液晶模块使用说明书

型号：ZX320240C-H-BTSSWE-YAA

（带中文字库,RA8806 控制器）

修改记录

版本号	日期	PAGE	内容
1.0	2008-9-19		NEW RELEASE

LCD MODULE NUMBERING SYSTEM

ZX320240C -__ __ __ __ __ __ -__ __ __

① ② ③ ④ ⑤ ⑥ ⑦ ⑧ ⑨

ZX ZHONGXIAN TECHNOLOGY

G GRAPHIC TYPE

320240 SERIALS NUMBER FOR SM 320 COLUMNS Vs. 240 ROWS

C VERSION OF PCB

N0	Remark	Code Value	Description
①	LCD Type	Y	STN yellow/green type LCD
		B	STN blue type LCD
		G	STN gray type LCD
		F	FSTN positive type LCD
		N	FSTN negative type LCD
		P	TN positive type LCD
		E	TN negative type LCD
②	Polarizer Type	R	Reflecive type
		F	Transflecive type
		T	Transmissive type
③	Viewing Angel	S	Six o'clock
		T	Twelve o'clock
④	Backlight Type	N	Without backlight
		E	EL backlight
		C	CCFL backlight
		D	Bottom LED backlight
		S	Side LED backlight
⑤	Backlight Colour	N	Without backlight
		Y	Yellow/green
		W	White
		G	Green
		B	Blue
		A	Amber
		0	Orange
⑥	LCM Temperation Type	N	Normal
		E	Extended
		S	Super extended
⑦	DC-DC Converter Circuit	N	Without
		Y	With
⑧	Power Supply For Logic	A	5.0V
		B	3.3V
⑨	Power Supply For Backlight	A	5.0V
		B	4.2V
		C	3.3V

目录

一：主要技术参数和性能

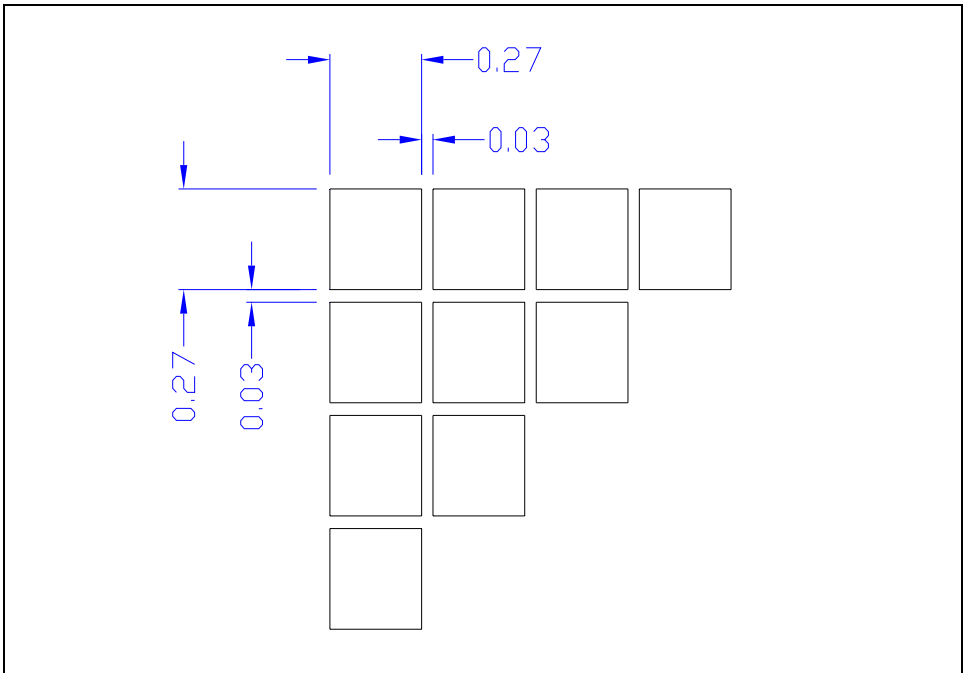
1. 电源： VDD= +5.0V ±5% ,模块上自带-24V 电压，用于 LCD 的驱动电压，背光电压：5.0V
2. 显示内容：320（列）X 240（行）
3. 驱动方式：1/240 DUTY , 1/17 BIAS
4. 显示模式：STN 蓝模 负显（蓝底白字）
5. 背光特性：白色 LED 侧背光, DC+5V 驱动
6. 控制芯片：RA8806（自带中文字库，带触摸屏驱动）
7. 参观视角：6 点
8. 工作温度：-20℃--+70℃
9. 存储温度：-30℃--+80℃
10. 与 MCU 接口时序：INTEL 8080

二：外型尺寸图

1．主要外型尺寸：

项　　目	标准尺寸	单位
模块体积	139.0 X 100.0 X 13.0T	MM
视窗尺寸	103.0 X 79.0	MM
点阵数	320 X 240	DOTS
点间距	0.30 X 0.30	MM
点大小	0.27 X 0.27	MM

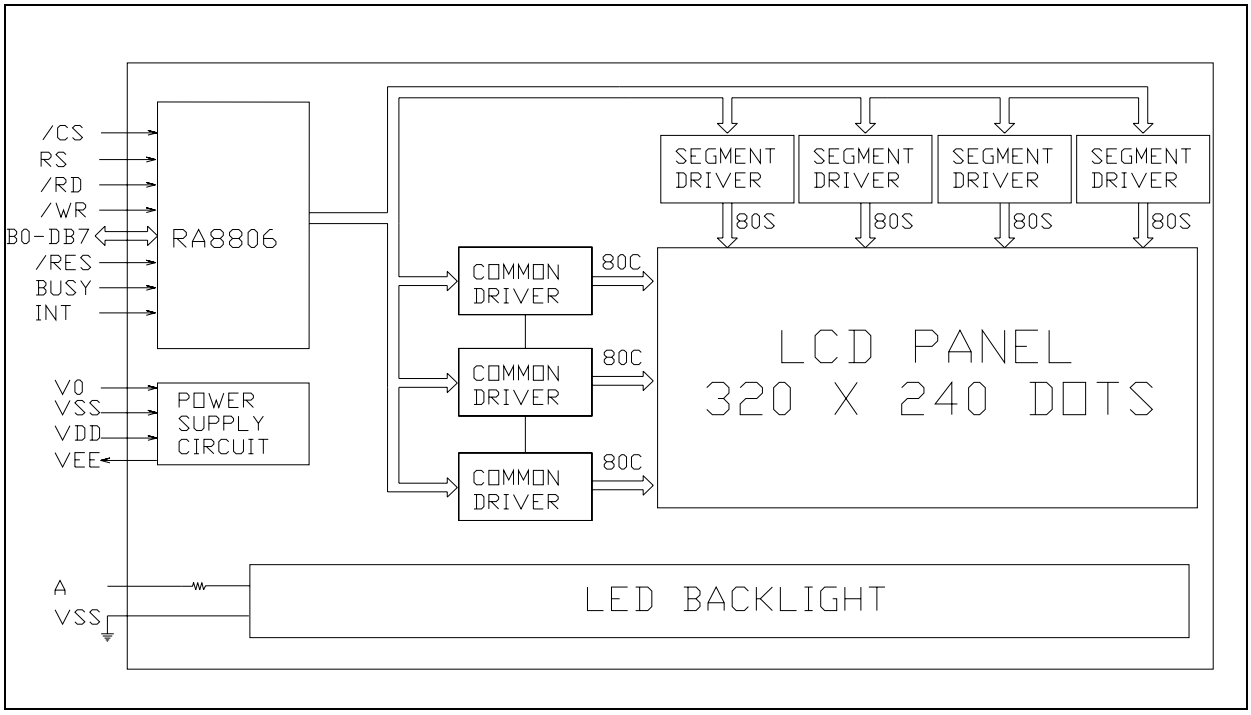
2．点阵尺寸：



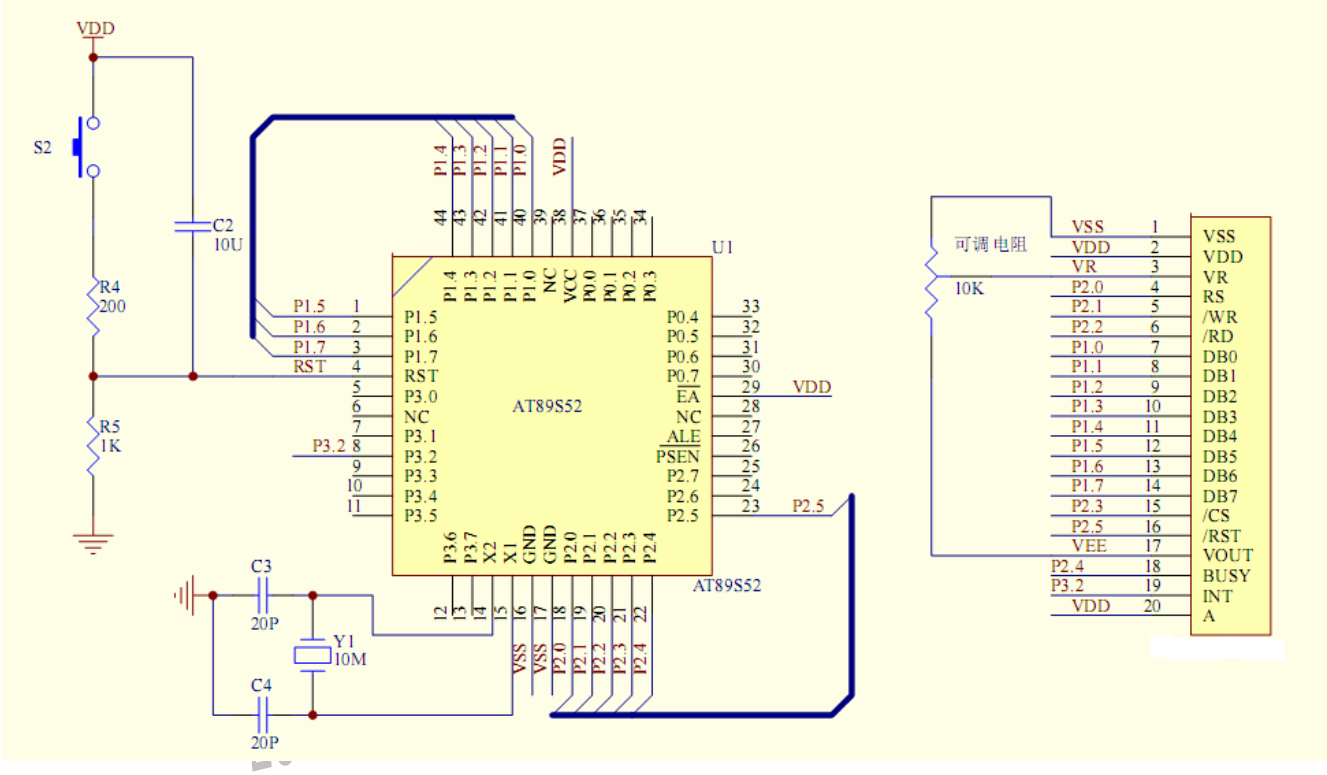
3．模块外型尺寸：

PDF

三：硬件结构图



四：与单片机通讯应用电路参考



五：引脚定义 （RA8806 控制器，LED 背光）

引脚号	符　号	电　平	功　能
1	V_{SS}	0	电源地
2	V_{DD}	+5.0V	逻辑电压
3	V_R	—	LCD 驱动电压调节
4	RS	H	H：指令端口　L：数据端口
5	/WR	L	L：数据写入　H：无效
6	/RD	L	L：数据读出　H：无效
7	DB_0	H / L	数据总线
8	DB_1	H / L	
9	DB_2	H / L	
10	DB_3	H / L	
11	DB_4	H / L	
12	DB_5	H / L	
13	DB_6	H / L	
14	DB_7	H / L	
15	/CS1	L	芯片选择信号，低电平有效
16	RST	L	复位信号（低电平有效）
17	VOUT	-24.0V	模块内部负压输出（-24.0V）
18	BUSY	H/L	系统忙判断信号
19	INT	H/L	系统发生中断信号
20	A	+5.0V	LED 背光电压输入（+）（5.0V）

六：电气参数

1. 极限参数

1.1 电气极限参数

参数	符号	条件	最小值	最大值	单位
逻辑电压	Vdd - Vss	-	-0.3	7.0	V
LCD 驱动电压	Vdd - V0	-	0	30.0	V
输入电压	Vi	-	-0.3	Vdd +0.3	V

1.2 环境极限参数

参数	符号	条件	最小值	最大值	单位
工作温度	Topr	-Normal temp. version-	-20	70	deg C
存储温度	Ttsg		-30	80	deg C
Humidity Endurance	RH	no ondensation Ta<=40 deg	-	95	%
振动压力	-	100-300Hz, X/Y/Z directions, 1 hour	-	4.9m/ss 0.5g	-
震动	-	10 mS X/Y/Z direction 1 time each		29.4m/ss 3.0g	-

2. 电气特性

2.1 直流特性

电气特性 at Ta=25 deg C, Vdd = 5V + / - 5%

参数	符号	条件	最小值	典型	最大值	单位
逻辑电压	Vdd-Vss	-	4.5	5.0	5.5	V
LCD 驱动电压	Vdd-V0	Vdd = 5V	-	21.0	-	V
输入电压 (for RS, DB0-7, /WR, /RD /CS1, CS2)	V-ih	“H” level	2.2	-	Vdd	V
	V-il	“L” level	0	-	0.6	V
逻辑电流	Icc	-	-	1	1.2	mA
LCD 驱动电流	Io	-	0.15	0.22	0.27	mA

2.2　交流特性

(1) 8080 时序

时序说明 at Ta = 25 deg C, Vdd = 5V+/-10%, Vss =0V

项目	符号	最小值	最大值	单位
地址延时时间	Tah8	20	-	ns
地址上升时间	Taw8	30	-	ns
系统循环时间	Tcyc8	2TC	-	ns
存储脉冲建立时间	Tcc8	50	-	ns
数据保持时间	Tdh8	20	-	ns
数据建立时间	Tds8	30		ns
数据稳态时间	Tacc8	0	20	ns

时序说明 at Ta = 25 deg C, Vdd = 3V+/-10%, Vss =0V

项目	符号	最小值	最大值	单位
地址延时时间	Tah8	20	-	ns
地址上升时间	Taw8	30	-	ns
系统循环时间	Tcyc8	2TC	-	ns
存储脉冲建立时间	Tcc8	50	-	ns
数据保持时间	Tdh8	20	-	ns
数据建立时间	Tds8	30		ns
数据稳态时间	Tacc8	0	20	ns

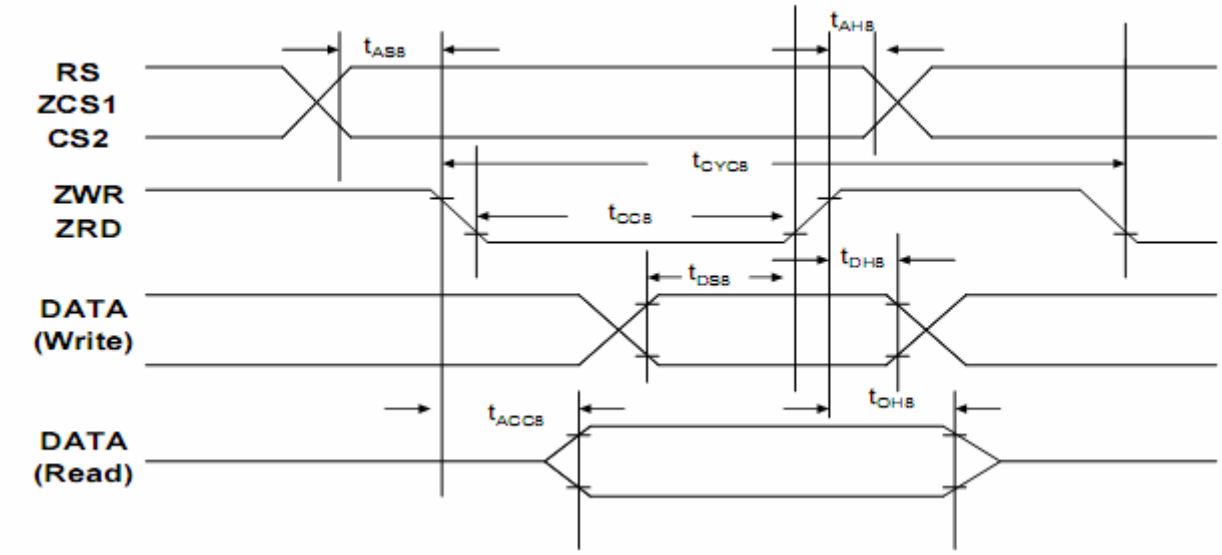
说明：

*1. 输入信号的上升/下降时间不应该小于 20 NS

*2. 对于存储控制和系统控制指令:tcyc8=2tc+Tcea+75>tacv+245

*3. 对于所有其他的指令:tcyc8=4tc+tcc8+30

*4. 关于详细的说明请参考 RA8806 的数据手册



(2) 6800 时序

时序说明 at Ta = 25 deg C, Vdd = 5V+/-10%, Vss =0V

项目	符号	最小值	最大值	单位
地址延时时间	Tah6	20	-	ns
地址上升时间	Taw6	30	-	ns
系统循环时间	Tcyc6	2TC	-	ns
存储脉冲建立时间	Tcc6	50	-	ns
数据保持时间	Tdh6	20	-	ns
数据建立时间	Tds6	30		ns
数据稳态时间	Tacc6	0	20	ns

