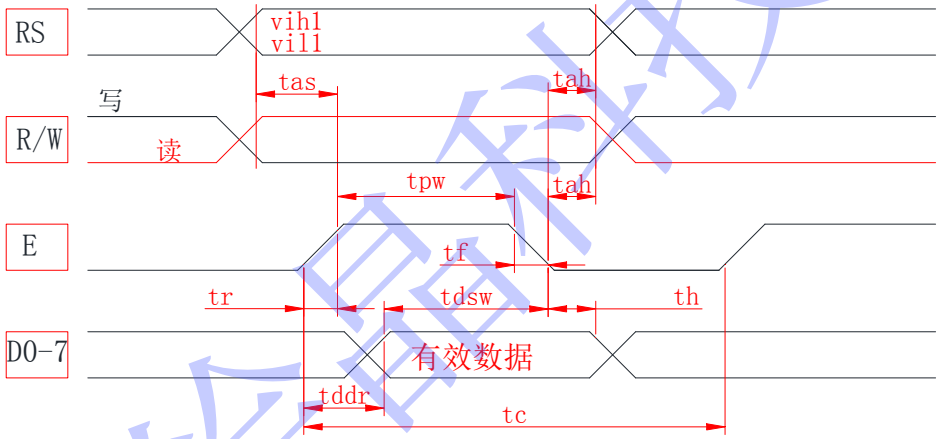
0x40~0x4f 有 16 行，每行可以放入 16 点，就完成一个汉字 16*16 写入要显示出自编汉字，根据对应的关系，写地址 0x00,0x00；

b) HCGROM

内部的字符

六．时序图

并口时序



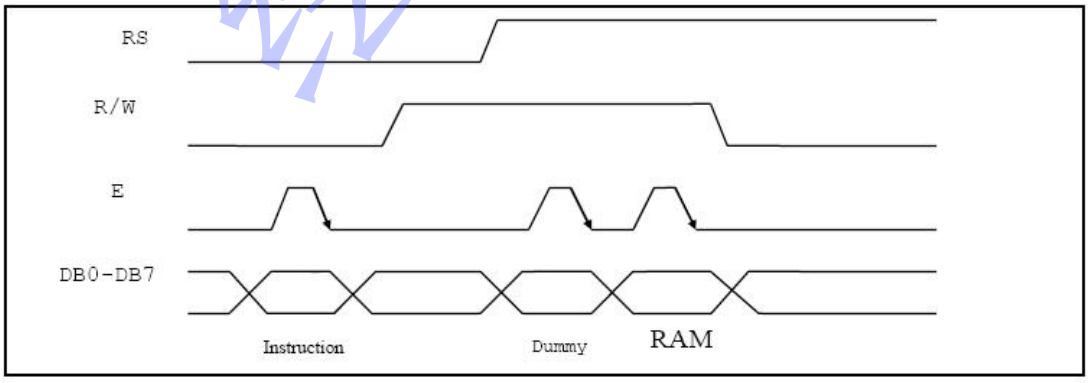
并口模式常温（环境25度，VDD=4.5V 或者 VDD=2.7V)

参数	符号	测试条件	最小	典型	最大	单位
内部时钟						
振荡频率	Fosc	R=33kr/R=18kr	480/470	540/530	600/590	Khz
外部时钟						
外部频率	fex	-	480/470	540/530	600/590	Khz
占空比			45	50	55	%
上升/下降时间	Tr,tf	-	-	-	0.2	us
写模式（从单片机写数据到模块）						
E周期	Tc	E	1200/1800	-	-	Ns
E脉冲宽度	Tpw	E	140/160	-	-	Ns

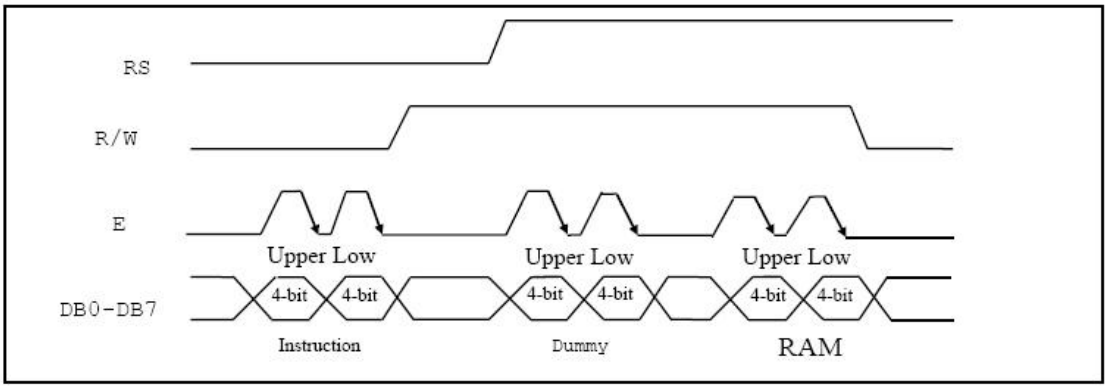
E上升/下降时间	Tr,tf	E	-	-	25	Ns
地址建立时间	Tas	Rs,r/w,e	10	-	-	Ns
地址保持时间	Tah	Rs,r/w,e	20	-	-	Ns
数据建立时间	Tdsw	D0-d7	40	-	-	Ns
数据保持时间	Th	D0-d7	20	-	-	Ns
读模式（从模块读数据到单片机）						
E周期	Tc	E	1200/1800	-	-	Ns
E脉冲宽度	Tpw	E	140/320	-	-	Ns
E上升/下降时间	Tr,tf	E	-	-	25	Ns
地址建立时间	Tas	Rs,r/w,e	10	-	-	Ns
地址保持时间	Tah	Rs,r/w,e	20	-	-	Ns
数据建立时间	Tddr	D0-d7	-	-	100/260	Ns
数据保持时间	Th	D0-d7	20	-	-	Ns

并口传输方式：

当 PSB 脚（串/并口选择）接高电平时，模块将进入并口模式，在并口模式下可由指令 DL FLAG 来选择 8-位或 4-位接口，主控制系统将配合(RS、RW、E、DB0..DB7)来达成传输动作。从一个完整的流程来看，当设定地址指令后(CGRAM、DDRAM)若要读取数据时需先 DUMMY READ 一次，才会读取到正确数据第二次读取时则不需 DUMMY READ 除非又下设定地址指令才需再次 DUMMY READ。在 4-位传输模式中，每一个八位的指令或数据都将被分为两个字节动作：较高 4(DB7~DB4)的资料将会被放在第一个字节（DB7~DB4）部分，而较低 4 位（DB3~DB0）的资料则会被放在第二个字节的（DB7~DB4）部分，至于相关的另四位则在 4-位传输模式中 DB3~DB0 接口未使用。相关接口传输讯号请参考下图说明：

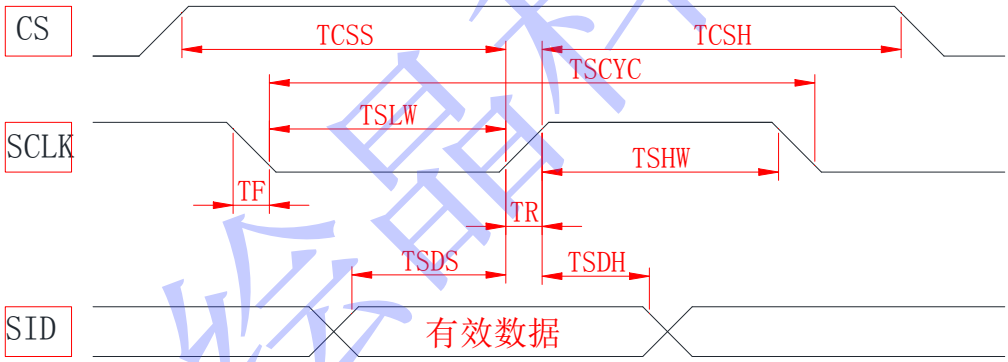


Timing Diagram of 8-bit Parallel Bus Mode Data Transfer



Timing Diagram of 4-bit Parallel Bus Mode Data Transfer

串口时序



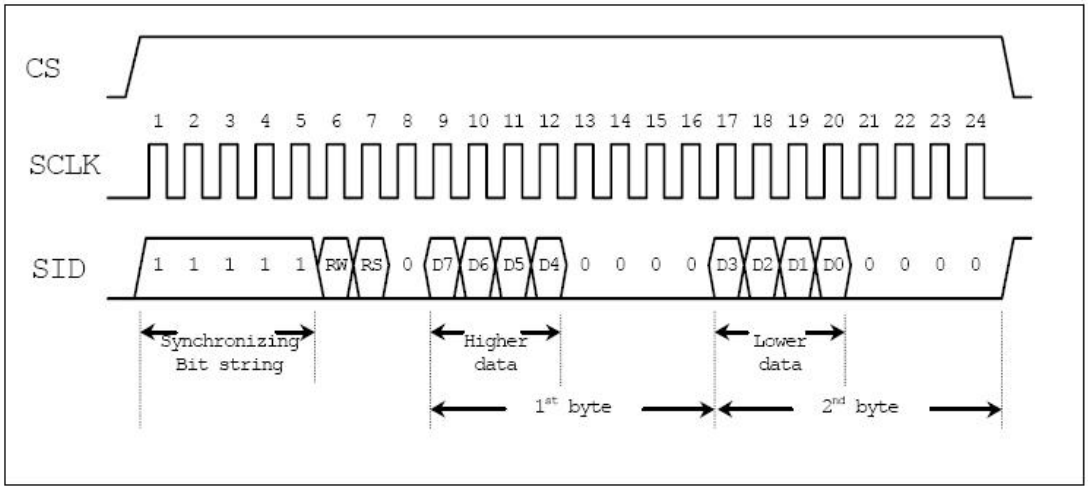
串口模式常温（环境25度，VDD=4.5V 或者 VDD=2.7V)