TIMING SPECIFICATIONS at Ta = 25 deg C, Vdd = 3V+/-10%, Vss =0V

项目	符号	最小值	最大值	单位
地址延时时间	Tah6	20	-	ns
地址上升时间	Taw6	30	-	ns
系统循环时间	Tcyc6	2TC	-	ns
存储脉冲建立时间	Tcc6	50	-	ns
数据保持时间	Tdh6	20	-	ns
数据建立时间	Tds6	30		ns
数据稳态时间	Tacc6	0	20	ns

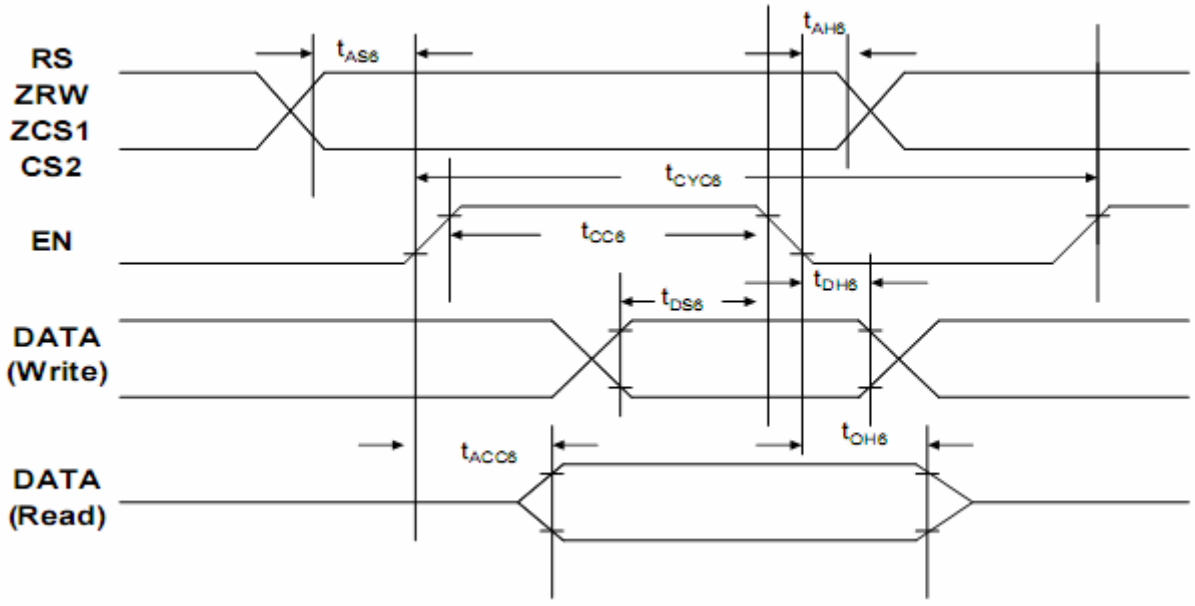
NOTE:

*1. 输入信号的上升/下降时间不应该小于 20 NS

*2. 对于存储控制和系统控制指令:tcyc6=2tc+ Tew6+tcea+75>tacv+245

*3. 对于所有其他的指令:tcyc8=4tc+tcc+30

*4. 关于详细的说明请参考 RA8806 的数据手册



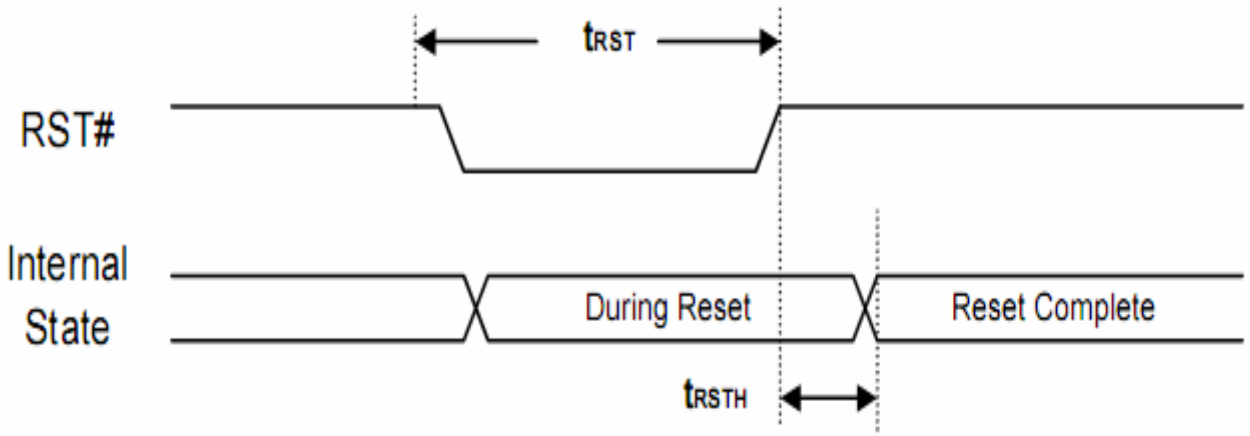
（3）复位时序

时序说明 at Ta = 25 deg C, Vdd = 5V+/-10%, Vss =0V

项目	符号	最小值	最大值	单位
复位脉冲	Trst	1.0	-	ms

时序说明 at Ta = 25 deg C, Vdd = 3V+/-10%, Vss =0V

项目	符号	最小值	最大值	单位
复位脉冲	Trst	1.0	-	ms



七：光电特性

项目	符号	条件	最小值	典型值	最大值	单位	参考.
对比度	CR	25℃	--	12	--		备注1
上升时间	tr	25℃	--	160	240	ms	备注2
下降时间	tf	25℃	--	100	150	ms	备注 2
参观视角	θ1-θ2	25℃	--	--	60	DEG	备注 3
	Ø1, Ø2		-40	--	40		
帧频率	Ff	25℃	--	70	--	Hz	备注 2

备注(1): 对比率是由以下条件决定的:

CR= 选择情况的亮度
非选择情况的亮度

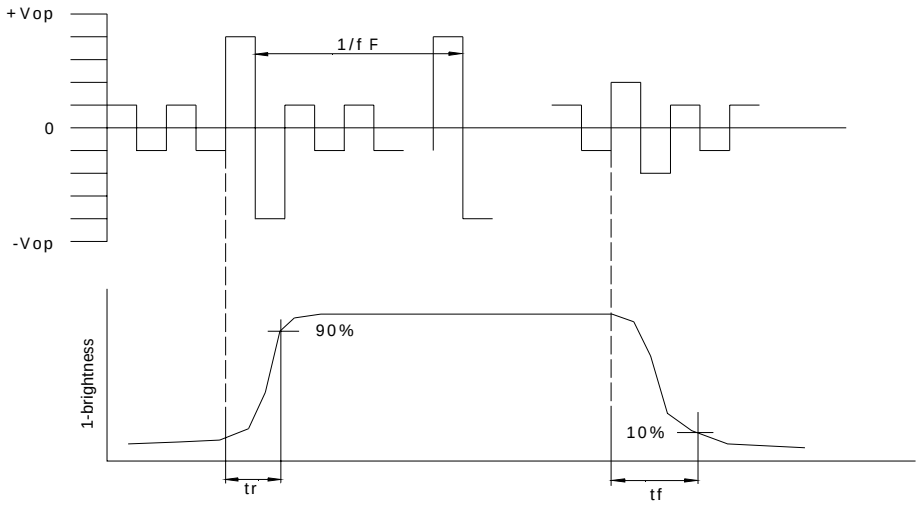
(a). 温度----------25C

(b). 帧频率------64Hz

(c). 参观视角---------- θ=0, Ø=0

(d). 操作电压---5.0V

备注(2)：响应时间的定义：



条件：

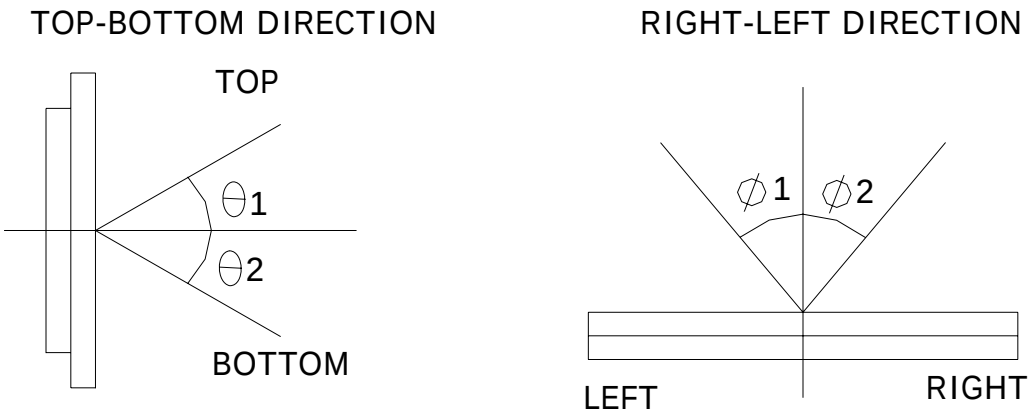
(a). 温度----------25C

(b). 帧频率------64Hz

(c). 参观视角---------- θ =0, Ø=0

(d). 操作电压---5.0V

备注(3)：视角定义：



八：指令说明

1. 指令表

REG#	Name	D7	D6	D5	D4	D3	D2	D1	D0	初始值
--	**STATUS**	MBUSY	SBUSY	SLEEP			WAKE_STS	KS_STS	TP_STS	--
00h	**WLCR**	PWR	LINEAR	SRST	--	TEXT_MD	ZDOFF	GBLK	GINV	00h
01h	**MISC**	NO_FLICKER	CLKO_SEL	BUSY_LEV	INT_LEV	XCK_SEL1	XCK_SEL0	SDIR	CDIR	04h
03h	**ADSR**	SCR_PEND	--	--	--	BIT_INV	SCR_DIR	SCR_HV	SCR_EN	00h
0Fh	**INTR**	--	WAKI_EN	KEYI_EN	TPI_EN	TP_ACT	WAK_STS	KEY_STS	TP_STS	00h
10h	**WCCR**	CUR_INC	FULL_OFS	BIT_REV	BOLD	T90DEG	CUR_EN	CUR_BLK	--	00h
11h	**CHWI**	CURH3	CURH2	CURH1	CURH0	ROWH3	ROWH 2	ROWH 1	ROWH 0	00h
12h	**MAMR**	CUR_HV	DISPMD2	DISPMD1	DISPMD0	L_MIX1	L_MIX 0	MW_MD1	MW_MD0	11h
20h	**AWRR**	--	--	AWR5	AWR4	AWR3	AWR2	AWR1	AWR0	27h
21h	**DWWR**	--	--	DWW5	DWW 4	DWW 3	DWW 2	DWW 1	DWW 0	27h
30h	**AWBR**	AWB7	AWB6	AWB5	AWB4	AWB3	AWB2	AWB1	AWB0	EFh
31h	**DWHR**	DWH7	DWH6	DWH5	DWH4	DWH3	DWH2	DWH1	DWH0	EFh
40h	**AWLR**	--	--	AWL5	AWL4	AWL3	AWL2	AWL1	AWL0	00h
50h	**AWTR**	AWT7	AWT6	AWT5	AWT4	AWT3	AWT2	AWT1	AWT0	00h
60h	**CURX**	--	--	CURX5	CURX4	CURX3	CURX2	CURX1	CURX0	00h
61h	**BGSG**	--	--	BGSG5	BGSG4	BGSG3	BGSG2	BGSG1	BGSG0	00h
62h	**EDSG**	EDSG7	EDSG6	EDSG5	EDSG4	EDSG3	EDSG2	EDSG1	EDSG0	00h
70h	**CURY**	CURY7	CURY6	CURY5	CURY4	CURY3	CURY2	CURY1	CURY0	00h
71h	**BGCM**	BGCM7	BGCM6	BGCM5	BGCM4	BGCM3	BGCM2	BGCM1	BGCM0	00h
72h	**EDCM**	EDCM7	EDCM6	EDCM5	EDCM4	EDCM3	EDCM2	EDCM1	EDCM0	00h
80h	**BTMR**	BLKT7	BLKT6	BLKT5	BLKT4	BLKT3	BLKT2	BLKT1	BLKT0	00h
90h	**ITCR**	ITC7	ITC6	ITC5	ITC4	ITC3	ITC2	ITC1	ITC0	00h
A0h	**KSCR1**	KEY_EN	KEY4X8	KSAMP1	KSAMP0	LKEY_EN	KF2	KF1	KF0	00h
A1h	**KSCR2**	KWAK_EN	--	--	--	LKEY_T1	LKEY_T0	KEYNO1	KEYNO0	00h
A2h	**KSDR0**	KSD07	KSD06	KSD05	KSD04	KSD03	KSD02	KSD01	KSD00	00h
A3h	**KSDR1**	KSD17	KSD16	KSD15	KSD14	KSD13	KSD12	KSD11	KSD10	00h
A4h	**KSDR2**	KSD27	KSD26	KSD25	KSD24	KSD23	KSD22	KSD21	KSD20	00h
B0h	**MWCR**	MWD7	MWD6	MWD5	MWD4	MWD3	MWD2	MWD1	MWD0	--
B1h	**MRCR**	MRD7	MRD6	MRD5	MRD4	MRD3	MRD2	MRD1	MRD0	--

(Continued)

REG#	Name	D7	D6	D5	D4	D3	D2	D1	D0	初始值
C0h	TPCR1	TP_EN	TP_SMP2	TP_SMP1	TP_SMP0	TPWAK_EN	ACLK2	ACLK1	ACLK0	00h
C1h	TPXR	TPX9	TPX8	TPX7	TPX6	TPX5	TPX4	TPX3	TPX2	00h
C2h	TPYR	TPY9	TPY8	TPY7	TPY6	TPY5	TPY4	TPY3	TPY2	00h
C3h	TPZR	TPX1	TPX0	--	--	TPY1	TPY0	--	--	00h
C4h	TPCR2	MTP_MD	--	--	--	--	--	MTP_PH1	MTP_PH2	00h
D0h	PCR	PWM_EN	PWM_DIS_LEV	--	--	PCLK_R3	PCLK_R2	PCLK_R1	PCLK_R0	00h
D1h	PDCR	PDUTY7	PDUTY6	PDUTY5	PDUTY4	PDUTY3	PDUTY2	PDUTY1	PDUTY0	00h
E0h	PNTR	PND7	PND6	PND5	PND4	PND3	PND2	PND1	PND0	00h
F0h	FNCR	ISO8859_EN	--	--	--	MCLR	ASC	ASC_SEL1	ASC_SEL0	00h
F1h	FVHT	FH1	FH0	FV1	FV0	--	--	--	--	00h

2. 基本指令详解

(1).其本功能概述

此款中文液晶显示模块是一个中英文文字与绘图模式的点矩阵液晶显示模块，内建 512KByte 的 ROM 字形码，可以显示中文字型、数字符号、英日欧文等字母，并且内建双图层(Two Page)的显示内存。在文字模式中，可接收标准中文文字内码直接显示中文，而不需要进入绘图模式以绘图方式描绘中文，可以节省许多微处理器时间，提升液晶显示中文之处理效率。

支持文字与绘图两种混和显示模式支持 2 Page 显示模式(And, Or, Nor, Xor)，内建两个 4.8K / 9.6 K（15x20D） Byte 的显示 RAM (Display DataRAM)，共 9.6K / 19.2 K（15x20D） Byte RAM，并且可做成 4 阶的显示效果。内建 512KByte ROM，控制 IC 分带繁体字库 IC 和带简体字库 IC，其中标准体中文 BIG5 码，包含 13,094 个常用与次常用字型、408 个特殊字与两组 ASCII CODE，简体字库储存 7602 个标准 GB 码的简体中文。

提供全角(16x16)与半角(8x16)文字显示模式

支持 4/8 位之 6800/8080 MCU 接口

带光标、反白、闪烁功能，且光标高度与宽度可调

支持屏幕水平卷动及垂直卷动功能

内建 512Byte SRAM 可自行造字

提供中/英文文字对齐功能

显示字型可放大到 32x32、48x48 或 64x64，以及混合显示模式

支持可将字型由 ROM 直接读出使用

内建粗体字形与行距设定

(2). 缓存器内容描述

状态缓存器 STATUS Register(RS=1, WR=1)