参数	符号	测试条件	最小	典型	最大	单位
内部时钟						
振荡频率	Fosc	R=33kr/R=18kr	470	530	590	Khz
外部时钟						
外部频率	fex	-	470	530	590	Khz
占空比			45	50	55	%
上升/下降时间	Tr,tf	-	-	-	0.2	us
写模式（从单片机写数据到模块）						
串行时钟周期	TSCYC	E/SCLK	400/600	-	-	Ns

SCLK高脉宽	TSHW	E/SCLK	200/300	-	-	Ns
SCLK脉宽	TSLW	E/SCLK	200/300	-	-	Ns
SID数据建立时间	TSDS	RW/SID	40	-	-	Ns
SID数据保持时间	TSDH	RW/SID	40	-	-	Ns
CS建立时间	TCSS	RS/CS	60	-	-	Ns
CS保持时间	TCSH	RS/CS	60	-	-	Ns

串口传输方式：

当PSB脚（串/并口选择）接低电位时，模块将进入串口模式。从一个完整的串口传输流程来看，一开始先传输启始字节，它需先接收到五个连续的‘1’（同步位字符串），在启始字节，此时传输计数将被重置并且串行传输将被同步，再跟随的两个位字符串分别指定传输方向位（RW）及寄存器选择位（RS），最后第八的位则为‘0’。在接收到同步位及RW和RS资料的启始字节后，每一个八位的指令将被分为两个字节接收到：较高4位（DB7~DB4）的指令资料将会被放在第一个字节的LSB部分，而较低4位（DB3~DB0）的指令资料则会被放在第二个字节的LSB部分，至于相关的另四位则都为0。串行传输讯号请参考下图说明



Timing Diagram of Serial Mode Data Transfer

七．指令说明

1、**指令表：**（RE=0：基本指令集）

指令	指令码										HEX	说明	执行时间（540KHZ）
	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0			
清除显示	0	0	0	0	0	0	0	0	0	1	0X01	将DDRAM填满”20H”，并且设定DDRAM的地址计数器（AC）到“00H”	1.6ms
地址归零	0	0	0	0	0	0	0	0	1	X	0X02	设定DDRAM的地址计数器（AC）到”00H”，并且将光标移到开头原点位置，这个指令并不改变DDRAM的内容	72us

输入点设定	0	0	0	0	0	0	0	1	I/D	S	0X4X	指定在数据的读取与写入时，设定光标的移动方向及指定显示的移位	72us
显示状态开/关	0	0	0	0	0	0	1	D	C	B	0x8x	D=1；整体显示ON C=1；光标ON B=1；光标位置反白ON	72us
光标或显示移位控制	0	0	0	0	0	1	S/C	R/L	X	X	0x1x	设定光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	72us
功能设定	0	0	0	0	1	DL	X	ORE	X	X	0X2X	DL=1；8位控制模式 DL=0；4位控制模式 RE=1；选择扩展指令集 RE=0；选择基本指令集	72us
设定CGRAM地址	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	0X4X	设定CGRAM地址到地址计数器（AC） 需确认扩展指令中SR=0（卷动地址或RAM地址选择	72us
设定DDRAM地址	0	0	1	OAC6	AC5	AC4	AC3	AC2	AC1	AC0	0X8X	设定CGRAM地址到地址计数器（AC） AC6固定为0	72us
读取忙碌标志（BF）和地址	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0		读取忙碌标志（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	0us
写数据到RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0		写入数据到内部RAM（DDRAM/CGRAM/GDRAM）	72us
读出RAM的数据	1	0	D7	D6	D5	D4	D3	D2	D1	D0		从内部RAM读取数据（DDRAM/CGRAM/GDRAM）	72us

指令表二：（RE=1：扩充指令集）

指令	指令码										HEX	说明	执行时间（540KHZ）
	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0			
待机模式	0	0	0	0	0	0	0	0	0	1	0X01	进入待机模式，执行任何其它指令都可终止待机	72us

												模式（COM1~32停止动作）	
卷动地址或RAM地址选择	0	0	0	0	0	0	0	0	1	SR	0X02	SR=1；允许输入垂直卷动地址SR=0；允许设定CGRAM地址（基本指令）	72us
反白选择	0	0	0	0	0	0	0	1	R1	R0	0X4X	选择4行中的任一行反白显示，并可决定反白与否R1，R0初什为‘00，当第一次设定时为反白显示，再一次设定时为正常显示	72us
扩充功能设定	0	0	0	0	1	DL	X	1RE	G	0	0X3X	DL=1；8位控制模式 DL=0；4位控制模式 RE=1；选择扩展指令集 RE=0；选择基本指令集 G=1；绘图显示ON G=0；绘图显示OFF	72us
设定IRAM地址或卷动地址	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	0X4X	SR=1：AC5~AC0为垂直卷动地址	72us
设定绘图RAM地址	0	0	1	00	0AC5	0AC4	AC3 AC3	AC2 AC2	AC1A C1	AC0 AC0	0X8X	设定（GDRAM地址到地址计数器（AC） 先设垂直地址再设水平地址（连续写入两个字节的坐标地址） 垂直地址范围AC5~AC0 水平地址范围AC3~AC0	2us

2、具体指令介绍：

基本指令（RE＝0）

1） 清除显示：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	0	0	0	0	0	0	1	0x01

功能：将DDRAM 填满”20H”(空格)，把DDRAM 地址计数器调整“00H”，重新进入点设定将I/D设为”1”，光标右移AC 加1。

2） 地址归位：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	0	0	0	0	0	1	x	0x02

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

3） 输入点设置：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	0	0	0	0	1	I/D	S	0x04

功能：设定光标移动方向并指定整体显示是否移动。
I/D=1 光标右移，AC自动加1； I/D=0 光标左移，AC自动减1。
S=1 且DDRAM为写状态：整体显示移动，方向由I/D决定(I/D=1左移，I/D=0右移)
S=0 或DDRAM 为读状态： 整体显示不移动.

4） 显示状态开/关：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	0	0	0	1	D	C	B	0x08

功能：D=1： 整体显示ON ； D=0： 整体显示OFF。
C=1： 光标显示ON ； C=0： 光标显示OFF。
B=1： 光标位置反白且闪烁 ； B=0： 光标位置不反白闪烁。

5） 光标或显示移位控制：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	0	0	1	S/C	R/L	X	x	0x1x

功能：S/C： 光标左/右移动，AC减/加1。

R/L：整体显示左/右移动，光标跟随移动，AC值不变。

S/C	R/L	说明	AC值
L	L	光标向左移动	AC=AC-1
L	H	光标向右移动	AC=AC+1
H	L	显示向左移动，且光标跟着移动	AC=AC
H	H	显示向右移动，且光标跟着移动	AC=AC

6） 功能设定：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	0	1	DL	X	RE	X	x	0x2x

功能：DL=1： 8-BIT 控制接口； DL=0： 4-BIT 控制接口。
RE=1： 扩充指令集动作； RE=0： 基本指令集动作。

7） 设定CGRAM地址：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	0x4x

功能：设定CGRAM地址到地址计数器（AC），需确定扩充指令中SR=0(卷动地址或RAM地址选择)

8） 设定DDRAM地址：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	0x8x

功能：设定DDRAM地址到地址计数器（AC）

9） 读取忙碌状态（BF）和地址：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	0xXx

功能：读取忙碌状态（BF）可以确认内部动作是否完成，同时可以读出地

址计数器（AC）的值，当BF=1，表示内部忙碌中此时不可下指令需等BF=0才可下新指令

10） 写资料到RAM：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	1	0	D7	D6	D5	D4	D3	D2	D1	D0	0xXx

功能：写入资料到内部的RAM（DDRAM/CGRAM/GDRAM），每个RAM地址都要连续写入两个字节的资料。

11） 读RAM的值：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	1	1	D7	D6	D5	D4	D3	D2	D1	D0	0xXx

功能：从内部RAM 读取数据（DDRAM/CGRAM/GDRAM），当设定地址指令后，若需读取数据时需先执行一次空的读数据，才会读取到正确数据，第二次读取时则不需要，除非又下设定地址指令。

扩充指令（RE＝1）

1） 待命模式：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	0	0	0	0	0	0	1	0x1x

功能：进入待命模式，执行其它命令都可终止待命模式

2） 卷动地址或RAM地址选：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	0	0	0	0	0	1	SR	0x2x

功能：SR=1：允许输入卷动地址；
SR=0：允许设定CGRAM地址（基本指令）

3） 反白选择：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	0	0	0	0	1	R1	R0	0x4x

功能：选择4 行中的任一行作反白显示，并可决定反白与否。第一次设定为反白显示，再次设定时为正常显示

19264	R1	R0	行地址参数
上屏	L	L	第一行反白或正常显示
	L	H	第二行反白或正常显示
下屏	L	L	第三行反白或正常显示
	L	H	第四行反白或正常显示

4）睡眠模式：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	0	0	0	1	SL	X	X	0x08

功能：SL=1：脱离睡眠模式； SL=0：进入睡眠模式。

5）扩充功能设定：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	0	1	DL	X	RE	G	X	0x3x

功能：DL=1：8-BIT 控制接口； DL=0：4-BIT 控制接口

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

6）卷动地址设定：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	0x4x

功能：SR=1，AC5~AC0 为垂直卷动地址

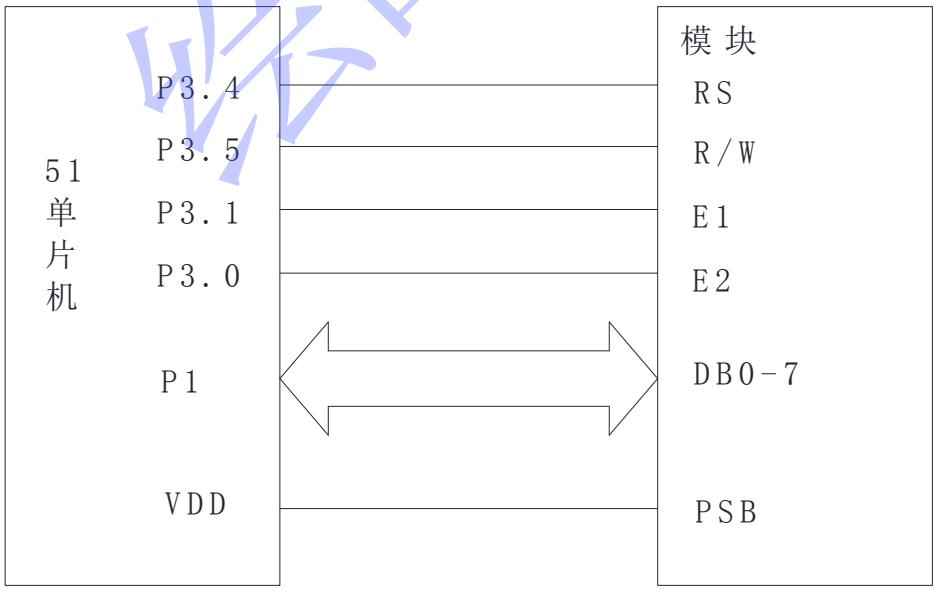
7）绘图RAM地址设定：

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	0x8x

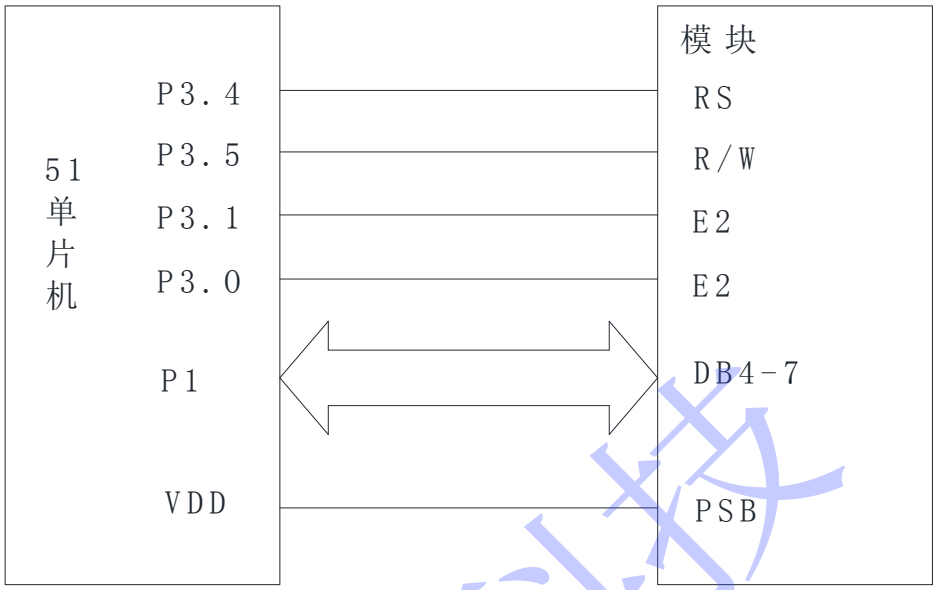
功能：设定GDRAM地址到地址计数器（AC）

八、单片机接线图

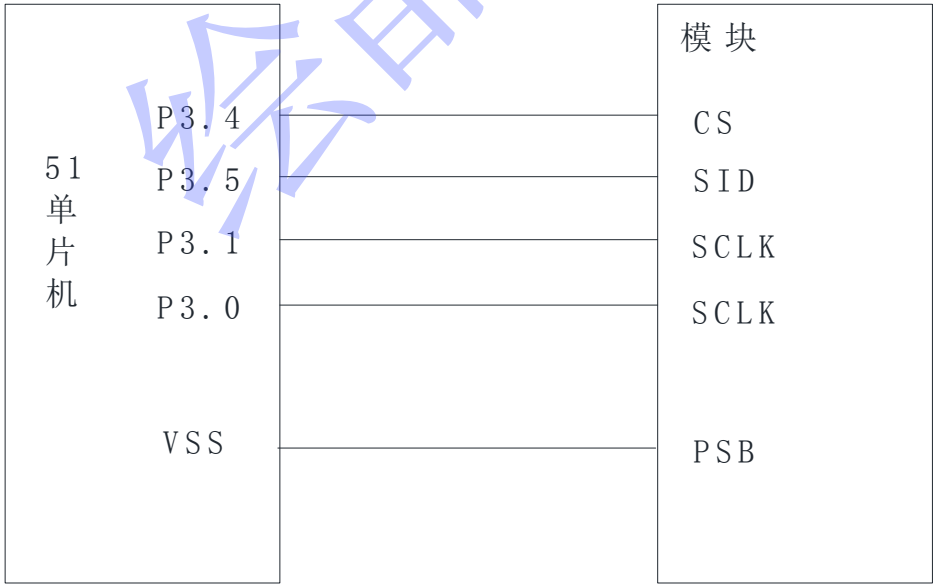
1）、8位并联接口：



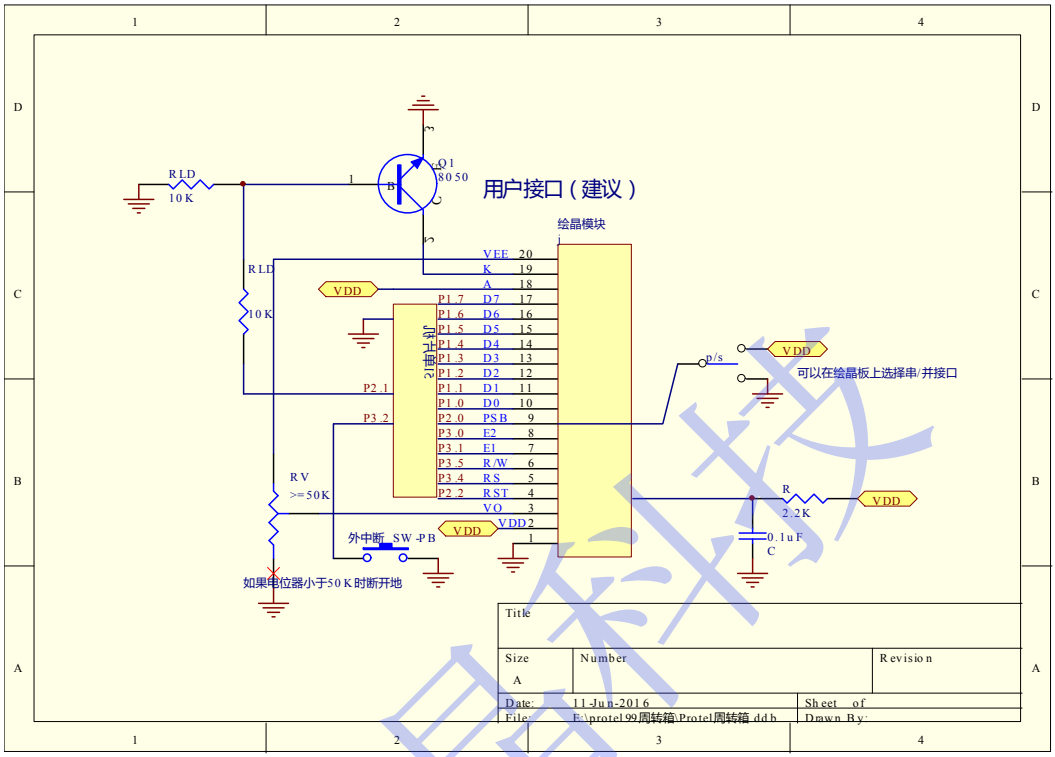
2）、4位并联接口：



3）、串行联接口：



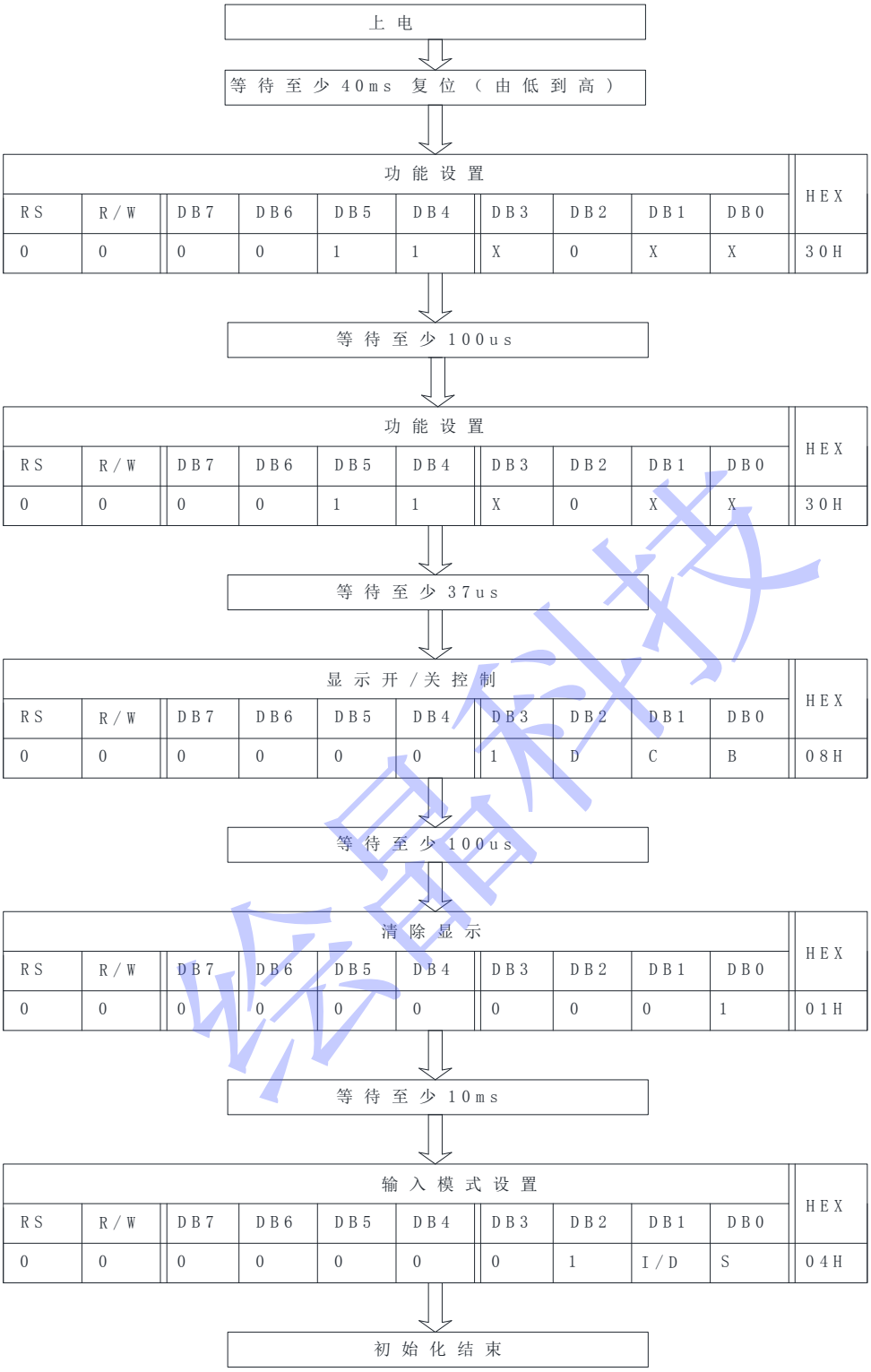
开发详细的接线方法



九、初始化程序

a) 在写程序之前都要先对模块的硬件软件初始化，

8位接口模式


上 电
等 待 至 少 40ms 复 位 （ 由 低 到 高 ）
功 能 设 置
RS R/W DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0 HEX
0 0 0 0 1 1 X 0 X X 30H
等 待 至 少 100us
功 能 设 置
RS R/W DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0 HEX
0 0 0 0 1 1 X 0 X X 30H
等 待 至 少 37us
显 示 开 / 关 控 制
RS R/W DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0 HEX
0 0 0 0 0 0 1 D C B 08H
等 待 至 少 100us
清 除 显 示
RS R/W DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0 HEX
0 0 0 0 0 0 0 0 0 1 01H
等 待 至 少 10ms
输 入 模 式 设 置
RS R/W DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0 HEX
0 0 0 0 0 0 0 1 I/D S 04H
初 始 化 结 束

4位接口模式


上 电
等 待 至 少 40ms 复 位 由 低 到 高
功 能 设 置
RS R/W DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0 HEX
0 0 0 0 1 0 X X X X 20H
等 待 至 少 100us
功 能 设 置
RS R/W DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0 HEX
0 0 0 0 1 0 X X X X 20H
0 0 x 0 x x X X X X
等 待 至 少 100us
显 示 开 / 关 控 制
RS R/W DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0 HEX
0 0 0 0 0 0 X X X X 00H
0 0 1 D C B X X X X 80H
等 待 至 少 100us
清 除 显 示
RS R/W DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0 HEX
0 0 0 0 0 0 X X X X 00H
0 0 0 0 0 1 X X X X 10H
等 待 至 少 10ms
输 入 模 式 设 置
RS R/W DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0 HEX
0 0 0 0 0 0 X X X X 00H
0 0 0 1 I/D S X X X X 20H
初 始 化 结 束