Bit	说　明	Access
7	**内存写入忙碌（Memory Write Busy）旗标** 0：非忙碌 1：忙碌：当于字型写入内存或内存清除动作时，此旗标为 "high"。	R
6	**扫瞄忙碌（Scan Busy）旗标** 0：非忙碌 1：当驱动扫瞄逻辑非为闲置时（例: XCK 为 active 时）， SCAN_BUSY 为 "high"。	R
5	**睡眠状态（SLEEP）** 0：正常模式 1：睡眠模式	R
4-3	**保留**	R
2	**唤醒（Wakeup）状态** （和 REG[0Fh] Bit-2 相同）	R
1	**键盘扫瞄（KS）状态** （和 REG[0Fh] Bit-1 相同）	R
0	**触控扫瞄（TP）状态** （和 REG[0Fh] Bit-0 相同）	R

REG [00h] Whole Chip LCD Controller Register（**WLCR**）

Bit	说　明	初始值	Access
7	**电源模式（Power Mode）** 0：正常模式 ➔ RA8806 于此模式下所有功能皆可使用。 1：睡眠模式 ➔ RA8806 于睡眠模式下，除了唤醒（Wake-up）电路工作外，其它功能都被关闭，若唤醒电路被触发，RA8806 则回到正常模式。	0	R/W
6	**线性译码模式（Linear Decode mode）** 此位为用来定义 Font ROM 地址线的译码规则。标准产品被设定为“low”。当使用者要创造一个新的 Font Code 地址对应时，则设定为“high”来实现此特殊的应用。 0：BIG5/GB ROM 地址对应规则。 1：使用者自行定义 ROM 的地址对应规则。	0	R/W
5	**软件重置（Software Reset）** 0：正常模式 1：除了显示数据存储器（DDRAM）的数据外，所有缓存器的数据都被重置（只有在正常模式下动作），当此位被设定为“high”时，要给 RA8806 的 MPU 周期（cycle）至少需等待 3 个 clock 周期的时间。	0	R/W
4	**保留**	0	R
3	**选择文字工作模式（Text Mode Selection）** 0：绘图模式 ➔ 写入的数据会被视为是 Bit-Map 的模式。 1：文字模式 ➔ 写入的资料会被视为是 GB/BIG/ASCII 等字码。	0	R/W
2	**选择屏幕显示为开启或关闭（Set Display On/Off Selection）** 此位用来控制连接到 LCD 驱动器接口的“DISP_OFF”讯号。 0：DISP_OFF 输出“low”（屏幕显示关闭）。 1：DISP_OFF 输出“high”（屏幕显示开启）。	0	R/W
1	**屏幕闪烁模式选择（Blink Mode Selection）** 0：正常显示。 1：整个屏幕闪烁。用缓存器 BTMR 来设定闪烁周期。	0	R/W
0	**屏幕反白模式选择（Inverse Mode Selection）** 0：正常显示。 1：整个屏幕反白显示。 将使显示出来的数据反向。	0	R/W

REG [01h] Misc. Register（**MISC**）

Bit	说　明	初始值	Access
7	**雪花消除（Eliminating Flicker）** 1：雪花消除模式，当忙碌时扫瞄将会自动暂停。 0：正常模式。	0	R/W
6	**Clock 输出（Pin CLK_OUT）控制** 1：CLK_OUT 此脚位代表状态缓存器的睡眠状态。（0: 正常模式 1: 睡眠模式） 0：CLK_OUT 此脚位输出系统频率（System Clock）。	0	R/W
5	**设定忙碌触发准位（Busy Polarity for "BUSY" pin）** 1：设定为高电位触发动作。 0：设定为低电位触发动作。	0	R/W
4	**设定中断触发准位（Interrupt Polarity for "INT" pin）** 1：设定为高电位触发动作。 0：设定为低电位触发动作。	0	R/W
3-2	**驱动器 clock 选择（Driver Clock Selection）** 此二位为用来选择 XCK 的频率。 0 0：XCK = CLK/8 0 1：XCK = CLK/4（初始值） 1 0：XCK = CLK/2 1 1：XCK = CLK "CLK" 代表系统频率。	01	R/W
1	**SEG 扫瞄方向（SEG Scan Direction（SDIR））** 0：SEG 扫瞄顺序为 0 ~ 319。 1：SEG 扫瞄顺序为 319 ~ 0。	0	R/W
0	**COM 扫瞄方向（COM Scan Direction（CDIR））** 0：COM 扫瞄顺序为 0 ~ 239。 1：COM 扫瞄顺序为 239 ~ 0。	0	R/W

REG [03h] Advance Display Setup Register（**ADSR**）

Bit	说　明	初始值	Access
7	**卷动功能暂停选择（Scroll Function Pending）** 1：卷动功能暂停 0：卷动功能动作 **注**：当 SCR_HV（Bit-1）和 SCR_EN（Bit-0）被改变时，此功能不支持。	0	R/W
6-4	**保留**	000	R

3	**设定驱动数据输出位顺序（BIT_ORDER）** 1：反向驱动数据输出位顺序（Bit-7 to Bit-0，Bit-6 to Bit-1 依续到 Bit-0 to Bit-7。） 0：正常模式。	0	R/W
2	**卷动方向选择（SCR_DIR）** 当 SCR_HV = 0 时（水平卷动） 0：从左到右卷动。 1：从右到左卷动。 当 SCR_HV = 1 时（垂直卷动） 0：从上到下卷动。 1：从下到上卷动。	0	R/W
1	**水平/垂直卷动方向选择（SCR_HV）** 0：Segment 卷动（水平）。 1：Common 卷动（垂直）。	0	R/W
0	**卷动致能（SCR_EN）** 1：卷动功能开启。 0：卷动功能关闭。	0	R/W

REG [0Fh] Interrupt Setup and Status Register（**INTR**）

Bit	说　明	初始值	Access
7	**保留**	0	R
6	**唤醒（Wakeup）中断屏蔽** 1：致能唤醒中断。 0：禁能唤醒中断。	0	R/W
5	**键盘扫瞄（Key-Scan）中断屏蔽** 1：致能键盘扫瞄中断。 0：禁能键盘扫瞄中断。	0	R/W
4	**触控扫瞄（Touch Panel）中断屏蔽** 1：当触控扫瞄侦测到输入讯号时，产生中断输出讯号。 0：当触控扫瞄侦测到输入讯号时，不产生中断输出讯号。	0	R/W
3	**触控扫瞄触发**（只有在手动模式下有效） 1：触控扫瞄侦测到输入讯号。 0：触控扫瞄没有侦测到输入讯号。	0	R
2	**唤醒中断状态位** 1：当从睡眠模式中唤醒而产生的中断。 0：没有唤醒中断产生。 使用者必须写 "0" 来清除此状态位。	0	R/W

1	**键盘扫瞄中断状态位** 1：键盘扫瞄侦测到键盘输入讯号。 0：键盘扫瞄没有侦测到键盘输入讯号。 使用者必须写 "0" 来清除此状态位。	0	R/W
0	**触控扫瞄侦测状态位** 1：触控扫瞄侦测到输入讯号。 0 :触控屏幕没有侦测到输入讯号。 使用者必须写 "0" 来清除此状态位。	0	R/W

REG [10h] Whole Chip Cursor Control Register（**WCCR**）

Bit	说　明	初始值	Access
7	**CUR_INC（当对 DDRAM 作读写操作时，光标位置自动增加）** 1：禁能。 0：致能（自动增加）。	0	R/W
6	**FULL_OFS（全型和半型字符对齐）** 1：致能，当于全型和半型混和模式时，中文字都对齐于全型字的起始位置。 0：禁能。	0	R/W
5	**反向写入数据模式** 0：直接把目前资料写入 DDRAM。 1：反向地将目前资料写入 DDRAM。（例如: 01101101➔ 10010010）	0	R/W
4	**粗体字（只有在文字模式时生效）** 1：粗体字。 0：正常字。	0	R/W
3	**文字旋转模式（T90DEG）** 1：文字旋转 90 度（参照第 6-10-4 节 "文字垂直显示"） 0：正常字。	0	R/W
2	**光标显示** 1：设定光标为显示。 0：设定光标为不显示。	0	R/W
1	**游标闪烁** 1：游标闪烁。（REG BTMR 决定光标闪烁的周期） 0：游标不闪烁。	0	R/W
0	**保留**	0	R

REG [11h] Cursor Height and Word Interval Register（**CHWI**）

Bit	说　明	初始值	Access
7-4	**设定光标高度** 0000 b → 游标高度为 1 pixel。 0001 b → 游标高度为 2 pixels。 0010 b → 游标高度为 3 pixels。 ⋮ 1111 b → 游标高度为 16 pixels。 **注**：在正常模式光标的宽度固定为 8 pixels，光标的高度由 Bit[7:4] 决定。文字垂直旋转模式，光标的高度固定为 16 pixels，光标的宽度由 Bit[6:4] 决定。	0000	R/W
3-0	**设定行与行间的间距** 0000 b → 间距为 1 pixel。 0001 b → 间距为 2 pixels。 0010 b → 间距为 3 pixels。 ⋮ 1111 b → 间距为 16 pixels。	0000	R/W

REG [12h] Memory Access Mode Register（**MAMR**）

Bit	说　明	初始值	Access
7	**光标自动移动方向** 0：光标先由水平方向（从左到右）移动，再垂直方向（从上到下）移动。 1：光标先由垂直方向移动，再水平方向移动。 **注：**于绘图模式下，水平方向光标为以 byte 为单位移动，而垂直方向为以 bit 为单位移动。当于文字模式下，此位可被忽略，光标的移动方向一定为水平方向移动。	0	R/W
6-4	**显示图层和显示模式选择** 0 0 0：灰阶模式。在此模式下，每一显示位包含了内存中的二笔连续的数据，此 4 灰阶是依 FRC 的方法达成，此显示位的配置如下： bit1　bit0　灰阶 0　0　Level1（最亮） 0　1　Level2 1　0　Level3 1　1　Level4（最暗） **注：**于灰阶模式下没有支持文字输入。 0 0 1：将 DDRAM1 的数据显示于屏幕上。 0 1 0：将 DDRAM2 的数据显示于屏幕上。 0 1 1：双图层显示模式。显示规则依底下的 Bit-3 和 Bit-2 。 1 0 X：NA。 1 1 0：扩展模式（1），将 DDRAM1 和 DDRAM2 的数据显示于屏幕上。RA8806 于此模式下支持 640x240 的显示屏幕。 1 1 1：扩展模式（2），将 DDRAM1 和 DDRAM2 的数据显示于屏幕上。 RA8806 于此模式下支持 320x480 的显示屏幕。	001	R/W

3-2	**双图层显示规则选择** 当 Bit[6:4] 被设定为 "011" 时，RA8806 将结合 DDRAM1 和 DDRAM2 的数据来显示于屏幕上。 0 0 : DDRAM1 "OR" DDRAM2。 0 1 : DDRAM1 "XOR" DDRAM2。 1 0 : DDRAM1 "NOR" DDRAM2。 1 1 : DDRAM1 "AND" DDRAM2。	00	R/W
1-0	**MPU 读取/写入图层选择** 0 0 : 存取 CGRAM（512Byte）。 0 1 : 存取 DDRAM1。 1 0 : 存取 DDRAM2。 1 1 : 同时存取 DDRAM1 和 DDRAM2。	01	R/W

REG [20h] Active Window Right Register（**AWRR**）

Bit	说　明	初始值	Access
7-6	**保留**	00	R
5-0	**设定工作窗口（Active Window）右边边界 → Segment-Right** **注 :** AWRR 必须大于或等于 AWLR，且值须小于或等于 27h。	27h	R/W

注 :

REG[20h，30h，40h 和 50h] 用来控制写入数据时，行与列在工作窗口内的变化，使用者可以使用此四个缓存器来设定工作窗口的上/下/左/右边界，当写入的数据超过右边的边界时，光标会自动跳到下一列（Line）来写入数据，也就是说，光标会移动到工作窗口左边的边界，当数据写到所设定之右边且下方的边界时，下一笔数据写入将使光标移动到所设定之左上方边界位置。

REG [21h] Display Window Width Register（**DWWR**）

Bit	说　明	初始值	Access
7-6	**保留**	00	R
5-0	**设定显示窗口（Display Window）宽度→ Segment-Width** **Segment-Right =（Segment Number / 8）– 1** 假设 LCD 的分辨率为 320x240 时，此缓存器应被设定为: （320 / 8）- 1 = 39 = 27h	27h	R/W

注 :

REG[21h，31h] 用来设定显示窗口的分辨率，使用者可以设定显示内存的可视范围。RA8806 的 Column 宽度（DWWR）可被设定在 0h ~ 27h 之间，且 Row 高度（DWHR）可被设定在 0h ~ EFh 之间。

REG [30h] Active Window Bottom Register（**AWBR**）

Bit	说　明	初始值	Access
7-0	**设定工作窗口（Active Window）下方边界→ Common-Bottom** **注：**AWBR 必须大于或等于 AWTR，且值须小于或等于 EFh。	EFh	R/W

REG [31h] Display Window Height Register（**DWHR**）

Bit	说　明	初始值	Access
7-0	**设定显示窗口（Display Window）高度→ Common- Height** **Common_ Height = LCD Common Number –1** 假设 LCD 的分辨率为 320x240 时，此缓存器应被设定为: 240 – 1 = 239 = EFh	EFh	R/W

REG [40h] Active Window Left Register（**AWLR**）

Bit	说　明	初始值	Access
7-6	**保留**	00	R
5-0	**设定工作窗口（Active Window）左边边界→ Segment-Left** **注：**AWLR 必须小于或等于 AWRR，且值须小于 27h。	00h	R/W

REG [50h] Active Window Top Register（**AWTR**）

Bit	说　明	初始值	Access
7-0	**设定工作窗口（Active Window）上方边界→ Common-Top** **注：**AWTR 必须小于或等于 AWBR，且值须小于 EFh。	00h	R/W

REG [60h] Cursor Position X Register（**CURX**）

Bit	说　明	初始值	Access
7-6	**保留**	00	R
5-0	**设定光标 Segment 位置/ RAM0 地址[4:0]** 定义光标 segment 的位置，其值在 0h ~ 27h 之间。 当被设定为 CGRAM 写入模式时（REG[12h] Bit[1:0] = 00b），此缓存器 Bit[4:0] 为用来写入数据的位对应地址。于创造全型字时，通常设定为 0h，而当要创造奇数个半型字时，通常设定为 0h，创造偶数个半型字时，通常设定为 10h。	00h	R/W

REG [61h] Begin Segment Position Register of Scrolling（**BGSG**）

Bit	说　明	初始值	Access
7-6	**保留**	00	R
5-0	**设定于卷动模式下 Segment 的起始位置** REG[61h] 定义卷动窗口的起始位置（左边边界），其值必须小于或等于缓存器 REG[62h]（定义卷动窗口终点位置（右边边界））所设定的值。此外，对应到显示内存的限制，其值必须小于 27h。	00h	R/W

注：

REG[61h，62h，71h 和 72h] 是设定卷动的窗口，这些缓存器必须在把卷动功能打开前先设定完成。

REG [62h] End Segment Position Register of Scrolling（**EDSG**）

Bit	说　明	初始值	Access
7-6	**保留**	00	R
5-0	**设定于卷动模式下 Segment 的终点位置** REG[62h] 定义卷动窗口的终点位置（右边边界），其值必须大于或等于缓存器 REG[61h]（定义卷动窗口起始位置（左边边界））所设定的值。此外，对应到显示内存的限制，其值必须小于或等于 27h。	00h	R/W

REG [70h] Cursor Position Y Register（**CURY**）

Bit	说　明	初始值	Access
7-0	**设定光标 Common 位置/ RAM0 地址[8:5]** 定义光标 common 的位置，其值在 0h ~ EFh 之间。 当被设定为 CGRAM 写入模式时（REG[12h] Bit[1:0] = 00b），此缓存器 Bit[3:0] 为用来指定哪一个字被创造，缓存器 Bit[7:4] 没有使用到。	00h	R/W

REG [71h] Scrolling Action Range Begin Common Register（**BGCM**）

Bit	说　明	初始值	Access
7-0	**设定卷动模式下 Common 的起始位置** REG[71h] 定义卷动窗口的起始位置（上方边界），其值必须小于或等于缓存器 REG[72h]（定义卷动窗口终点位置（下方边界））所设定的值。此外，对应到显示内存的限制，其值必须小于 EFh。	00h	R/W