十、程序例程

```
//深圳绘晶科技有限公司19264中文字库系列,单片机:89S52, 晶振:12M,
//并口连接方式,

#include<reg52.h>
#include <intrins.h>
sbit REST=P2^1;                          //4
sbit RS=P3^4;//串口时为CS--              //5
sbit RW=P3^5; //串口为SID            //6
sbit E1=P3^1;  //串口为时钟SCLK          //7
sbit E2=P3^0;  //串口为时钟SCLK          //8
sbit PSB=P2^0;//并口时, PSB=1;串口时, PSB=0   //9
sbit stop=P3^2;

typedef unsigned int Uint;
typedef unsigned char Uchar;

Uchar m1,num,ii,z,z1,d,d1,s,s1,s10,s100;
Uchar code  num_8x16[11][16];//宋体12
Uchar code  num_16x32[11][62];//DS-Digital24
Uchar code  num_24x48[11][144];//宋体36

//这个是在串口时指令和数据之间的延时
/*
void delay10US(Uchar x)
{
   Uchar k;
 for(k=0;k<x;k++);
 }
*/
const Uchar delay=250;  //延时时间常数
static void Wait1ms(void)//延迟1 ms
{
  Uchar cnt=0;
  while (cnt<delay) cnt++;
}
```

```
//延迟n ms
void WaitNms(Uint n)
{
    Uint i;
    for(i=1;i<=n;i++)
    Wait1ms();
}

void time_nms(Uint x)//0.5ms
{
    Uchar j;
 while(x--)
  {for(j=0;j<50;j++)
    {;}
  }
}




//*****************************************************************************//
//以下是并口时才开的
//读忙标志,

void RDBF(void)
{
    Uchar temp;
    RS=0;    // RS=0
    RW=1;    // RW=1
    while(1)
    {
    P1=0xFF;       //数据线为输入
    E1=1;
    E2=1;
    temp=P1;
    E1=0;    // E=0
    E2=0;    // E=0
    if ((temp&0x80)==0) break;
    }
}
//上屏写数据到指令寄存器
```

```
void WRCommandH(Uchar comm)
{
    RDBF();
    RW=0;
    P1=comm;
    E2=0;
    E1=1;
    E1=0;
}
//下屏写数据到指令寄存器

void WRCommandL(Uchar comm)
{
    RDBF();
    RW=0;
    P1=comm;
    E1=0;
    E2=1;
    E2=0;
}
//上屏写数据到数据寄存器
void WRDataH(Uchar TEMP)
{
    RDBF();
    RS=1;
    RW=0;
    P1=TEMP;
    E2=0;
    E1=1;
    E1=0;

}
//写数据到数据寄存器
void WRDataL(Uchar TEMP)
{
    RDBF();
    RS=1;
    RW=0;
    P1=TEMP;
    E1=0;
    E2=1;
```

```
    E2=0;

}

////////////////////////////////////////////////////////////////////////////
//以下是串口时开的读写时序

/*
void SendByteLCDH(Uchar WLCDData)
{
 Uchar i;
 for(i=0;i<8;i++)
 {
 if((WLCDData<<i)&0x80)RW=1;
 else RW=0;
 E1=0;
 E1=1 ;
 }
}


 SPIWH(Uchar Wdata,Uchar WRS)
 {
   SendByteLCDH(0xf8+(WRS<<1));//寄存器选择WRS
   SendByteLCDH(Wdata&0xf0);
   SendByteLCDH((Wdata<<4)&0xf0);
 }


void WRCommandH(Uchar CMD)
{
  RS=0;
  RS=1;
  SPIWH(CMD,0);
  delay10US(90);//89S52来模拟串行通信,所以,加上89S52的延时,
}



void WRDataH(Uchar Data)
{
```

```
  RS=0;
  RS=1;
  SPIWH(Data,1);

}


void SendByteLCDL(Uchar WLCDData)
{
 Uchar i;
 for(i=0;i<8;i++)
 {
 if((WLCDData<<i)&0x80)RW=1;
 else RW=0;
 E2=0;
 E2=1 ;
 }
}


 SPIWL(Uchar Wdata,Uchar WRS)
 {
   SendByteLCDL(0xf8+(WRS<<1));
   SendByteLCDL(Wdata&0xf0);
   SendByteLCDL((Wdata<<4)&0xf0);
 }


void WRCommandL(Uchar CMD)
{
  RS=0;
  RS=1;
  SPIWL(CMD,0);
  delay10US(90);//89S52来模拟串行通信,所以,加上89S52的延时,
}




void WRDataL(Uchar Data)
{
```

```
  RS=0;
  RS=1;
  SPIWL(Data,1);

}
*/

/*********************************************************************/
//初始化LCD-8位接口
void LCDInit(void)
{   //PSB=0; //串口
    PSB=1;//并口时选这个,上一行取消
    REST=1;
    REST=0;
    REST=1;

    WRCommandH(0x30);     //基本指令集,8位并行
    WRCommandL(0x30);     //基本指令集,8位并行

    WRCommandH(0x06);     //启始点设定：光标右移
    WRCommandL(0x06);     //启始点设定：光标右移

    WRCommandH(0x01);     //清除显示DDRAM
    WRCommandL(0x01);     //清除显示DDRAM

    WRCommandH(0x0C);     //显示状态开关：整体显示开，光标显示关，光标显示反白关
    WRCommandL(0x0C);     //显示状态开关：整体显示开，光标显示关，光标显示反白关

    WRCommandH(0x02);     //地址归零
    WRCommandL(0x02);     //地址归零

}

void ShowNUMCharH(Uchar addr,Uchar i,Uchar count)
{
     Uchar j;
    for(j=0;j<count;)
    {
        WRCommandH(addr);     //设定DDRAM地址
        WRDataH(i+j);//必为两个16*8位字符拼成一个16*16才能显示
        j++;
```

```
        WRDataH(i+j);
        addr++;
        j++;
    }
}
//下半屏显示连续字串(半宽字符)
void ShowNUMCharL(Uchar addr,Uchar i,Uchar count)
{
     Uchar j;
    for(j=0;j<count;)
    {
        WRCommandL(addr);     //设定DDRAM地址
        WRDataL(i+j);
        j++;
        WRDataL(i+j);
        addr++;
        j++;
    }
}

void WRCGRAM(Uchar data1,Uchar data2,Uchar addr)
{
      Uchar i;
      for(i=0;i<16;)
  {
      WRCommandH(addr+i); WRCommandL(addr+i);      //设定CGRAM地址
      WRDataH(data1); WRDataL(data1);
      WRDataH(data1); WRDataL(data1);
      i++;
      WRCommandH(addr+i);  WRCommandL(addr+i);     //设定CGRAM地址
      WRDataH(data2); WRDataL(data2);
      WRDataH(data2); WRDataL(data2);
      i++;
  }
}


void ShowCGChar(Uchar addr,Uchar i)
{
     Uchar j;
    for(j=0;j<0x20;)
```

```
    {
        WRCommandH(addr+j);  //设定DDRAM地址
        WRCommandL(addr+j);  //设定DDRAM地址
        WRDataL(0x00);
        WRDataL(i);
        WRDataH(0x00);//字符地址低八位
        WRDataH(i);//字符地址高八位
        j++;
    }
}



void reg_h(Uchar x,Uchar y,Uchar x2,Uchar y2,Uchar d1,Uchar d2 )
{
       Uchar j,i;
       WRCommandH(0x34);          //去扩展指令寄存器
       WRCommandH(0x36);          //打开绘图功能
       for(j=0;j<y2;j++)               //2行   画两横上边框
          {
               WRCommandH(0x80+y+j);  //Y总坐标,即第几行
               WRCommandH(x);     //X坐标，即横数第几个字节开始写起,80H为第一个字节
               for(i=0;i<x2;i++) //写入一行
              {
               WRDataH(d1);
               WRDataH(d2);
              }
          }
}

void reg_l(Uchar x,Uchar y,Uchar x2,Uchar y2,Uchar d1,Uchar d2 )
{
       Uchar j,i;
       WRCommandL(0x34);          //去扩展指令寄存器
       WRCommandL(0x36);          //打开绘图功能
       for(j=0;j<y2;j++)               //2行   画两横上边框
          {
               WRCommandL(0x80+y+j);  //Y总坐标,即第几行
               WRCommandL(x);     //X坐标，即横数第几个字节开始写起,80H为第一个字节
               for(i=0;i<x2;i++) //写入一行
              {
```

```
                WRDataL(d1);
                WRDataL(d2);
               }
          }
}
void display_rom_8x16(bit y,Uchar column,Uchar *text)
{
Uint i=0,j;

if(y==0)
{
    WRCommandH(column);
    while(text[i]>0x00)
    {
    if((text[i]>=0x20)&&(text[i]<0x7e))