REG [72h] Scrolling Action Range END Common Register（**EDCM**）

Bit	说　明	初始值	Access
7-0	**设定卷动模式下 Common 的终点位置** REG[72h] 定义卷动窗口的终点位置（下方边界），其值必须大于或等于缓存器 REG[71h]（定义卷动窗口起始位置（上方边界））所设定的值。此外，对应到显示内存的限制，其值必须小于或等于 EFh。	00h	R/W

REG [80h] Blink Time Register（**BTMR**）

Bit	说　明	初始值	Access
7-0	**设定光标闪烁和卷动时间周期** **闪烁时间周期 = Bit[7:0] x（Frame width）** **Frame width = 1/Frame Rate** Frame Rate 依照 DWWR、DWHR 和 ITCR 所设定的值来决定。	00h	R/W

REG [90h] Idle Time Counter Register（**ITCR**）

Bit	说　明	初始值	Access
7-0	**空闲时间（idle time）设定，依照系统频率来计数** 此值用来决定每个 LCD COM 的扫瞄时间。 **COM_PRD =（COM_SCAN + ITCR）x XCK_PRD** 于此之中， **COM_SCAN =（SEG_NO/LD_WIDTH）x（1 + EXT_MD）** **XCK_PRD = 1 / XCK** **COM_PRD:** 每个 COM 的最后扫瞄周期（Unit：ns）。 **COM_SCAN:** 每个 COM 的原始扫瞄周期。 **XCK_PRD:** 一个 XCK 的周期时间。XCK 的周期依照系统频率（system clock）和缓存器 REG[01h] Bit[3:2] 所设定的值来决定。假设系统频率为 8MHz，缓存器 REG[01h] Bit[3:2] 设定为 10b，则 XCK_PRD = 250ns。 **SEG_NO:** Segment 数目，如 240x160 的屏，SEG_NO = 240。 **EXT_MD:** 在扩展模式 1 或 2 下（REG[12h] Bit[6:4] = 111b 或 110b），EXT_MD = 1，除此之外 EXT_MD = 0。 **LD_WIDTH:** 驱动接口数据总线宽度。假设 LCD 驱动数据总线宽度为 4-bits 时，则 LD_WIDTH = 4，假设 LCD 驱动数据总线宽度为 8-bits 时，则 LD_WIDTH = 8。请参照第 4-2 节脚位 "DW" 的描述。	00h	R/W

REG [A0h] Key-Scan Control Register 1（**KSCR1**）

Bit	说　明	初始值	Access
7	**设定键盘扫瞄功能开启或关闭** 1：开启。 0：关闭。	0	R/W
6	**选择键盘扫瞄矩阵** 1：4x8 Matrix（使用 KOUT[3:0]，KOUT[7:4] 请保持浮接） 0：8x8 Matrix（使用 KOUT[7:0]）	0	R/W
5-4	**设定键盘扫瞄 De-bounce 取样的次数（Sampling Times）** 0 0：4 次 0 1：8 次 1 0：16 次 1 1：32 次	00	R/W
3	**设定长按键功能开启或关闭（LNGKEY_EN）** LNGKEY_EN = 0 → 长按键功能关闭。 LNGKEY_EN = 1 → 长按键功能开启。	0	R/W
2-0	**设定键盘扫瞄频率（KF2-0）** 假设系统频率为 10MHz，则键盘扫瞄频率的关系如下:	000	R/W

KF2	KF1	KF0	Key-Scan Pulse Width (KOUT period)	Key-Scan Cycle(4x8)	Key-Scan Cycle(8x8)
0	0	0	16μs	64μs	128μs
0	0	1	32μs	128μs	256μs
0	1	0	64μs	256μs	512μs
0	1	1	128μs	512μs	1.024ms
1	0	0	256μs	1.024ms	2.048ms
1	0	1	512μs	2.048ms	4.096ms
1	1	0	1.024ms	4.096ms	8.192ms
1	1	1	2.048ms	8.192ms	16.384ms

REG [A1h] Key-Scan Controller Register 2（**KSCR2**）

Bit	说　明	初始值	Access
7	**设定键盘扫瞄唤醒功能开启或关闭** 0: 键盘扫瞄唤醒功能关闭。 1: 键盘扫瞄唤醒功能开启。	0	R/W
6-4	**保留**	000	R
3-2	**长按键时间调整** 00 : 约 0.625 sec 01 : 约 1.25 sec 10 : 约 1.875 sec 11 : 约 2.5 sec **注：**以上时间是假设系统频率为 8MHz。	00	R/W
1-0	**告知几个按键被按到** 00 : 没有按键被按到。 01 : 一个按键被按到，读取缓存器 REG[A2h] 来获取按键值。 10 : 二个按键被按到，读取缓存器 REG[A2h~A3h] 来获取按键值。 11 : 三个按键被按到，读取缓存器 REG[A2h~A4h] 来获取按键值。	00	R

REG [A2h ~ A4h] Key-Scan Data Register（**KSDR0 ~ 2**）

Bit	说　明	初始值	Access
7-0	**按键撷取数据** 代表所按到的键对应之值。请参照第 6-5 节 "键盘扫瞄功能"。	00h	R

REG [B0h] Memory Write Command Register（**MWCR**）

Bit	说　明	初始值	Access
7-0	**内存写入指令（从光标位置）** **注：**当要写数据到内存时，使用者必须先下 MWCR（Command Write cycle）指令后，再写数据进去（Data Write cycle）。	NA	R/W

REG [B1h] Memory Read Command Register（**MRCR**）

Bit	说　明	初始值	Access
7-0	**内存读取指令（从光标位置）** **注：**于内存读取周期，光标的移动在文字模式下的行为和在绘图模式下一样。B1h 将作预读的行为，故在执行完 MRCR 指令后，光标位置会增加。	NA	R/W

REG [C0h] Touch Panel Control Register 1（**TPCR1**）

Bit	说　明	初始值	Access
7	**触控扫瞄功能开启或关闭** 1：开启。 0：关闭。	0	R/W
6-4	**触控扫瞄取样时间调整** 000：等待 50μs 001：等待 100μs 010：等待 200μs 011：等待 400μs 100：等待 800μs 101：等待 1.6ms 110：等待 3.2ms 111：等待 6.4ms **注：**当触控屏幕被接触到时，为了避免讯号还不稳定，故延迟一段取样时间來等待讯号变稳定，而此处的触控扫瞄取样时间与触控扫瞄频率（ADC Clock）转换速度有相对的关系，相关建议值请参考第6-4-3节。	000	R/W
3	**触控扫瞄唤醒开启或关闭** 1：触控扫瞄开启可以唤醒睡眠模式。（触控扫瞄功能必须是开启的状态下） 0：触控扫瞄关闭唤醒睡眠模式。	0	R/W
2-0	**触控扫瞄频率（ADC Clock）转换速度**。“CLK”代表系统频率。 0 0 0：CLK / 4 0 0 1：CLK / 8 0 1 0：CLK /16 0 1 1：CLK / 32 1 0 0：CLK / 64 1 0 1：CLK / 128 1 1 0：CLK / 256 1 1 1：CLK / 512	000	R/W

REG [C1h] Touch Panel X High Byte Data Register（**TPXR**）

Bit	说　明	初始值	Access
7-0	**触控扫瞄 X 资料 Bit[9:2]（Segment）**	00h	R

REG [C2h] Touch Panel Y High Byte Data Register（**TPYR**）

Bit	说　明	初始值	Access
7-0	**触控扫瞄 Y 资料 Bit[9:2]（Common）**	00h	R

REG [C3h] Touch Panel Segment/Common Low Byte Data Register（**TPZR**）

Bit	说　明	初始值	Access
7-4	**保留**	0000	R
3-2	**触控扫瞄 Y 资料 Bit[1:0]（Common）**	00	R
1-0	**触控扫瞄 X 资料 Bit[1:0]（Segment）**	00	R

REG [C4h] Touch Panel Control Register 2（**TPCR2**）

Bit	说　明	初始值	Access
7	**触控扫瞄手动模式开启或自动模式** 1：使用手动模式。 0：使用自动模式。	0	R/W
6-2	**保留**	00h	R
1-0	**触控扫瞄手动模式程序选择** 00：IDLE 模式: 触控扫瞄闲置（ADC Idles）。 01 ： 等待触控屏幕被接触，触控扫瞄电路将产生中断讯号或是从缓存器 REG[0Fh] Bit-3 读出状态。 10：栓锁住 X 数据，在此期间，X 数据将被栓锁在缓存器 REG[C1h] 和 REG[C3h] 里。 11：栓锁住 Y 数据，在此期间，Y 数据将被栓锁在缓存器 REG[C2h] 和 REG[C3h] 里。	00	R/W

REG [D0h] PWM Control Register（**PCR**）

Bit	说　明	初始值	Access
7	**脉波宽度调变（PWM）开启或关闭** 1：开启。 0：关闭。于此状态下，PWM_OUT 准位依照此缓存器 Bit-6 来决定。	0	R/W
6	**PWM 关闭时的准位** 0：当 PWM 关闭或于睡眠模式时，PWM_OUT 一般为 "low" 状态。 1：当 PWM 关闭或于睡眠模式时，PWM_OUT 一般为 "high" 状态。	0	R/W

5-4	**保留**	00	R
3-0	**PWM 电路所接受的 Clock 来源速度选择** 0000 b → CLK / 1 0001 b → CLK / 2 0010 b → CLK / 4 0011 b → CLK / 8 : : 1111 b → CLK / 32768 "CLK" 代表系统频率，例如: CLK 为 8MHz: 0000 b → PWM clock source = 8MHz 0001 b → PWM clock source = 4MHz : : 1111 b → PWM clock source = 256Hz	0000	R/W

REG [D1h] PWM Duty Cycle Register（**PDCR**）

Bit	说　明	初始值	Access
7-0	**PWM 责任周期（Cycle Duty）选择** 00h → 1 / 256 01h → 2 / 256 High period 02h → 3 / 256 High period : : FFh → 256 / 256 High period	00h	R/W

REG [E0h] Pattern Data Register（**PNTR**）

Bit	说　明	初始值	Access
7-0	**要写入 DDRAM 里的资料（Display Data RAM）** 当缓存器 REG[F0h] Bit-3 被填为 "1" 时（内存清除模式），此缓存器的数据将填满整个工作窗口。	00h	R/W

REG [F0h] Font Control Register（**FNCR**）

Bit	说　明	初始值	Access
7	**ISO8859 模式** 0：关闭。 ASCII 区块 1~4 的内容为附录C内的表C- 1 ~ 表C- 4 所示。 1：开启。 ASCII 区块 1~4 的内容为代表标准的ISO8859-1 ~ 4，且为附录C内表C- 5 ~ 表C- 8 所示。	0	R/W
6-4	**保留**	000	R

3	**内存清除功能** 对此位作写入时代表 0 : 不动作 1 : 内存清除功能开启，将 FNTR 数据填满整个工作窗口。 对此位作读取时代表 0 : 内存清除动作已完成。 1 : 内存清除动作尚未完成。 当此位设定为 "1" 时，RA8806 将自动读取缓存器 PNTR 的数据，然后将此数据填满整个工作窗口（工作窗口范围: [AWLR，AWTR] ~ [AWRR，AWBR]），而当填满动作完后，此位自动清除为 "0"。	0	R/W
2	**ASCII 模式选择** 1 : 所有的输入数据将译码为 ASCII（00h ~ FFh）。 0 : 在文字模式下（REG[00h] Bit-3 = 1），RA8806 将会先检查被写入数据的第一个字节（byte）。当此字节小于 80h 时，将把此笔数据当成 ASCII（半型字）来解码，反之，则当成文字（全型字的 GB、BIG-5 或是使用者自创字型）来译码。	0	R/W
1-0	**ASCII 区块选择** 0 0 : 对应到 ASCII block 1。（附录C的表C- 1 和表C- 5） 0 1 : 对应到ASCII block 2。（附录C的表C- 2 和表C- 6） 1 0 : 对应到ASCII block 3。（附录C的表C- 3 和表C- 7） 1 1 : 对应到ASCII block 4。（附录C的表C- 4 和表C- 8）	00	R/W

REG [F1h] Font Size Control Register（**FVHT**）

Bit	说　明	初始值	Access
7-6	**设定字符水平放大倍率** 0 0 : 原本字符宽度。 0 1 : 字符宽度放大为原本字符宽度的二倍。 1 0 : 字符宽度放大为原本字符宽度的三倍。 1 1 : 字符宽度放大为原本字符宽度的四倍。	00	R/W
5-4	**设定字符垂直放大倍率** 0 0 : 原本字符高度。 0 1 : 字符高度放大为原本字符高度的二倍。 1 0 : 字符高度放大为原本字符高度的三倍。 1 1 : 字符高度放大为原本字符高度的四倍。	00	R/W
3-0	**保留**	0000	R

(3) 功能描述

1. 分辨率设定

RA8806 可支持数种不同分辨率（Resolution）的显示，如表 6-5 所示。在使用不同分辨率的LCD Panel，使用者必须去设定显示窗口（Display Window）大小相关的缓存器，例如：DWWR与DWHR。除此之外，使用者亦可设定AWRR，AWBR，AWLR和AWTR等缓存器来设定工作窗口（Active Window）之边界。

举例来说，如果 Panel Resolution 为 320x240，则相关的缓存器设定如下：

显示窗口宽度（DWWR）=（**320** / 8）- 1 = 39 = 27h

显示窗口高度（DWHR）= **240** – 1 = 239 = EFh

在应用上，使用者必须注意，工作窗口的范围通常是比显示窗口来得小，如下所示：

1. 显示窗口宽度（DWWR）≥ 工作窗口右边界（AWRR）≥ 工作窗口左边界（AWLR）
2. 显示窗口高度（DWHR）≥ 工作窗口下边界（AWBR）≥ 工作窗口上边界（AWTR）

RA8806 可支持各式各样的LCD模块，表 6-5 列出几种较为大家所常用的LCD模块及其相关缓存器设定。

表 6-5：常用 LCD 模块之显示窗口设定

Panel Resolution	Segment	Common	REG[21h] DWWR	REG[31h] DWHR
160*80	160	80	13h	4Fh
160*128	160	128	13h	7Fh
160*160	160	160	13h	9Fh
240*64	240	64	1Dh	3Fh
240*128	240	128	1Dh	7Fh
240*160	240	160	1Dh	9Fh
320*240	320	240	27h	EFh

表 6-6

Reg.	Bit_Num	说　明	缓存器编号
AWLR	Bit [5:0]	定义工作窗口之左边界。	REG[40h]
AWRR	Bit [5:0]	定义工作窗口之右边界。	REG[20h]
AWTR	Bit [7:0]	定义工作窗口之上边界。	REG[50h]
AWBR	Bit [7:0]	定义工作窗口之下边界。	REG[30h]
DWWR	Bit [5:0]	定义显示窗口之宽度。	REG[21h]
DWHR	Bit [5:0]	定义显示窗口之高度。	REG[31h]

(2) 显示窗口与工作窗口

实际应用上，RA8806 提供两种窗口，分别是显示窗口（Display Window）和工作窗口（Active Window）。显示窗口所表示的就是「实际液晶显示屏的分辨率」，亦即当液晶显示屏分辨率为 320x240 时，就表示显示窗口的大小也必须为 320x240（REG[21h] = 27h，REG[31h] = EFh）。而工作窗口则是比显示窗口还小的窗口，举凡光标移动、换行、换页都是以工作窗口的边界为基准。这两个窗口之相关缓存器如上表 6-6。

图 6-12 表示显示窗口与工作窗口之间的关系。毫无疑问的，当液晶显示屏分辨率为 320x240 时，就表示显示窗口的大小也是 320x240，在图 6-12 中，我们设定一个 160x160 的工作窗口，其相关缓存器的设定如下所示：

```
LCD_CmdWrite ( 0x40 );     // AWLR = 09h = 9 → ( 80 / 8) – 1
LCD_DataWrite ( 0x09 );
LCD_CmdWrite ( 0x20 );     // AWRR = 1Dh = 29 → ( 240 / 8 ) – 1
LCD_DataWrite ( 0x1D );
LCD_CmdWrite ( 0x50 );     // AWTR = 00h = 0
LCD_DataWrite ( 0x00 );
LCD_CmdWrite ( 0x30 );     // AWBR = 9Fh = 159 → 160 – 1
LCD_DataWrite ( 0x9F );
LCD_CmdWrite ( 0x40 );     // DWWR = 27h = 39 → ( 320 / 8 ) – 1
LCD_DataWrite ( 0x27 );
LCD_CmdWrite ( 0x40 );     // DWHR = EFh = 239 → 240 – 1
LCD_DataWrite ( 0xEF );
```

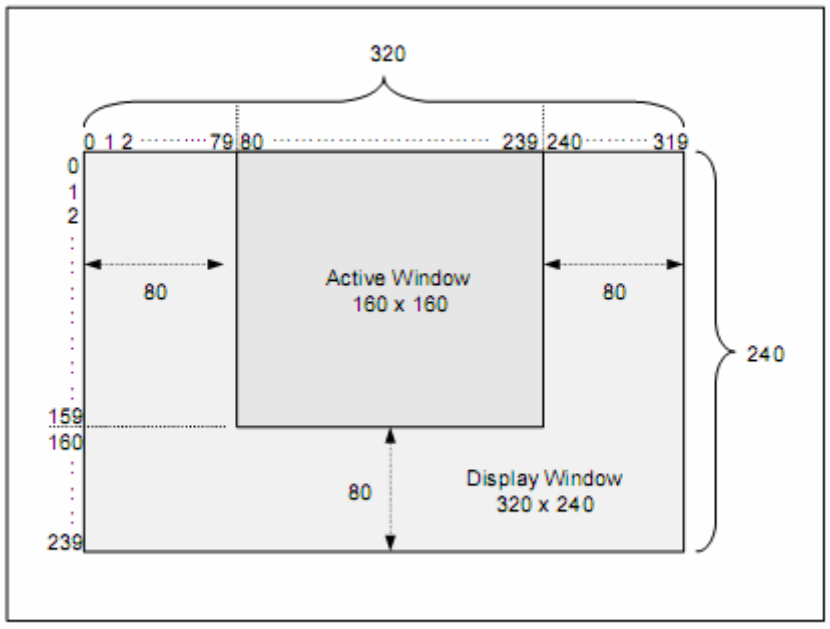


图 6-12 : RA8806 「显示窗口」与「工作窗口」

(3) COM/SEG 扫描方向

RA8806 有一很特殊的功能，亦即使用者可以反相 Common 和 Segment 的显示顺序。当使用 320x240 来进行显示时，如果将旋转 90 度（文字）的功能开启，同时反相 Common 的显示顺序，此即为「垂直显示」，也就是以 240x320 来进行显示。相关内容请参考第 6-10-4 节。

表 6-7

Reg.	Bit_Num	说　明	缓存器编号
MISC	Bit 1	定义 Segment 的显示顺序（SDIR）。	REG[01h]
	Bit 0	定义 Common 的显示顺序（CDIR）。	

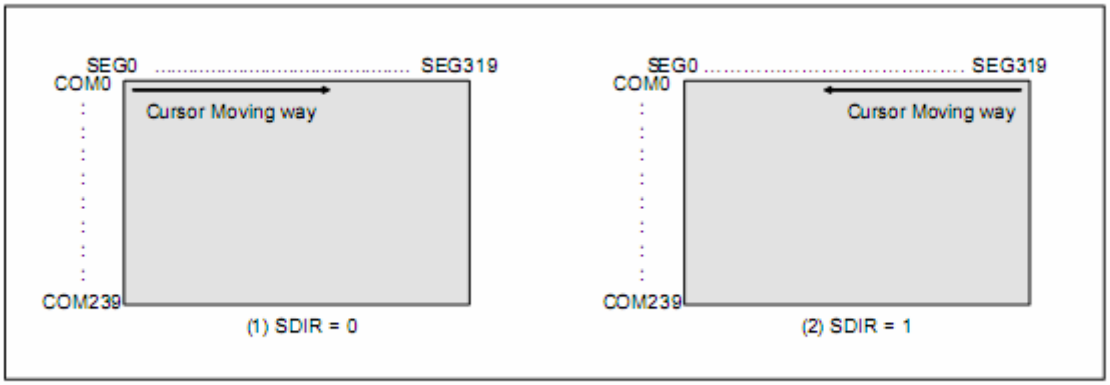


图 6-14 : 改变 Segment 显示顺序之范例

（4） 扫描闲置时间

RA8806 有一缓存器 ITCR，是被用来决定「每个 LP 信号之间的闲置时间」，进一步来说，它具有两项主要作用：

1. 可用来调整 Frame Rate（当 ITCR 的内容值愈大，表示每个 LP 信号之间的闲置时间就愈长，亦即 Frame Rate 愈小，反之则反）。
2. 可用来避免「雪花」（Flicker）问题的发生，提升画面显示的质量。

「雪花」的发生主要是因为当 LCD 在进行扫瞄显示的同时，恰巧 MPU 在对 Display RAM 写入数据而产生的冲突。因此，所谓的「雪花」是指发生冲突时的错误显示，使用者可藉由设定缓存器 ITCR，确保 MPU 只在每个 LP 信号之间的闲置时间来写入数据，以避免或减少雪花的发生。

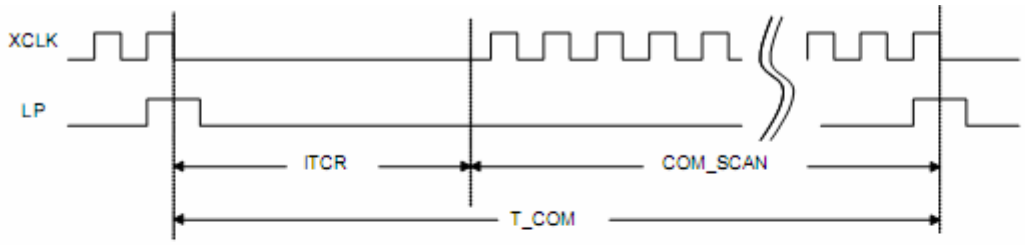


图 6-15 : LP 与 LP 信号之间的闲置时间

RA8806 每一个显示扫瞄线的时间长度的计算公式如下：

COM_PRD = ((SEG_NO/LD_WIDTH) x (1 + EXT_MD)) ＋ITCR) x XCK_PRD

其中 EXT_MD 是用来表示是否开启扩展模式功能，若 EXT_MD = 1，则为开启扩展模式功能，反之，若 EXT_MD = 0，则是关闭。XCK_PRD 为一个 XCK 频率的周期宽度，其中 XCK 频率频率为系统频率（System Clock）除频的结果，使用者可透过缓存器 MISC 的 Bit[3:2] 来进行设定。关于一个 Frame 的时间长度和 Frame Rate 的计算公式如下所示：

FRM_PRD = COM_PRD x COM#

和

FRM_Rate = 1 / FRM_PRD

举例来说，当 Panel Resolution 为 320x240，系统频率频率为 8MHz，缓存器 MISC 的 Bit[3:2] 设定为 10b 以及 LCD Driver 为 4 bit 的数据总线时，那么 Frame Rate 则为多少呢？

由于系统频率频率为 8MHz，而且缓存器 MISC 的 Bit[3:2] 被设定为 10b，那么一个 XCK 周期为 250 ns，如下所示：

XCK_PRD = 1 / (CLK/2) = 1/4MHz = 250ns

COM_PRD = (320 / 4 ＋ITCR) x XCK_PRD = (80+ ITCR) x 250(ns)

假设缓存器 ITCR 设定为 A0h（换算为十进制为 160）

COM_PRD = (80+160) x 250ns = 240 x 250ns = 60μs

另外，显示扫瞄线共有 240 条，因此，一个 Frame 的时间长度为：

FRM_PRD = 60μs x 240 = 14.4 ms

而 Frame Rate 则是一个 Frame 时间长度的倒数，如下：

Frame Rate = 1 / 14.4 ms = 69.4 Hz

由此可知缓存器ITCR和Frame Rate保有一定的关系，使用者可透过设定缓存器ITCR来调整 Frame Rate。在附录B中的表B- 1 到 表B- 3 有整理了各种Panel Resolution，因应不同的系统频

率和ITCR的设定，所计算出的Frame Rate一览表，适当地调整Frame Rate，可以改善显示的质量，但值得注意的是，显示质量的好坏同时亦与模块的设计和液晶本身的材料有关。

表 6-8

Reg.	Bit_Num	说　明	缓存器编号
ITCR	Bit [7:0]	定义每个 LP 信号之间的闲置时间之长度。	REG[90h]
MISC	Bit [3:2]	用来选择 XCK 的频率。	REG[01h]

（5） 显示数据存贮

RA8806 本身内建有两块容量为 9.6K 字节大小的显示数据存储器，分别是 DDRAM1 和 DDRAM2。它可用来做单色的显示或者四灰阶的显示，每一块显示数据存储器最大均支持 320x240 大小的显示，显示模式包括「文字模式」和「图形模式」。总之，RA8806 的诸多功能可让使用者既弹性又方便来进行各种显示。

这两个显示数据存储器有以下四种最常见的应用：

1. ***仅显示 DDRAM1 或 DDRAM2:*** 当仅使用其中一个 DDRAM 来进行显示时，另一个 DDRAM 则可以备用或者当成「使用者自建字型」的内存。相关内容请参考第 6-11 节 “使用者自创字型”。

2. ***双层显示模式:*** 在此一模式，可用来显示两个 DDRAM 画面合成的效果，使用者可以透过缓存器 [12h] 的 Bit[3:2] 来选择画面合成的模式，如下所示：

   - DDRAM1 “OR” DDRAM2
   - DDRAM1 “XOR” DDRAM2
   - DDRAM1 “NOR” DDRAM2
   - DDRAM1 “AND” DDRAM2

   相关内容请参考第 6-10-1-5 节 “双图层显示”。

3. ***四灰阶显示模式:*** 此模式下，LCD 上每一 Pixel 的灰度由存在 DDRAM 的每 2 个连续 Bit 来决定。

4. ***扩展显示模式:*** RA8806 支持两种扩展模式，包括：

   - 水平扩展模式（最大可显示 640x240 点）
   - 垂直扩展模式（最大可显示 320x480 点）

RA8806 内建一个 512 字节的「字型产生内存」（CGRAM）和两个 9.6K 字节的「显示数据存储器」（DDRAM）。其中 CGRAM 可用来储存造字的字型数据，而 DDRAM 可用来储存欲显示的数据，另外，当仅用一个 DDRAM 来进行显示时，另一个 DDRAM 亦可当成 CGRAM，来储存造字的字型数据。在应用上，至于微处理机（MPU）要对那一个内存进行存取（Access），使用者可透过缓存器 [12h] 的 Bit[1:0] 来进行设定。相关内容请参考第 5-2 节 "缓存器内容描述"。

表 6-9

Reg.	Bit_Num	说　明	缓存器编号
MAMR	Bit [6:4]	「显示层」与「显示模式」的选择。	REG[12h]
	Bit [3:2]	双层显示模式之选择。	
	Bit [1:0]	微处理机存取内存之选择。	

（6）触摸屏幕功能

RA8806 内建一组 10 位ADC和控制电路，以连接四线电阻式的触控屏幕。一般来说，电阻式的触控屏幕是由两层非常薄的电阻式屏幕所组成，如图 6-16。在两层屏幕中间有一小缝隙，当有外力施加在面板上的某一点时，两层电阻式屏幕将被触碰（touch），形成回路而导通。由于两层电阻式屏幕的端点含有电极（XL、XR、YU、YD），如图 6-17，因此，相对于触碰的位置，系统将侦测到一个XY的坐标值。

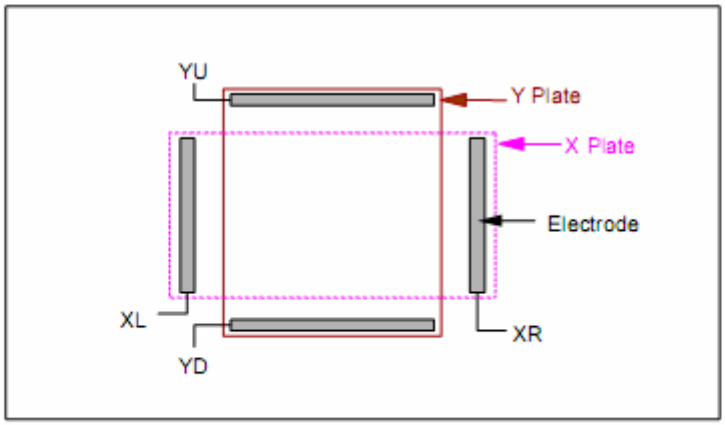


图 6-16：触控屏幕

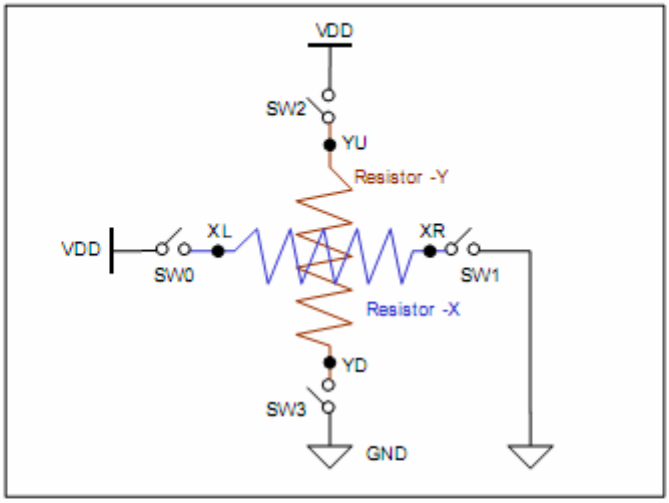


图 6-17：触控屏幕的控制开关

对使用者而言，应用触控屏幕的功能只需连接 XL、XR、YU 和 YD 等四条信号线到 RA8806 即可。系统就能不断监测，直到触控的事件（touch event）发生为止。当触控事件发生时，在屏幕电阻上所产生的分压将决定触控的所在位置。在 XY 的坐标值被传回系统（RA8806）并个别储存在特定的缓存器后，触控屏幕控制器（touch panel controller）将发出一中断告知微处理机（MPU）。

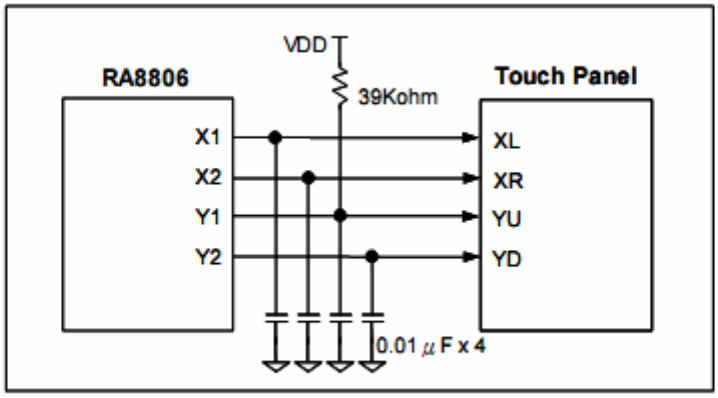


图 6-18：RA8806 触控屏幕电路

在触控屏幕功能的应用上，RA8806 提供两种操作模式，分别是「手动模式」和「自动模式」。如下表：

表 6-10

Operation mode	Event detection	说　明
Auto	Interrupt	当触控事件发生时，读回对应的 XY 坐标值。
Manual	Interrupt	当触控事件发生时，读回对应的 XY 坐标值。
	Polling	持续轮询触控事件，并读回对应的 XY 坐标值。

自动模式

自动模式是触控屏幕功能的应用当中最简单的。其原理与相关作法，请参考下列之流程图（Flow chart）。

(1) 流程图：

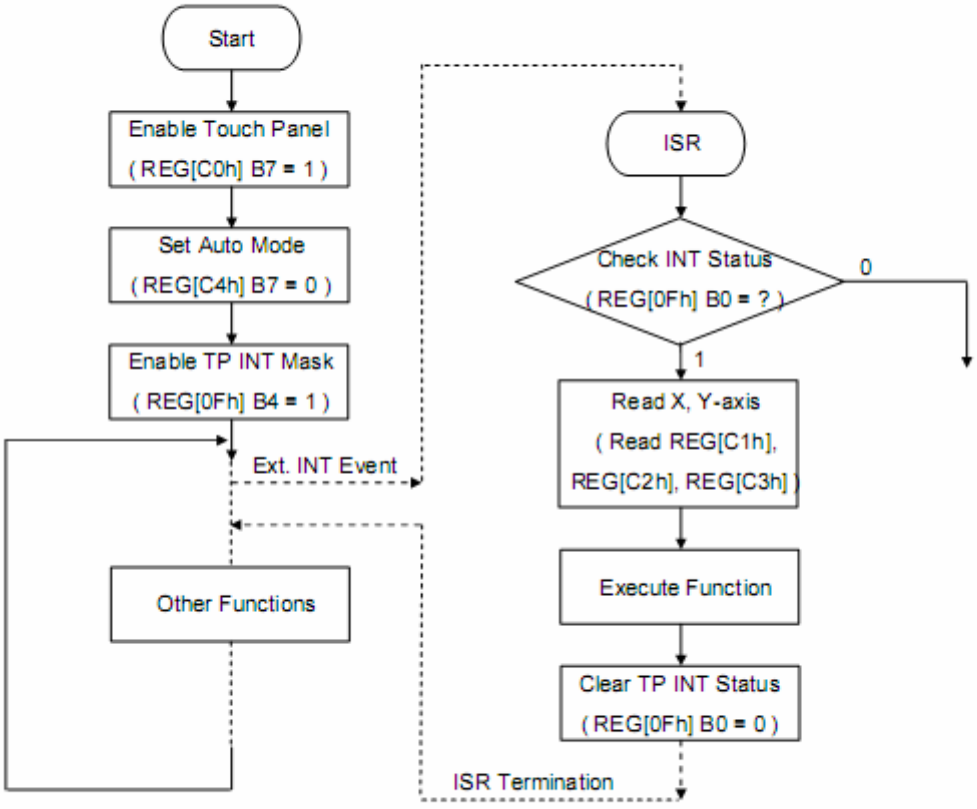


图 6-19：触控屏幕「自动模式」应用之流程图

自动模式应用之相关缓存器如表 6-11。

表 6-11

Reg.	Bit_Num	说　明	缓存器编号
TPCR1	Bit 7	触控屏幕功能的致能位。	REG[C0h]
TPCR2	Bit 7	用来选择「手动模式」或「自动模式」。	REG[C4h]
INTR	Bit 4	触控屏幕硬件中断的致能位。	REG[0Fh]
	Bit 0	触控事件之状态位。	
TPXR	Bit [7:0]	触控屏幕 X 轴数据高字节 Bit[9:2]。	REG[C1h]
TPYR	Bit [7:0]	触控屏幕 Y 轴数据高字节 Bit[9:2]。	REG[C2h]
TPZR	Bit [3:2]	触控屏幕 Y 轴数据低二位 Bit[1:0]。	REG[C3h]
	Bit [1:0]	触控屏幕 X 轴数据低二位 Bit[1:0]。	