    {
        j=text[i];
        WRDataH(j);
        i++;
        j=text[i];
        WRDataH(j);
        WaitNms(10);////延时 x ms
        i++;
    }
    else
    i++;
    }
}
else
{
    WRCommandL(column);
    while(text[i]>0x00)
    {
    if((text[i]>=0x20)&&(text[i]<0x7e))
    {
        j=text[i];
        WRDataL(j);
        i++;
        j=text[i];
        WRDataL(j);
        WaitNms(10);////延时 x ms
```

```
        i++;
    }
    else
    i++;
    }
}

}

void ShowText(bit y,Uchar column,Uchar *text)
{
if(y==0)
    {
        WRCommandH(column);
        while(*text>0)
          {
           WRDataH(*text);
           text++;
          }
    }
else{
        WRCommandL(column);
        while(*text>0)
          {
           WRDataL(*text);
           text++;
          }
    }
}

void fb1234(Uchar xn)
{
    reg_h(0x80,0,12,32,0x00,0x00);          //绘图演示-洗屏
    reg_l(0x80,0,12,32,0x00,0x00);          //绘图演示-洗屏
if(xn==1)
    {
    reg_h(0x80,0,12,16,0xff,0xff);          //单行反白
    }
else if (xn==2)
    {
    reg_h(0x80,16,12,16,0xff,0xff);         //单行反白
```

```
    }
else if(xn==3)
    {
    reg_l(0x80,0,12,16,0xff,0xff);              //单行反白
    }
else if(xn==4)
    {
    reg_l(0x80,16,12,16,0xff,0xff);             //单行反白
    }
else
    _nop_();
}
/*
void time_rom_8x16(Uchar column,Uchar z,Uchar z1,Uchar d,Uchar d1,Uchar s,Uchar
s1,Uchar s10,Uchar s100)
{
//  Uchar j;
    WRCommandH(column);
    WRDataH(z+48);
    WRDataH(z1+48);
    WRDataH(':');
    WRDataH(d+48);
    WRDataH(d1+48);
    WRDataH(':');
    WRDataH(s+48);
    WRDataH(s1+48);
    WRDataH(':');
    WRDataH(s10+48);
    WRDataH(s100+48);
}
*/
void time_ram_8x16(Uchar x,Uchar y,Uchar z,Uchar z1,Uchar d,Uchar d1,Uchar s,Uchar
s1,Uchar s10,s100)
{
    Uchar k;
    WRCommandH(0x34);               //去扩展指令寄存器
    WRCommandH(0x36);               //打开绘图功能
    for(k=0;k<16;k++)
    {
    WRCommandH(0x80+y+k);  //Y总坐标,即第几行
    WRCommandH(x);   //X坐标，即横数第几个字节开始写起,80H为第一个字节
```

```
    WRDataH(num_8x16[z][k]);
    WRDataH(~(num_8x16[z1][k]));
    WRDataH(num_8x16[10][k]);
    WRDataH(num_8x16[d][k]);
    WRDataH(~(num_8x16[d1][k]));
    WRDataH(num_8x16[10][k]);
    WRDataH(num_8x16[s][k]);
    WRDataH(~(num_8x16[s1][k]));
    WRDataH(num_8x16[10][k]);
    WRDataH(num_8x16[s10][k]);
    WRDataH(~(num_8x16[s100][k]));
    }
}

void time_ram_16x32(Uchar x,Uchar y,Uchar d,Uchar d1,Uchar s,Uchar s1)
{
    Uchar k;
    WRCommandL(0x34);             //去扩展指令寄存器
    WRCommandL(0x36);             //打开绘图功能
    for(k=0;k<31;k++)
    {
    WRCommandL(0x80+y+k);  //Y总坐标,即第几行
    WRCommandL(x);   //X坐标，即横数第几个字节开始写起,80H为第一个字节

    WRDataL(num_16x32[d][k*2]);       WRDataL(num_16x32[d][k*2+1]);
    WRDataL(num_16x32[d1][k*2]); WRDataL(num_16x32[d1][k*2+1]);
    WRDataL(num_16x32[10][k*2]); WRDataL(num_16x32[10][k*2+1]);
    WRDataL(num_16x32[s][k*2]);       WRDataL(num_16x32[s][k*2+1]);
    WRDataL(num_16x32[s1][k*2]); WRDataL(num_16x32[s1][k*2+1]);
    }
}

void time_ram_24x48(Uchar x,Uchar y,Uchar z,Uchar z1,Uchar d,Uchar d1)
{
    Uchar k,j;
    WRCommandH(0x34);    WRCommandL(0x34);             //去扩展指令寄存器
    WRCommandH(0x36);    WRCommandL(0x36);             //打开绘图功能
    for(k=0;k<48;k++)
    {
        if(k<32)
```

```
        {
        WRCommandH(0x80+y+k);  //Y总坐标,即第几行
        WRCommandH(x);   //X坐标，即横数第几个字节开始写起,80H为第一个字节

        for(j=0;j<3;j++)      {WRDataH(num_24x48[z][k+k*2+j]);}
        for(j=0;j<3;j++)      {WRDataH(~(num_24x48[z1][k+k*2+j]));}
        for(j=0;j<3;j++)      {WRDataH(num_24x48[d][k+k*2+j]);}
        for(j=0;j<3;j++)      {WRDataH(~(num_24x48[d1][k+k*2+j]));}
        }
        else
        {
        WRCommandL(0x80+y+k-32);  //Y总坐标,即第几行
        WRCommandL(x);   //X坐标，即横数第几个字节开始写起,80H为第一个字节

        for(j=0;j<3;j++)      {WRDataL(num_24x48[z][k+k*2+j]);}
        for(j=0;j<3;j++)      {WRDataL(~(num_24x48[z1][k+k*2+j]));}
        for(j=0;j<3;j++)      {WRDataL(num_24x48[d][k+k*2+j]);}
        for(j=0;j<3;j++)      {WRDataL(~(num_24x48[d1][k+k*2+j]));}
        }
    }
}
//P3.2按键中断
void ini_int1(void)
{
EA=1;
EX0=1;//允许外部INT0的中断
IT0=1;// 允许中断
}
 int scankey1() interrupt 0 using 3 //使用外部中断1,寄存器组3
{
while(P3^2==0)
 {for(;;)
 {;}
 }
   IE1=0;//中断标志清零
}

//主函数
void main(void)

{
```

```
    ini_int1();//开中断
    WaitNms(250);
    LCDInit();//初始化

for(m1=0;m1<50;m1++)
    {
    reg_h(0x80,0,12,32,0x00,0x00);              //绘图演示-洗屏
    reg_l(0x80,0,12,32,0x00,0x00);              //绘图演示-洗屏
    LCDInit();//初始化
    ShowText(0,0x94,"绘晶科技");//引用汉字串显示
    ShowText(1,0x84," 192x64");//引用汉字串显示
    reg_h(0x80,0,12,32,0x00,0x00);              //绘图演示-洗屏  (x,y,x2,y2,d1,d2)
    reg_l(0x80,0,12,32,0x00,0x00);              //绘图演示-洗屏 (x,y,x2,y2,d1,d2)

    reg_h(0x80,0,12,2,0xff,0xff);           //绘图演示-画上边框 (x,y,x2,y2,d1,d2)
    reg_h(0x80,2,1,30,0xc0,0x00);           //绘图演示-画上左边框(x,y,x2,y2,d1,d2)
    reg_h(0x80+11,2,1,30,0x00,0x03);        //绘图演示-画上右边框 (x,y,x2,y2,d1,d2)

    reg_l(0x80,29,12,2,0xff,0xff);              //绘图演示-画下边框
(x,y,x2,y2,d1,d2)
    reg_l(0x80,0,1,29,0xc0,0x00);           //绘图演示-画下左边框 (x,y,x2,y2,d1,d2)
    reg_l(0x80+11,0,1,29,0x00,0x03);        //绘图演示-画下右边框 (x,y,x2,y2,d1,d2)

    WaitNms(250);           //等待时间
    WRCommandH(0x34);       WRCommandH(0x34);       //关绘图
    WRCommandL(0x34);       WRCommandL(0x34);       //关绘图
//-------------------------------------------------------------------------汉字加图形

    WRCommandH(0x01);       WRCommandL(0x01);
    ShowText(0,0x94,"绘晶科技");//引用汉字串显示
    ShowText(1,0x84," 192X64");//引用汉字串显示
    reg_h(0x80,0,12,32,0xff,0xff);              //绘图演示-全屏反白
    reg_l(0x80,0,12,32,0xff,0xff);              //绘图演示-全屏反白
    WaitNms(250);           //等待时间
//-------------------------------------------------------------------------全屏反白