手动模式

所谓「手动模式」是指从「侦测触控事件」到「闩锁 X data 与 Y data」以及「读出 XY 坐标值」的整个过程，都是由程序设计师以手动操作方式来完成。使用此一模式的优点在于，它给予程序设计师更弹性的应用空间。换句话说，在手动模式下，所有触控屏幕功能相关的缓存器都必须由程序设计师来设定，以软件（程序）控制的方法来实现触控屏幕应有之功能。

另外，根据不同的设计，使用者可以「外部中断告知模式」或「持续轮询模式」来侦测触控事件，其中之差异将陆续加以说明。

外部中断模式

在此一模式下，触控事件的侦测几乎和「自动模式」相同。其操作步骤如下所示：

1. 致能触控屏幕功能。
2. 切换触控屏幕的操作模式为「手动模式」。
3. 切换触控屏幕的相位为「等待触控事件发生」。
4. 当外部中断发生时，检查是否为触控事件所产生的中断。
5. 若是触控事件，则切换触控屏幕的相位为「闩锁 X data」（亦即设定缓存器 TPCR2[1:0] 为 10b），并等待足够长的时间，使 X data 能稳定地储存在缓存器 TPXR 和 TPZR。
6. 切换触控屏幕的相位为「闩锁 Y data」（亦即设定缓存器 TPCR2[1:0] 为 11b），并等待足够长的时间，使 Y data 能稳定地储存在缓存器 TPYR 和 TPZR。
7. 从 TPXR、TPYR 和 TPZR 读回 XY 坐标值，并清除中断的状态值。

有关「外部中断模式」之缓存器列表说明如下：

表 6-12

Reg.	Bit_Num	说　明	缓存器编号
TPCR1	Bit 7	触控屏幕功能的致能位。	REG[C0h]
TPCR2	Bit 7	用来选择「手动模式」或「自动模式」。	REG[C4h]
	Bit [1:0]	触控屏幕手动模式之选择位。	
INTR	Bit 4	触控屏幕硬件中断的致能位。	REG[0Fh]
	Bit 0	触控事件之状态位。	
TPXR	Bit [7:0]	触控屏幕 X 轴数据高字节 Bit[9:2] 。	REG[C1h]
TPYR	Bit [7:0]	触控屏幕 Y 轴数据高字节 Bit[9:2] 。	REG[C2h]
TPZR	Bit [3:2]	触控屏幕 Y 轴数据低二位 Bit[1:0] 。	REG[C3h]
	Bit [1:0]	触控屏幕 X 轴数据低二位 Bit[1:0] 。	

应用外部中断模式的流程图与范例程序如下所示。

(1) 流程图:

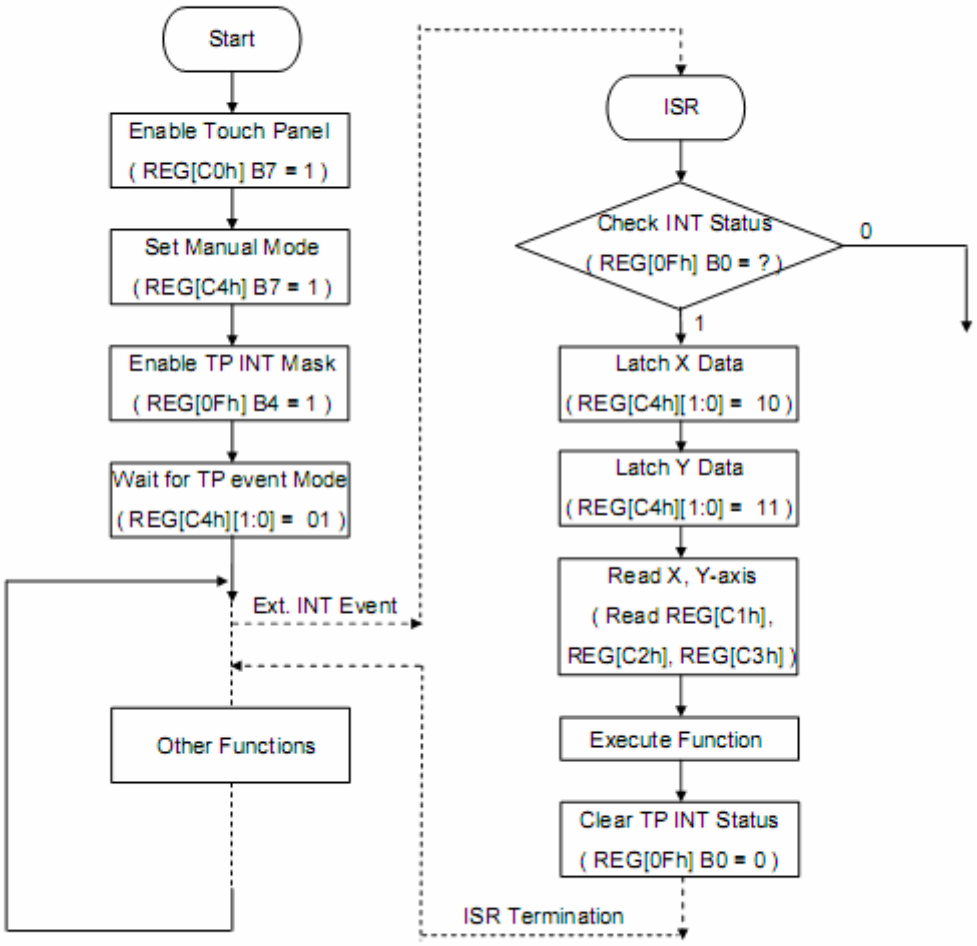


图 6-20：触控屏幕之手动模式流程图

轮询模式

在轮询模式（Polling Mode）模式下，使用者需要去决定触控事件之后「消除机械弹跳」（de-bounce）的时间，以及栓锁（latch）之后的取样时间，使用者运用此一模式在实际的应用上将有更多的弹性。此一模式之操作步骤如下：

1. 致能触控屏幕功能。
2. 切换触控屏幕的操作模式为「手动模式」。
3. 切换触控屏幕的相位为「等待触控事件发生」。
4. 从状态缓存器读取触控事件状态值，检查是否已发生触控事件。
5. 当触控事件发生时，且确认其为「有效」的事件后，即切换触控屏幕的相位为「栓锁 X data」（亦即设定缓存器 TPCR2[1:0]为 10b），并等待足够长的时间，使 X data 能稳定地储存在缓存器 TPXR 和 TPZR。

6. 切换触控屏幕的相位为「栓锁 Y data」（亦即设定缓存器 TPCR2[1:0] 为 11b），并等待足够长的时间，使 Y data 能稳定地储存在缓存器 TPYR 和 TPZR。

7. 从 TPXR、TPYR 和 TPZR 读回 XY 坐标值，并清除中断的状态值。

关于此一模式的缓存器设定，表列说明如下：

表 6-13

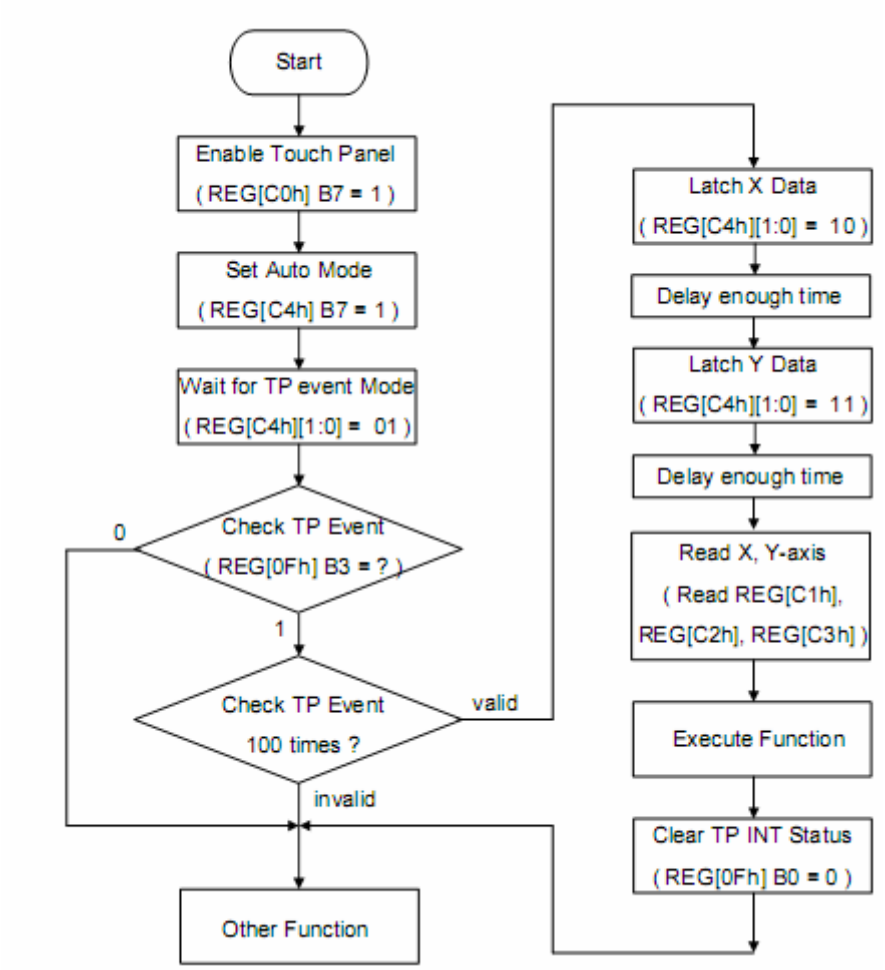
Reg.	Bit_Num	说　明	缓存器编号
TPCR1	Bit 7	触控屏幕功能的致能位。	REG[C0h]
TPCR2	Bit 7	用来选择「手动模式」或「自动模式」。	REG[C4h]
	Bit [1:0]	触控屏幕手动模式之选择位。	
INTR	Bit 3	触控事件之侦测位（仅用于触控屏幕下之手动模式）。	REG[0Fh]
	Bit 0	触控事件之状态位。	
TPXR	Bit [7:0]	触控屏幕 X 轴数据高字节 Bit[9:2]。	REG[C1h]
TPYR	Bit [7:0]	触控屏幕 Y 轴数据高字节 Bit[9:2]。	REG[C2h]
TPZR	Bit [3:2]	触控屏幕 Y 轴数据低二位 Bit[1:0]。	REG[C3h]
	Bit [1:0]	触控屏幕 X 轴数据低二位 Bit[1:0]。	

为了侦测触控事件之发生，程序设计师可以去检查缓存器 INTR 的位 3 或位 0，其中之差异，说明如下：

1. 缓存器 INTR 的位 3「实时反应」了目前的触控状态。当触控事件发生时，此位是 1；反之，当无触控事件发生时，此位是 0，而且此位是只读（read only）的，通常使用于「轮询模式」。
2. 缓存器 INTR 的位 0 则是记录了触控的状态。当触控事件发生时，此位是 1，但当无触控事件发生时，它却不会被自动清除为 0，必须由程序设计师来将它清除。此位通常使用于「外部中断模式」。

值得注意的是，缓存器 INTR 的位 3 是 ADC 电路的直接输出，一旦有屏幕被碰触，位 3 的状态将实时被反应出来。当此一碰触状态未达稳定时，需要「消除机械弹跳」（de-bounced）来确保此一碰触为「有效的触控事件」。因此，此一位只有在手动模式才是有作用的。当 RA8806 设定为「自动模式」时，触控事件将自动被侦测，并由系统来检查是否为有效事件，只有是有效的触控事件，中断才会产生。

应用轮询模式的流程图与范例程序如下所示。


Start
Enable Touch Panel
(REG[C0h] B7 = 1)
Set Auto Mode
(REG[C4h] B7 = 1)
Wait for TP event Mode
(REG[C4h][1:0] = 01)
Check TP Event
(REG[0Fh] B3 = ?)
0
1
Check TP Event
100 times ?
valid
invalid
Other Function
Latch X Data
(REG[C4h][1:0] = 10)
Delay enough time
Latch Y Data
(REG[C4h][1:0] = 11)
Delay enough time
Read X, Y-axis
(Read REG[C1h],
REG[C2h], REG[C3h])
Execute Function
Clear TP INT Status
(REG[0Fh] B0 = 0)

（7） 睡眠模式

RA8806 提供了两种操作模式：正常模式（Normal Mode）和睡眠模式（Sleep Mode）。请参考第 5 章 "缓存器描述" 中的对缓存器 WLCR 的说明。进入睡眠模式后，RA8806 会关掉系统时序以最大限度地减少功耗，此时除了状态缓存器可以读取其数据外，其它的缓存器都不允许，同时除了 REG[00H] 的 Bit-7 外其它的缓存器都不允许对其写入数据。而进入睡眠模式后，RA8806 可以通过以下三种方法来唤醒：

1. 写入 REG[00h] Bit-7 = 0，此时会回到正常模式。
2. Touch Panel 发生被触摸事件。
3. Key Scan 被按下时。

而为了让 RA8806 退去睡眠模式返回到正常模式后避免误动作的发生，必须在收到唤醒指令后必须延时足够的时间（大概 1,000 个系统时序周期）才可对它进行其它操作。

另外，在 MPU 发指令让 RA8806 进入睡眠模式到离开睡眠模式的时间内，RA8806 是不能接受任何指令，所以要向其发指令时要避免这种情况的发生。例如，RA8806 收到进入睡眠模式的指令进入睡眠，当 MPU 收到一个外部中断，进入中断服务程序（ISR），而此时正处于睡眠模式下的 RA8806 就不能正确接收和识别 MPU 发来的指令。在硬件设计方面，这种情况是不能避免的。因为这外部中断是只针对 MPU 的，而并没有对 RA8806 有任何的操作，所以这情况出现的可能性是不可忽略的。

所以，我们建议在 MPU 程序设计时，当给 RA8806 发进入睡眠模式指令之前先把 MPU 的中断响应关闭，直到退去睡眠模式返回到正常模式后再打开 MPU 的中断响应。另外，因为在睡眠模式下可以正常读取其状态缓存器，所以也可以在中断服务程序（ISR）中判断 RA8806 的状态。如果检测到 RA8806 是处于睡眠状态，MPU 可以退去中断服务程序（ISR）不作任何操作，这可避免 RA8806 发生误动作。

表 6-19

Reg.	Bit_Num	说　明	缓存器编号
WLCR	Bit 7	电源模式选择 - 正常模式 或 睡眠模式。	REG[00h]

（8） 中断与忙碌

RA8806 提供有一中断信号输出脚（INT）给 MPU 去响应 RA8806 的中断事件，也提供了一忙碌（Busy）信号输出脚给 MPU 去判断 RA8806 是否处于忙碌状态。这两个信号都可以通过相关缓存器的设置来改变其是高电位触发还是低电位触发。

RA8806 的中断信号会在以下事件发生时产生：

- Touch Panel 发生被触摸事件时
- Key-Scan 被按下时
- 睡眠模式下被唤醒时

这些中断事件的屏蔽（Disable）或致能（Enable）可以通过对缓存器 REG[0Fh] 的设置来控制。另外，RA8806 还提供了软件中断功能，当用户的系统不能支持硬件中断信号时，可以通过询问的方式进行软件中断。要进行硬件中断时，用户必须要把中断屏蔽位（Interrupt Mask）设为 1（请参考缓存器"INTR" 的说明），其进行步骤如下：

- RA8806 发出中断信号给 MPU。
- MPU 收到中断信号完成电路转换后，其程序计数器（PC）会跳到中断服务程序的入口。
- 此时 RA8806 的中断标志位被置 "1"（REG[0Fh] Bit[2:0]）。例如，当 Key-Scan 中断产生，其 Key-Scan 中断标志位就会被置 1。

使用软件中断方式时，用户不须要任何外部设置，只要通过读取缓存器 INTR 的相关状态位就可以检测中断事件是否发生。顺便提一下，中断屏蔽（interrupt mask）设置只能屏蔽硬件中断的产生，但不能屏蔽缓存器 INTR 的相关状态。

RA8806 也提供一忙碌（BUSY）信号，当忙碌标志位为 "1" 时就意味着 RA8806 正处于忙碌状态，而不能把数据写入显示内存（DDRAM）里。而其忙碌情况可分两种，一种是扫描忙碌（Scan Busy），另一种是内存写入忙碌（Memory Write Busy），具体说明如下：

扫描忙碌（Scan Busy）：

在LCD显示时，当RA8806 的扫描电路对DDRAM进行读取操作时，同时又有另外一数据被写入到DDRAM，就会造成扫描电路读取的数据丢失。所以把当扫描电路在扫描时所引起的忙碌状态叫扫描忙碌（Scan Busy）。图 6-34 是数据在扫描电路和MPU存取周期的传输情况，图 6-35 和图 6-15 一样都是反映RA8806 扫描时的相关波形。这波形反映了RA8806 对每根COM线的扫描情况。每根COM线的扫描时间由空闲时间（Idle）和扫描时间（Scan time）两者组成，其中空闲时间（Idle）可以通过缓存器ITCR来设置，而扫描期间就是扫描忙碌。所以在扫描忙碌时写入数据就会造成数据丢失，但这样也并不会造成严重的错误，只会有显示残缺的现象。如果这种情况不是太频繁发生的话，对显示来说也不会有太大的影响。

（9）

1，字符

RA8806 支持两种自MPU写入内存的模式，字符模式和图形模式，当设定图形模式时，数据是以点阵的方式直接写入内存，而在字符模式下，写入的数据是以字码的形式被写入RA8806，而写入的字码会再到CGROM中读出相对的字型码而后写入内存。RA8806 内建的CGROM提供两种不同大小的字符：1）是半型字（8x16 点），2）是全型字（16x16 点），图 6-40 为字型的范例。

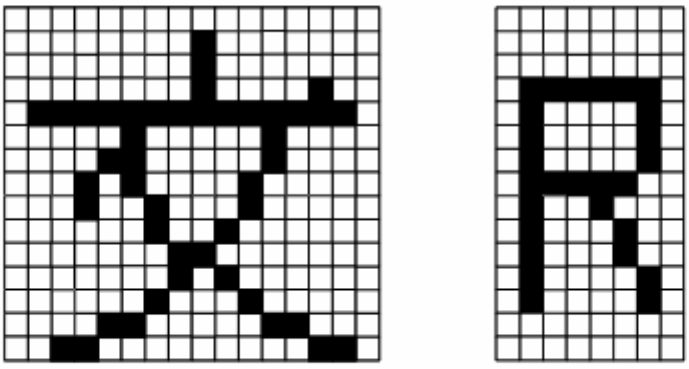


图 6-40：全型字与半型字

2, 图形显示

RA8806 图形模式是使用点阵方式来填数据到显示内存，图 6-41 为图形显示的例子。

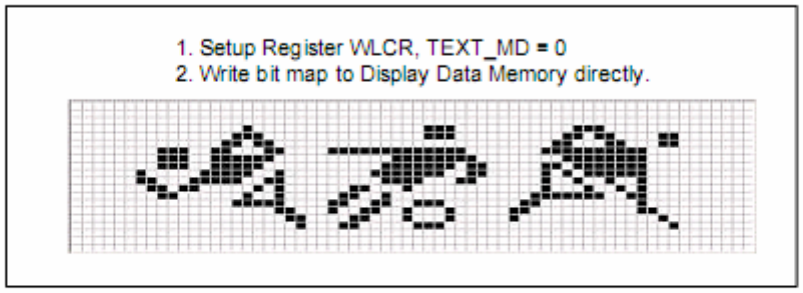


图 6-41：图形模式

RA8806 支持最大 320X240 的分辨率，因此它需要 9.6Kbyte（320X240/8 = 9600）的显示内存来储存每个画素的资料，图 6-42 显示了LCD 面板与显示内存间的对照方式

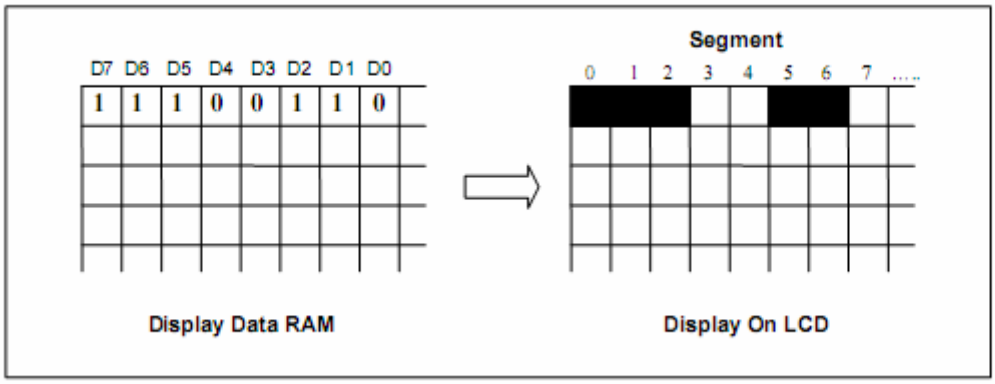


图 6-42 : LCD 面板与显示内存间的对照

RA8806 提供能将一笔数据填满整个显示内存的自动写入功能。首先，使用者把数据写入缓存器 PNTR，接着将自动写入功能启动（缓存器 FNCR 位-3）。RA8806 将把这笔资料在很短的时间内填满显示内存。这个功能通常被使用于清除屏幕或是想要填满固定的数据或背景在屏幕上。

表 6-22

Reg.	Bit_Num	说　明	缓存器编号
WLCR	Bit 3	文字模式选择。	REG[00h]

3， 半型字

RA8806 内建四个半型字的区块，每个区块包含了 00h ~ FFh的 256 个字型，而且能用缓存器 [F0h] bit[1:0] 来选择。依照着缓存器[F0h] bit[1:0]的设定，相对应的区块将被选择。关于字型码的对应位置，请参照附录C。RA8806 也支持可以提供大部份Latin 字型码设定的ISO8859 字型码（ISO - International Organization for Standardization）。在此模式下，RA8806 内部的字型码对照表可以对应到ISO8859-1 ~ ISO8859-4 的字型码。

表 6-23

Reg.	Bit_Num	说　明	缓存器编号
FNCR	Bit 7	ISO8859 模式。	REG[F0h]
	Bit 2	ASCII 模式致能。	
	Bit [1:0]	ASCII 区块选择。	

4， 全字型

RA8806 有二种形式的CGROM，例：RA8806-S 和 RA8806-T。RA8806-S包含了标准GB字型码的简体字型。RA8806-T则包含了标准BIG5 字型码的繁体字型。比较详细的叙述，请参照附录D和附录E。

表 6-24

Reg.	Bit_Num	说　明	缓存器编号
WLCR	Bit 3	文字模式选择。	REG[00h]

5， 粗体和反白

RA8806 支持粗体和反白字型，使用者只需在写入相对应的字型码给 RA8806 之前，先设定好对应的缓存器里的相关位。

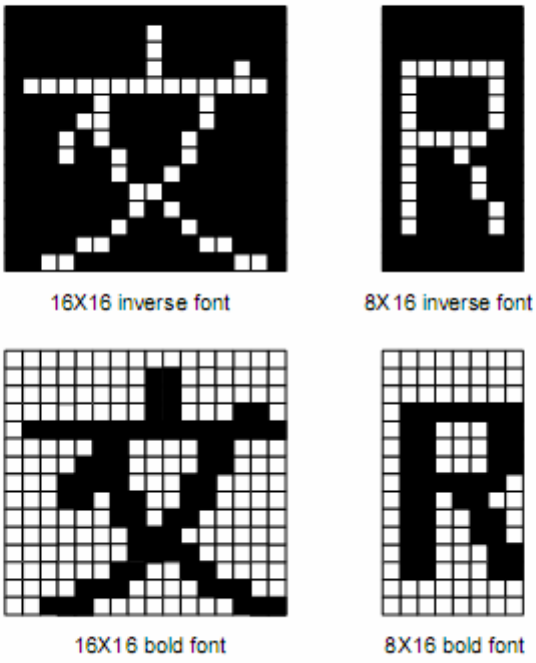


图 6-43：粗体和反白字型

表 6-25

Reg.	Bit_Num	说　明	缓存器编号
WLCR	Bit 3	文字模式选择。	REG[00h]
	Bit 2	显示开启/关闭选择设定。	
	Bit 1	闪烁模式选择。	
	Bit 0	反白模式选择。	
WCCR	Bit 4	粗体字（只支持文字模式）。	REG[10h]

6， 双图层显示

RA8806 内建为了双图层显示的二个显示内存，缓存器 MAMR 被使用来设定显示出来的效果和显示内存 1（DDRAM1）和显示内存 2（DDRAM2）间的关系，显示出来的效果如下列六种组合。

- 显示 DDRAM1 的数据
- 显示 DDRAM2 的数据
- 显示 "DDRAM1 OR DDRAM2" 的资料
- 显示 "DDRAM1 XOR DDRAM2" 的资料
- 显示 "DDRAM1 NOR DDRAM2" 的资料
- 显示 "DDRAM1 AND DDRAM2" 的资料

请参照图 6-44 和缓存器MAMR Bit[6:4] 和 Bit[3:2] 的叙述。

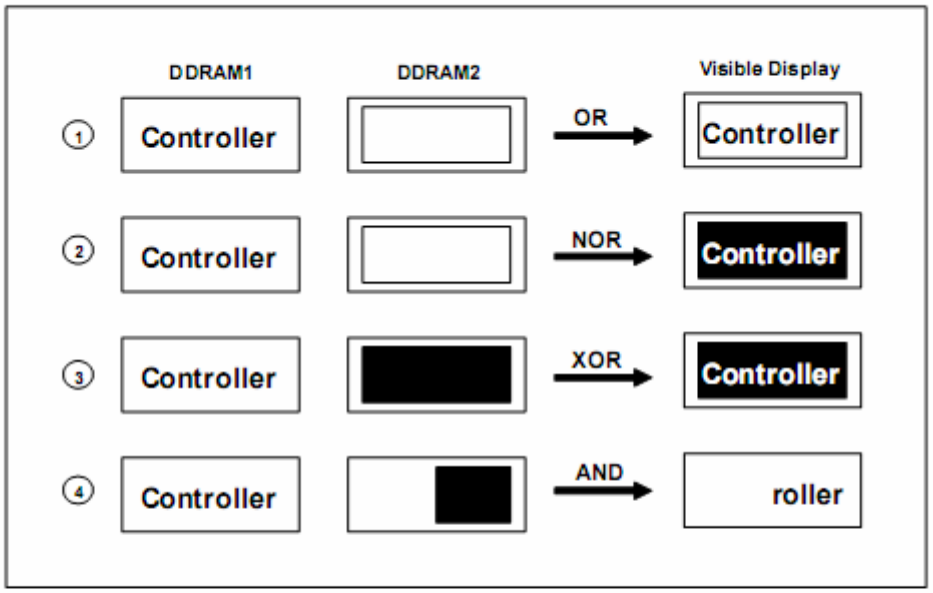


图 6-44 : 双图层显示

表 6-26

Reg.	Bit_Num	说　明	缓存器编号
MAMR	Bit [6:4]	显示图层选择。	REG[12h]
	Bit [3:2]	双图层模式选择。	

7， 行间距

RA8806 提供调整行与行间距的功能，尤其在显示中文时，在行与行之间加上一些空白看起来会更好，此间距的范围为 1 ~ 16 pixel。而只要此间距被设定好，光标将会自动移到每一行的适当位置。

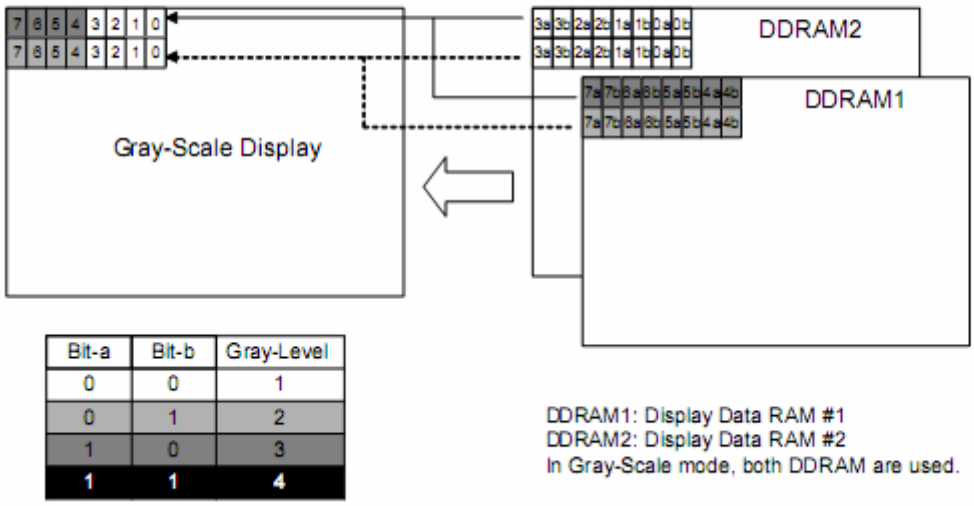
设定缓存器 CHWI 的低字节（Bit[3:0]），使用者可以调整行与行的间距。需提醒的是，当 RA8806 操作在 90 度模式时，不论字型的大小为何，行与行的间距只可被设定为 0 pixel 或 8 pixel。此二种间距的选择依据缓存器 CHWI Bit-3。

表 6-27

Reg.	Bit_Num	说　明	缓存器编号
CHWI	Bit [7:4]	光标高度设定。	REG[11h]
	Bit [3:2]	行间距设定。	

8， 灰阶扫描显示

RA8806 是依据FRC方法所产生的 4 灰阶扫描显示。在灰阶阶层显示模式下，RA8806 结合了显示内存 1 和显示内存 2 的数据来组成灰阶图案。每个灰阶位的显示包含了二个显示内存位的数据。数据[00b]代表空的位显示而数据[11b]代表全黑的位显示。[01b]和[10b]将为一共享时间方法（time sharing）而达成的 1/3 和 2/3 亮度。请参考图 6-45。



Bit-a	Bit-b	Gray-Level
0	0	1
0	1	2
1	0	3
1	1	4

DDRAM1: Display Data RAM #1
DDRAM2: Display Data RAM #2
In Gray-Scale mode, both DDRAM are used.