    reg_h(0x80,0,12,32,0x00,0x00);              //绘图演示-洗屏
    reg_l(0x80,0,12,32,0x00,0x00);              //绘图演示-洗屏
    LCDInit();//初始化
    ShowNUMCharH(0x80,0x01,24);//显示半宽特殊符号（地址，首字符，一行个数）
```

```
    ShowNUMCharH(0x90,0x30,24);//显示半宽0~?数字标点
    ShowNUMCharL(0x80,0x41,24);//显示半宽A~P大写
    ShowNUMCharL(0x90,0x61,24);//显示半宽a~p小写
    WaitNms(250);          //等待时间
//-------------------------------------------------------------------------调
ASCII-ROM,
    LCDInit();//初始化
    WRCGRAM(0xff,0x00,0x40);//写入横（2个8位，2个8位，自编地址）
    WRCGRAM(0x00,0xff,0x50);//写入横2
    WRCGRAM(0xaa,0xaa,0x60);//写入竖
    WRCGRAM(0x55,0x55,0x70);//写入竖2-----------自编符号
    ShowCGChar(0x80,0x00);//显示横并填满（显示地址，自编数据地址）
    WaitNms(350);          //等待时间-----------------横1
    WRCommandH(0x01);     WRCommandL(0x01);
    ShowCGChar(0x80,02);//显示横2并填满
    WaitNms(350);          //等待时间-----------------横
    WRCommandH(0x01);     WRCommandL(0x01);
    ShowCGChar(0x80,04);//显示竖并填满
    WaitNms(250);          //等待时间-----------------竖1
    WRCommandH(0x01);     WRCommandL(0x01);
    ShowCGChar(0x80,06);//显示竖2并填满
    WaitNms(250);          //等待时间-----------------竖

    WRCommandH(0x01);     WRCommandL(0x01);
    WRCGRAM(0x00,0x00,0x40);//清CGRAM1
    WRCGRAM(0x00,0x00,0x50);//清CGRAM2-----------清自编符号
    WRCGRAM(0xaa,0x55,0x40);//写入点
    WRCGRAM(0x55,0xaa,0x50);//写入点2
    ShowCGChar(0x80,00);//显示点并填满
    WaitNms(250);          //等待时间-----------------点1
    WRCommandH(0x01);     WRCommandL(0x01);
    ShowCGChar(0x80,02);//显示点2并填满
    WaitNms(250);          //等待时间-----------------点
//-------------------------------------------------------------------------扫描横竖点
    reg_h(0x80,0,12,32,0x00,0x00);              //绘图演示-洗屏
    reg_l(0x80,0,12,32,0x00,0x00);              //绘图演示-洗屏
    LCDInit();//初始化
    display_rom_8x16(0,0x81,"HUIJINGKEJI:19264ZW\0");//(0为上半屏,地址,字符串)
    display_rom_8x16(0,0x91,"PHONE:18923422341\0");
    display_rom_8x16(1,0x81,"TEL: 0755-23146001\0");
    display_rom_8x16(1,0x91,"www.huijinglcm.com\0 ");
```

```
    reg_h(0x85,0,1,32,0xff,0x00);          //绘图演示-单字反白 (x,y,x2,y2,d1,d2)
    reg_l(0x85,0,1,16,0xff,0x00);          //绘图演示-单字反白 (x,y,x2,y2,d1,d2)
    reg_h(0x86,17,1,16,0xff,0x00);             //绘图演示-单字反白
(x,y,x2,y2,d1,d2)
    reg_l(0x86,0,1,32,0xff,0x00);          //绘图演示-单字反白 (x,y,x2,y2,d1,d2)
    reg_h(0x87,17,1,16,0xff,0x00);             //绘图演示-单字反白 (x,y,x2,y2,d1,d2)

    reg_l(0x87,0,1,16,0xff,0x00);          //绘图演示-单字反白 (x,y,x2,y2,d1,d2)
    reg_h(0x88,0,1,32,0xff,0x00);          //绘图演示-单字反白 (x,y,x2,y2,d1,d2)
    reg_l(0x88,0,1,32,0xff,0x00);          //绘图演示-单字反白 (x,y,x2,y2,d1,d2)
    WaitNms(250);         //等待时间
    reg_h(0x80,0,12,32,0x00,0x00);             //绘图演示-洗屏
    WRCommandH(0x34);     WRCommandH(0x34);    //关绘图
//-------------------------------------------------------------------------引用字符串
演示
    reg_h(0x80,0,12,32,0x00,0x00);             //绘图演示-洗屏
    reg_l(0x80,0,12,32,0x00,0x00);             //绘图演示-洗屏
    LCDInit();//初始化
    ShowText(0,0x80,"深圳绘晶科技有限公司");//引用汉字串显示
    ShowText(0,0x90,"专业开发/ 生产/ 销售模块");//引用汉字串显示
    ShowText(1,0x80,"19264 带中文字库显示器");//引用汉字串显示
    ShowText(1,0x90,"可单行反白/ 上下左右滚动");//引用汉字串显示
    WaitNms(150);
    fb1234(1);   WaitNms(200);          //等待时间
    fb1234(2);   WaitNms(200);          //等待时间
    fb1234(3);   WaitNms(200);          //等待时间
    fb1234(4);   WaitNms(250);          //等待时间
//-------------------------------------------------------------------------调用汉字/
单行反白
    }
    reg_h(0x80,0,12,32,0x00,0x00);             //绘图演示-洗屏
    reg_l(0x80,0,12,32,0x00,0x00);             //绘图演示-洗屏
    WRCommandH(0x34);     WRCommandH(0x34);    //关绘图
    LCDInit();//初始化
    ShowText(0,0x80,"深圳绘晶科技");//引用汉字串显示
    ShowText(1,0x95,"您好! 欢迎光临");//引用汉字串显示
    for(z=0;z<10;z++)
    {
        for(z1=0;z1<10;z1++)
        {
            for(d=0;d<6;d++)
```

```
            {

                for(d1=0;d1<10;d1++)
                {   time_ram_24x48(0x86,0,z,z1,d,d1);
                    for(s=0;s<6;s++)
                    {
                        for(s1=0;s1<10;s1++)
                        {
                            time_ram_16x32(0x80,0,d,d1,s,s1);
                            time_nms(10);////延时 x ms
                            for(s10=0;s10<10;s10++)
                            {
                                time_nms(10);////延时 x ms

                                for(s100=0;s100<10;s100++)
                                {

                                time_nms(1);////延时 x ms
                                time_ram_8x16(0x80,16,z,z1,d,d1,s,s1,s10,s100);

//                              time_rom_8x16(0x80,z,z1,d,d1,s,s1,s10,s100);

                                }
                            }
                        }
                    }
                }
            }
        }
    }

}


Uchar code  num_8x16[11][16]={
/*--  文字:  0  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=8x16   --*/
0x00,0x00,0x00,0x18,0x24,0x42,0x42,0x42,0x42,0x42,0x42,0x42,0x24,0x18,0x00,0x00,
/*--  文字:  1  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=8x16   --*/
0x00,0x00,0x00,0x10,0x70,0x10,0x10,0x10,0x10,0x10,0x10,0x10,0x7C,0x00,0x00,
```

```
/*--  文字:  2  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=8x16   --*/
0x00,0x00,0x00,0x3C,0x42,0x42,0x42,0x04,0x04,0x08,0x10,0x20,0x42,0x7E,0x00,0x00,
/*--  文字:  3  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=8x16   --*/
0x00,0x00,0x00,0x3C,0x42,0x42,0x04,0x18,0x04,0x02,0x02,0x42,0x44,0x38,0x00,0x00,
/*--  文字:  4  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=8x16   --*/
0x00,0x00,0x00,0x04,0x0C,0x14,0x24,0x24,0x44,0x44,0x7E,0x04,0x04,0x1E,0x00,0x00,
/*--  文字:  5  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=8x16   --*/
0x00,0x00,0x00,0x7E,0x40,0x40,0x40,0x58,0x64,0x02,0x02,0x42,0x44,0x38,0x00,0x00,
/*--  文字:  6  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=8x16   --*/
0x00,0x00,0x00,0x1C,0x24,0x40,0x40,0x58,0x64,0x42,0x42,0x42,0x24,0x18,0x00,0x00,
/*--  文字:  7  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=8x16   --*/
0x00,0x00,0x00,0x7E,0x44,0x44,0x08,0x08,0x10,0x10,0x10,0x10,0x10,0x10,0x00,0x00,
/*--  文字:  8  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=8x16   --*/
0x00,0x00,0x00,0x3C,0x42,0x42,0x42,0x24,0x18,0x24,0x42,0x42,0x42,0x3C,0x00,0x00,
/*--  文字:  9  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=8x16   --*/
0x00,0x00,0x00,0x18,0x24,0x42,0x42,0x42,0x26,0x1A,0x02,0x02,0x24,0x38,0x00,0x00,
/*--  文字:  :  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=8x16   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x18,0x18,0x00,0x00,0x00,0x00,0x18,0x18,0x00,0x00,
};

Uchar code  num_16x32[11][62]={
/*--  文字:  0  --*/
/*--  DS-Digital24;  此字体下对应的点阵为:宽x高=16x31   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0xFF,0x1F,0xFF,
0x1F,0xFF,0x1E,0x0F,0x1E,0x0F,0x3E,0x0F,0x3E,0x0F,0x3E,0x0F,0x1C,0x07,0x00,0x00,
0x18,0x02,0x3C,0x0F,0x3C,0x0F,0x3C,0x0F,0x3C,0x1F,0x3C,0x1F,0x7C,0x0F,0x7F,0xFF,
0x7F,0xFF,0x3F,0xFE,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  1  --*/
/*--  DS-Digital24;  此字体下对应的点阵为:宽x高=9x31   --*/
/*--  宽度不是8的倍数,现调整为:宽度x高度=16x31  --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0x00,0x07,0x00,
0x0F,0x00,0x1F,0x00,0x1F,0x00,0x1F,0x00,0x1F,0x00,0x1F,0x00,0x0E,0x00,0x00,0x00,
```

```
0x04,0x00,0x1E,0x00,0x1E,0x00,0x3E,0x00,0x3E,0x00,0x3E,0x00,0x3E,0x00,0x1E,0x00,
0x0E,0x00,0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  2  --*/
/*--  DS-Digital24;  此字体下对应的点阵为：宽x高=16x31   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0xFF,0x07,0xFF,
0x03,0xFF,0x00,0x07,0x00,0x0F,0x00,0x0F,0x00,0x0F,0x00,0x0F,0x07,0xFF,0x0F,0xFE,
0x1F,0xFC,0x3C,0x00,0x3C,0x00,0x3C,0x00,0x3C,0x00,0x3C,0x00,0x3C,0x00,0x7F,0xF8,
0x7F,0xFC,0x3F,0xFE,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  3  --*/
/*--  DS-Digital24;  此字体下对应的点阵为：宽x高=16x31   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x1F,0xFF,0x0F,0xFF,
0x07,0xFF,0x00,0x0F,0x00,0x0F,0x00,0x0F,0x00,0x1F,0x00,0x0F,0x0F,0xFF,0x1F,0xFC,
0x0F,0xFE,0x00,0x0F,0x00,0x1F,0x00,0x1F,0x00,0x1E,0x00,0x1E,0x00,0x1E,0x0F,0xFE,
0x1F,0xFE,0x7F,0xFC,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  4  --*/
/*--  DS-Digital24;  此字体下对应的点阵为：宽x高=16x31   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x30,0x03,0x38,0x07,
0x38,0x0F,0x3C,0x1F,0x3C,0x1F,0x7C,0x1F,0x7C,0x1F,0x7C,0x1F,0x3F,0xFE,0x1F,0xFC,
0x0F,0xFC,0x00,0x1E,0x00,0x3E,0x00,0x3E,0x00,0x3E,0x00,0x3E,0x00,0x3E,0x00,0x1E,
0x00,0x0E,0x00,0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  5  --*/
/*--  DS-Digital24;  此字体下对应的点阵为：宽x高=16x31   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x1F,0xFF,0x3F,0xFE,
0x3F,0xFC,0x3C,0x00,0x3C,0x00,0x3C,0x00,0x3C,0x00,0x3C,0x00,0x3F,0xF8,0x1F,0xFC,
0x0F,0xFE,0x00,0x0F,0x00,0x1F,0x00,0x1F,0x00,0x1F,0x00,0x1E,0x00,0x1E,0x0F,0xFE,
0x1F,0xFE,0x7F,0xFC,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  6  --*/
/*--  DS-Digital24;  此字体下对应的点阵为：宽x高=16x31   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0xFF,0x1F,0xFE,
0x1F,0xFC,0x1E,0x00,0x1E,0x00,0x3E,0x00,0x3E,0x00,0x3E,0x00,0x1F,0xFC,0x0F,0xFE,
0x1F,0xFE,0x3C,0x07,0x3C,0x0F,0x3C,0x0F,0x3C,0x0F,0x3C,0x0F,0x7C,0x0F,0x7F,0xFF,
0x7F,0xFF,0x3F,0xFE,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  7  --*/
/*--  DS-Digital24;  此字体下对应的点阵为：宽x高=16x31   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x1F,0xFF,0x07,0xFF,
0x03,0xFF,0x00,0x0F,0x00,0x0F,0x00,0x0F,0x00,0x0F,0x00,0x0F,0x00,0x07,0x00,0x00,
0x00,0x06,0x00,0x0F,0x00,0x1F,0x00,0x1F,0x00,0x1F,0x00,0x1F,0x00,0x1F,0x00,0x0F,
0x00,0x0E,0x00,0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  8  --*/
/*--  DS-Digital24;  此字体下对应的点阵为：宽x高=16x31   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0xFF,0x1F,0xFF,
```

```
0x1F,0xFF,0x1E,0x0F,0x1E,0x0F,0x3E,0x0F,0x3E,0x0F,0x3E,0x0F,0x1F,0xFF,0x0F,0xFE,
0x1F,0xFE,0x3C,0x0F,0x3C,0x0F,0x3C,0x0F,0x3C,0x1F,0x3C,0x1F,0x7C,0x0F,0x7F,0xFF,
0x7F,0xFF,0x3F,0xFE,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  9  --*/
/*--  DS-Digital24;  此字体下对应的点阵为：宽x高=16x31   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x1F,0xFF,0x3F,0xFF,
0x3F,0xFF,0x3C,0x0F,0x3C,0x0F,0x3C,0x0F,0x3C,0x1F,0x3C,0x0F,0x3F,0xFF,0x1F,0xFC,
0x0F,0xFE,0x00,0x0F,0x00,0x1F,0x00,0x1F,0x00,0x1E,0x00,0x1E,0x00,0x1E,0x0F,0xFE,
0x1F,0xFE,0x7F,0xFC,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  -  --*/
/*--  DS-Digital24;  此字体下对应的点阵为：宽x高=15x31   --*/
/*--  宽度不是8的倍数，现调整为：宽度x高度=16x31  --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x3F,0xFC,0x7F,0xFE,
0x3F,0xFC,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
};


Uchar code  num_24x48[11][144]={
/*--  文字:  0  --*/
/*--  宋体36;  此字体下对应的点阵为：宽x高=24x48   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7E,0x00,0x01,0xE7,0x80,0x03,0xC3,
0xC0,0x07,0x81,0xE0,0x0F,0x80,0xF0,0x0F,0x00,0xF0,0x1F,0x00,0xF8,0x1E,0x00,0xF8,
0x3E,0x00,0x78,0x3E,0x00,0x7C,0x3E,0x00,0x7C,0x3E,0x00,0x7C,0x3E,0x00,0x7C,0x3E,
0x00,0x7C,0x3E,0x00,0x7C,0x3E,0x00,0x7C,0x3E,0x00,0x7C,0x3E,0x00,0x7C,0x3E,0x00,
0x7C,0x3E,0x00,0x7C,0x3E,0x00,0x7C,0x3E,0x00,0x7C,0x3E,0x00,0x7C,0x3E,0x00,0x78,
0x1E,0x00,0xF8,0x1F,0x00,0xF8,0x0F,0x00,0xF0,0x0F,0x81,0xF0,0x07,0x81,0xE0,0x03,
0xC3,0xC0,0x01,0xE7,0x80,0x00,0x7E,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  1  --*/
/*--  宋体36;  此字体下对应的点阵为：宽x高=24x48   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0C,0x00,0x00,0x1C,0x00,0x00,0x7C,
0x00,0x07,0xFC,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,
0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,
0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,
0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,
0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,0x3C,0x00,0x00,
0x3E,0x00,0x00,0x7F,0x00,0x07,0xFF,0xF0,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
```

```
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  2  --*/
/*--  宋体36;  此字体下对应的点阵为:宽x高=24x48   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0x00,0x03,0xC7,0xC0,0x07,0x01,
0xE0,0x0E,0x00,0xF0,0x1E,0x00,0xF8,0x1E,0x00,0xF8,0x3E,0x00,0x78,0x3E,0x00,0x78,
0x3F,0x00,0x78,0x3F,0x00,0x78,0x1F,0x00,0xF8,0x00,0x00,0xF8,0x00,0x00,0xF0,0x00,
0x01,0xF0,0x00,0x03,0xE0,0x00,0x03,0xC0,0x00,0x07,0x80,0x00,0x0F,0x00,0x00,0x1E,
0x00,0x00,0x3C,0x00,0x00,0x78,0x00,0x00,0xF0,0x00,0x01,0xE0,0x00,0x03,0xC0,0x00,
0x07,0x80,0x1C,0x07,0x00,0x1C,0x0E,0x00,0x38,0x1C,0x00,0x38,0x3C,0x00,0x78,0x3F,
0xFF,0xF8,0x3F,0xFF,0xF8,0x3F,0xFF,0xF8,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  3  --*/
/*--  宋体36;  此字体下对应的点阵为:宽x高=24x48   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFE,0x00,0x07,0x87,0x80,0x0F,0x03,
0xC0,0x1E,0x01,0xE0,0x1E,0x01,0xF0,0x1E,0x01,0xF0,0x1F,0x00,0xF0,0x1F,0x00,0xF0,
0x1E,0x00,0xF0,0x00,0x00,0xF0,0x00,0x01,0xF0,0x00,0x01,0xF0,0x00,0x03,0xE0,0x00,
0x03,0xC0,0x00,0x0F,0x00,0x00,0xFE,0x00,0x00,0x07,0x80,0x00,0x01,0xE0,0x00,0x00,
0xF0,0x00,0x00,0xF8,0x00,0x00,0xF8,0x00,0x00,0x78,0x00,0x00,0x7C,0x1E,0x00,0x7C,
0x3F,0x00,0x7C,0x3F,0x00,0x7C,0x3F,0x00,0x78,0x3E,0x00,0xF8,0x1E,0x00,0xF0,0x0F,
0x01,0xE0,0x07,0x87,0xC0,0x01,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  4  --*/
/*--  宋体36;  此字体下对应的点阵为:宽x高=24x48   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0xC0,0x00,0x03,0xC0,0x00,0x07,
0xC0,0x00,0x0F,0xC0,0x00,0x0F,0xC0,0x00,0x1F,0xC0,0x00,0x3F,0xC0,0x00,0x3F,0xC0,
0x00,0x77,0xC0,0x00,0x77,0xC0,0x00,0xE7,0xC0,0x01,0xC7,0xC0,0x01,0xC7,0xC0,0x03,
0x87,0xC0,0x07,0x07,0xC0,0x07,0x07,0xC0,0x0E,0x07,0xC0,0x1E,0x07,0xC0,0x1C,0x07,
0xC0,0x38,0x07,0xC0,0x38,0x07,0xC0,0x7F,0xFF,0xFE,0x7F,0xFF,0xFE,0x00,0x07,0xC0,
0x00,0x07,0xC0,0x00,0x07,0xC0,0x00,0x07,0xC0,0x00,0x07,0xC0,0x00,0x07,0xC0,0x00,
0x07,0xC0,0x00,0x07,0xE0,0x00,0x7F,0xFE,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  5  --*/
/*--  宋体36;  此字体下对应的点阵为:宽x高=24x48   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0xFF,0xF8,0x0F,0xFF,0xF8,0x0F,0xFF,
0xF8,0x0E,0x00,0x00,0x0E,0x00,0x00,0x0E,0x00,0x00,0x0E,0x00,0x00,0x0E,0x00,0x00,
0x0E,0x00,0x00,0x0E,0x00,0x00,0x0E,0x00,0x00,0x0E,0x7F,0x00,0x0D,0xFF,0xC0,0x0F,
0xC3,0xE0,0x1F,0x01,0xF0,0x1E,0x00,0xF8,0x1E,0x00,0xF8,0x00,0x00,0x78,0x00,0x00,
```

```
0x7C,0x00,0x00,0x7C,0x00,0x00,0x7C,0x00,0x00,0x7C,0x1E,0x00,0x7C,0x3F,0x00,0x7C,
0x3F,0x00,0x78,0x3F,0x00,0x78,0x3E,0x00,0xF8,0x1E,0x00,0xF0,0x1E,0x01,0xF0,0x0E,
0x01,0xE0,0x07,0x87,0xC0,0x00,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  6  --*/
/*--  宋体36;  此字体下对应的点阵为:宽x高=24x48   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x3F,0xC0,0x00,0xF1,0xE0,0x03,0xC1,
0xF0,0x07,0x81,0xF8,0x07,0x01,0xF8,0x0F,0x00,0xF0,0x1F,0x00,0x00,0x1E,0x00,0x00,
0x1E,0x00,0x00,0x3E,0x00,0x00,0x3E,0x00,0x00,0x3E,0x00,0x00,0x3E,0x3F,0x80,0x3E,
0xFF,0xE0,0x3F,0xE3,0xF0,0x3F,0x80,0xF8,0x3F,0x00,0xF8,0x3F,0x00,0x7C,0x3E,0x00,
0x7C,0x3E,0x00,0x7C,0x3E,0x00,0x3C,0x3E,0x00,0x3C,0x3E,0x00,0x3C,0x3E,0x00,0x3C,
0x3E,0x00,0x7C,0x1E,0x00,0x7C,0x1F,0x00,0x78,0x0F,0x00,0x78,0x0F,0x80,0xF0,0x07,
0xC0,0xE0,0x03,0xE3,0xC0,0x00,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  7  --*/
/*--  宋体36;  此字体下对应的点阵为:宽x高=24x48   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x1F,0xFF,0xFC,0x1F,0xFF,0xFC,0x1F,0xFF,
0xF8,0x1F,0x00,0x38,0x1C,0x00,0x70,0x1C,0x00,0xE0,0x38,0x00,0xE0,0x38,0x01,0xC0,
0x00,0x01,0xC0,0x00,0x03,0x80,0x00,0x03,0x80,0x00,0x07,0x80,0x00,0x07,0x00,0x00,
0x0F,0x00,0x00,0x0E,0x00,0x00,0x1E,0x00,0x00,0x1E,0x00,0x00,0x3C,0x00,0x00,0x3C,
0x00,0x00,0x3C,0x00,0x00,0x7C,0x00,0x00,0x78,0x00,0x00,0x78,0x00,0x00,0xF8,0x00,
0x00,0xF8,0x00,0x00,0xF8,0x00,0x00,0xF8,0x00,0x00,0xF8,0x00,0x00,0xF8,0x00,0x00,
0xF8,0x00,0x00,0xF8,0x00,0x00,0x78,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  8  --*/
/*--  宋体36;  此字体下对应的点阵为:宽x高=24x48   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0x00,0x07,0xC3,0xC0,0x0F,0x00,
0xE0,0x1E,0x00,0xF0,0x1E,0x00,0x78,0x3C,0x00,0x78,0x3C,0x00,0x78,0x3C,0x00,0x7C,
0x3E,0x00,0x78,0x3E,0x00,0x78,0x1F,0x00,0x78,0x1F,0x80,0xF0,0x0F,0xE1,0xE0,0x07,
0xFB,0xC0,0x01,0xFF,0x80,0x01,0xFF,0x80,0x07,0xBF,0xC0,0x0F,0x0F,0xE0,0x1E,0x03,
0xF0,0x3E,0x01,0xF8,0x3C,0x00,0xF8,0x3C,0x00,0x7C,0x7C,0x00,0x7C,0x78,0x00,0x3C,
0x78,0x00,0x3C,0x7C,0x00,0x3C,0x3C,0x00,0x78,0x3C,0x00,0x78,0x1E,0x00,0x70,0x0F,
0x00,0xE0,0x07,0xC3,0xC0,0x01,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  9  --*/
/*--  宋体36;  此字体下对应的点阵为:宽x高=24x48   --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0xFE,0x00,0x07,0xC7,0x80,0x0F,0x01,
```

```
0xE0,0x1E,0x01,0xE0,0x1E,0x00,0xF0,0x3E,0x00,0xF8,0x3C,0x00,0x78,0x3C,0x00,0x78,
0x7C,0x00,0x7C,0x7C,0x00,0x7C,0x7C,0x00,0x7C,0x7C,0x00,0x7C,0x7C,0x00,0x7C,0x3C,
0x00,0xFC,0x3E,0x00,0xFC,0x3E,0x01,0xFC,0x1F,0x03,0xFC,0x1F,0x8F,0xFC,0x0F,0xFF,
0x7C,0x03,0xFC,0x7C,0x00,0x00,0x7C,0x00,0x00,0xF8,0x00,0x00,0xF8,0x00,0x00,0xF8,
0x00,0x00,0xF0,0x00,0x01,0xF0,0x0F,0x01,0xE0,0x1F,0x01,0xE0,0x1F,0x03,0xC0,0x1F,
0x07,0x80,0x0F,0x9F,0x00,0x03,0xFC,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
/*--  文字:  :  --*/
/*--  宋体36;  此字体下对应的点阵为:宽x高=24x48  --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7C,0x00,0x00,0x7E,0x00,0x00,0xFE,0x00,0x00,
0xFE,0x00,0x00,0x7E,0x00,0x00,0x7C,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7C,0x00,0x00,0x7E,0x00,0x00,0xFE,0x00,0x00,
0xFE,0x00,0x00,0x7E,0x00,0x00,0x7C,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
};
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~完~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```