图 6-45：4 灰阶显示的对映方式

依照着数据写入的顺序，RA8806 支持了二种灰阶扫描显示的数据输入方法。使用者可以像一般图形显示方法一样来写入数据。

1. 先水平移动然后垂直移动。
2. 先垂直移动然后水平移动。

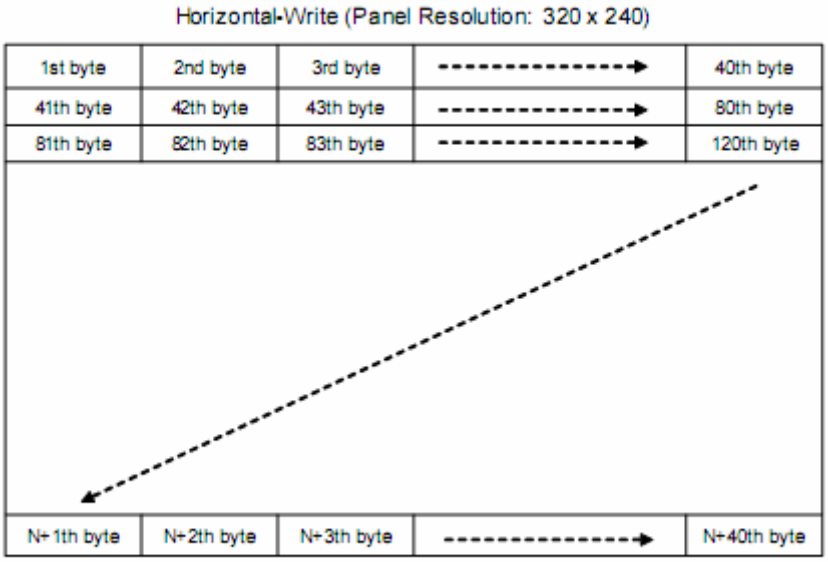


图 6-46：水平数据写入的顺序

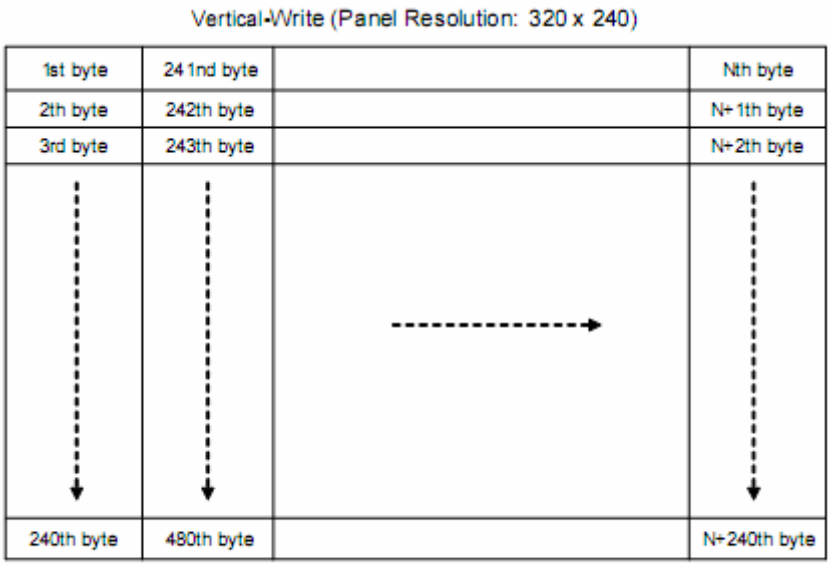


图 6-47：垂直数据写入的顺序

关于缓存器的设定，请参照下面的缓存器。

表 6-28

Reg.	Bit_Num	说　明	缓存器编号
MAMR	Bit 7	光标自动移动方向。	REG[12h]
	Bit [6:4]	灰阶模式选择。	

下面为一 4 灰阶图案显示的例子，而程序撰写的例子为在此图之下：

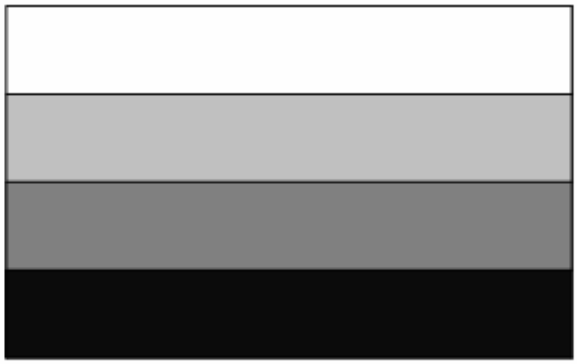


图 6-48：4 灰阶图案显示的例子

范例程序:

```
LCD_Graphic();                  // Set graphic mode
Gray_Mode();                    // Set REG[12h] Bit[6:4] to 000b, as gray level display
Scan_Diret_H_V();               // Set REG[12h] Bit-7 to 0, Cursor moves from left to
right
LCD_GotoXY( 0 , 0 );
LCD_CmdWrite( 0xB0 );           // Write memory command
for ( i = 0 ; I < 4800 ; i++ )  // Write data
{
    LCD_DataWrite( 0x00 ) ;     // Gray Level 1 ( 00 )
}
for ( i = 0 ; I < 4800 ; i++ )
{
    LCD_DataWrite( 0x55 ) ;     // Gray Level 2 ( 01 )
}
for ( i = 0 ; I < 4800 ; i++ )
{
    LCD_DataWrite( 0xAA ) ;   // Gray Level 3 ( 10 )
}
for ( i = 0 ; I < 4800 ; i++ )
{
    LCD_DataWrite( 0xFF ) ;     // Gray Level 4 ( 11 )
}
```

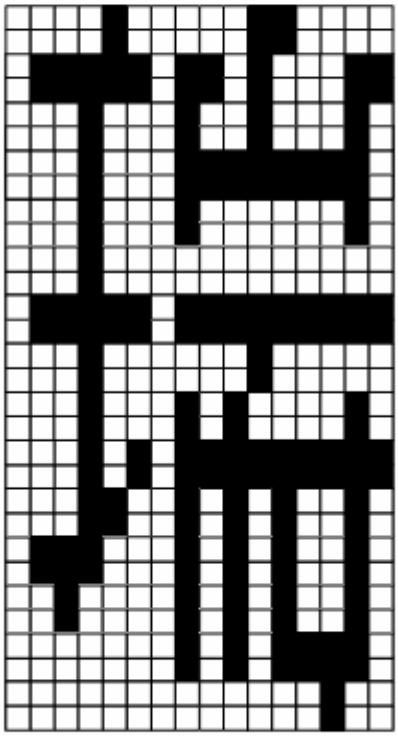
9， 字号调整和字写入的时间

RA8806 提供文字大小调整功能。存在 RA8806 字型 ROM 里有二种形式的字：半型字（8x16 pixels）和全型字（16x16 pixels）。文字大小调整提供字型在水平和垂直方向 1 到 4 倍的放大功能。程序设计者可以借着设定缓存器 FTHT[7:4]来调整字型的大小。

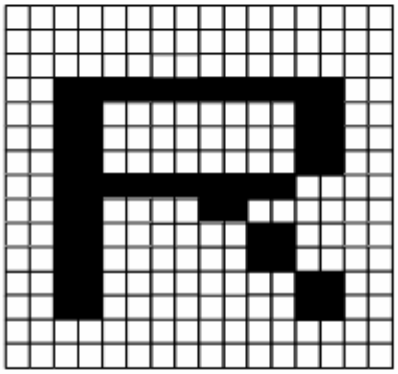
表 6-29

Reg.	Bit_Num	说　明	缓存器编号
FVHT	Bit [7:6]	文字水平宽度调整。	REG[F1h]
	Bit [5:4]	文字垂直高度调整。	

图 6-49 为字型放大的 2 个例子：



1x2 enlarged 16x16 font



2x1 enlarged 8x16 font

图 6-49 : Font Size Adjustment

此外，文字大小调整功能只有在文字显示模式下有效，但在下面的二种显示模式下也有效：

- 显示使用者自行创造出的字型。
- 90 度文字和旋转显示模式。

在写下完整的字型码之后（One byte：半型字、Two bytes：全型字），RA8806 将需要一段足够的时间把数据写进显示内存（请参照第 6-8 节 "中断与忙碌" 及第 6-8-2 节内 "Memory Write Busy" 的叙述）。写进的时间依字型放大的倍率而有所不同。其计算公式如下：

T_{fw} = (16*tc + 32*tc x (HW x VH)) x (1 + 10%)

T_{fw} : 文字写进时间（Font Write time）
tc ：系统频率（System clock period）
HW ：水平放大倍率（Horizontal enlargement multiplier）
VH : 垂直放大倍率（Vertical enlargement multiplier）

此处有 10% 是给弹性错误考虑。请参照以下的例子：（系统频率约为 4MHz，而 tc 为 250ns。）

表 6-30 : 文字写进显示内存

字 形 放 大 倍 数	写进显示内存时间(μS)	建议等待时间(μS)
1 x 1	12	13.2
2 x 2	36	39.6
3 x 3	76	83.6
4 x 4	132	145.2

除了计算写进的时间和延迟外，程序设计者也可以检查忙碌（Busy）的状态来知道文字是否已经被写进显示内存。请参照状态缓存器（Status Register）的忙碌（Busy）叙述。

10， 文字垂直显示

RA8806 另有一个非常独特的 "文字垂直显示" 功能。此功能使得一般显示变成垂直显示。此意思代表原本 320x240 的模块可以被使用成一个 240x320 的模块以垂直方式来显示内建的字型或数据，而不需要改变扫描的方向或是LCD驱动的线路编排。此功能为在缓存器WCCR Bit-3 设定的一个选择。当WCCR Bit-3 =1（文字旋转模式），文字将被写进为垂直显示。图 6-50 为文字垂直显示的一个例子：

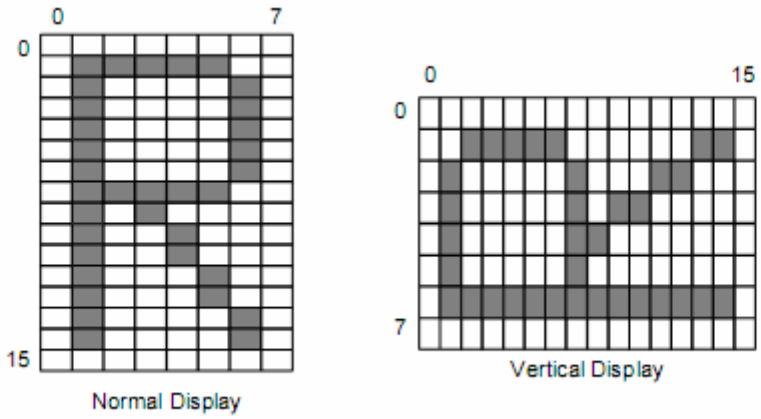


图 6-50: 文字垂直显示

在垂直显示模式下，光标移动方向是由上到下。且光标的显示样式和一般显示模式下不同。（请参照第 6-13-4 节 "游标的宽度和高度" 中完整的叙述。）

因为有此卓越的旋转功能，假如使用者选择一个 240x160 的模块，然后他将非常简单地使用原本的模块来当成一个 160x240 的模块。下方图 6-51 为一 320x240 模块旋转到 240x320 的应用。在一般模式下，假设写入的文字为"RAIO"，则屏幕显示如图 6-51（1）。而当先设定T90DEG（REG[10h] 的Bit-3）= 1 和CDIR（REG[01h] 的Bit-0）= 1 时，写入文字 "RAIO" 将屏幕显示将如图 6-51（2）。

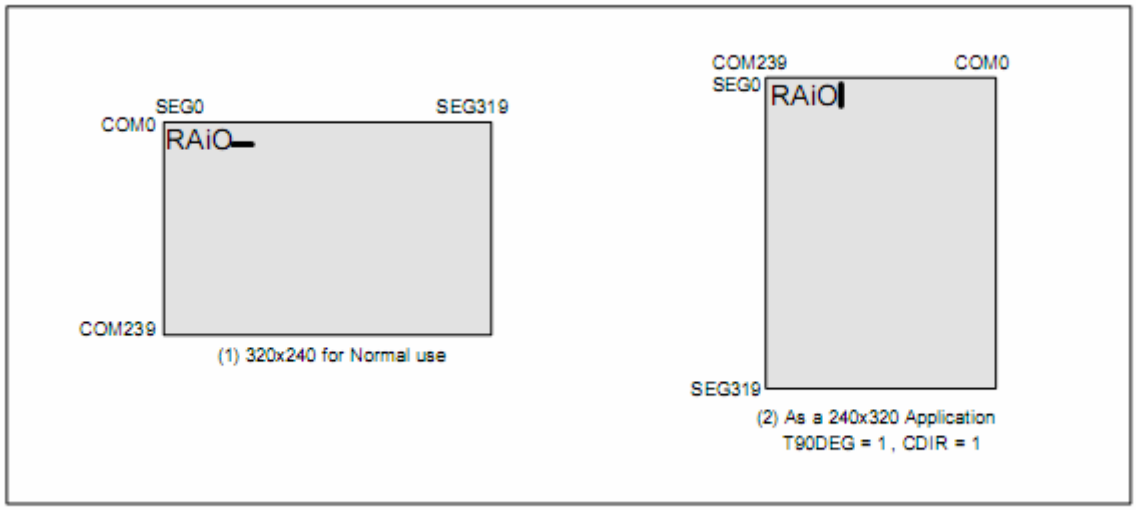


图 6-51 : 文字水平与垂直显示

有关文字旋转功能的相关的缓存器表如下：

表 6-31

Reg.	Bit_Num	说　明	缓存器编号
WCCR	Bit 3	文字旋转模式致能。	REG[10h]
MISC	Bit 0	COM 方向选择。	REG[01h]

除了光标显示功能外、其它的功能在文字旋转模式下运作皆和一般模式下的行为一样，包括：

1）文字放大功能

2）文字对齐功能

3）行间距调整功能

4）旋转功能

11， 使用者自创字型

RA8806 内建一个 512Byte CGRAM 提供给使用者创造新的字型。使用者可以创造字型、特别的符号或数据在此内存里。假设使用者只使用一个显示内存（DDRAM），另一个显示内存也可被当成一个 CGRAM 来使用。

因此，RA8806 提供了三个区域来给使用者创造新的字型（标帜或数据）如下：

1. 512 Bytes CGRAM。
2. 9.6K Bytes DDRAM1（当使用者只让 DDRAM2 的数据显示在屏幕上）。
3. 9.6K Bytes DDRAM2（当使用者只让 DDRAM1 的数据显示在屏幕上）。

表 6-32：字型码的对映范围

Region	Size	Code and Range	Capacity
CGRAM	512 bytes	半型字：8F00h ~ 8F1Fh 全型字：9F00h ~ 9F0Fh	32 Half-Size chars 16 Full-Size chars
DDRAM1	9.6 Kbytes	半型字：8000h ~ 8E27h 全型字：9000h ~ 9E13h	600 Half-Size chars 300 Full-Size chars
DDRAM2	9.6 Kbytes	半型字：8000h ~ 8E27h 全型字：9000h ~ 9E13h	600 Half-Size chars 300 Full-Size chars

6-11-1 在CGRAM创造字型

此CGRAM为一 512Byte RAM，所以它可以创造 32 个半型字或 16 个全型字。表 6-33 为一字型码对应到CGRAM的表。注意在RA8806 里字型码是限定的。且FONT#被定义在缓存器CURY Bit[3:0]里。

表 6-33 : CGRAM 字型码对映表

Font# / Code	0		1		2		3		4		5		6		7	
Half-Size	8F00	8F01	8F02	8F03	8F04	8F05	8F06	8F07	8F08	8F09	8F0A	8F0B	8F0C	8F0D	8F0E	8F0F
Full-Size	9F00		9F01		9F02		9F03		9F04		9F05		9F06		9F07	

Font# / Code	8		9		10		11		12		13		14		15	
Half-Size	8F10	8F11	8F12	8F13	8F14	8F15	8F16	8F17	8F18	8F19	8F1A	8F1B	8F1C	8F1D	8F1E	8F1F
Full-Size	9F08		9F09		9F0A		9F0B		9F0C		9F0D		9F0E		9F0F	

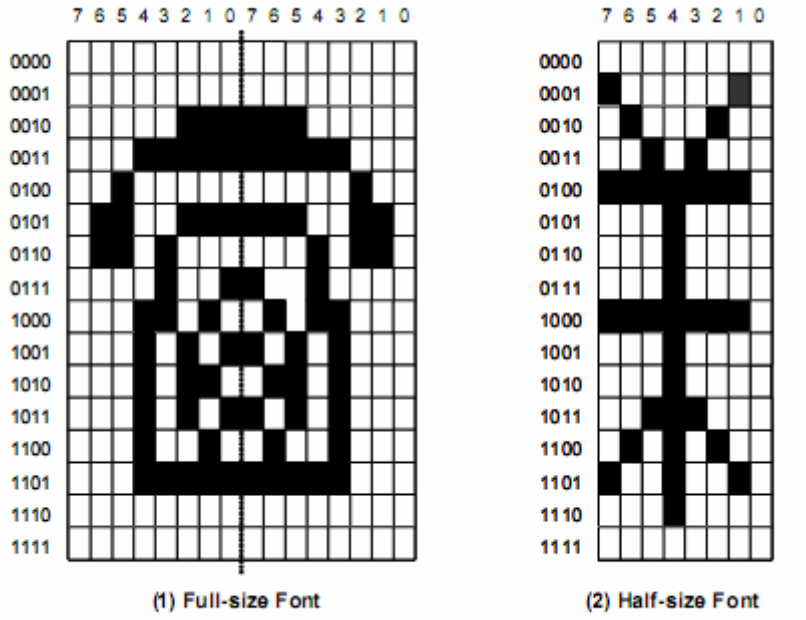


图 6-52 : 创造全型字与半型字的范例

在 CGRAM 里使用者创造字型缓存器的相关设定，请参照下面的表格：

表 6-34

Reg.	Bit_Num	说　明	缓存器编号
MAMR	Bit 7	光标自动增加致能。	REG[12h]
	Bit [6:4]	显示组合模式。	
	Bit [1:0]	写入周期内存选择。	
CURX	Bit [4:0]	Bit[4:0] 为给写入创造字型的位对应数据之地址。当要创造一个全角字，通常设定为 0。而在创造半角字时，奇数半角字码通常设定为 0，而偶数半角字则设定为 10 h。	REG[60h]
CURY	Bit [3:0]	指示哪一个字型码数据被创造。	REG[70h]

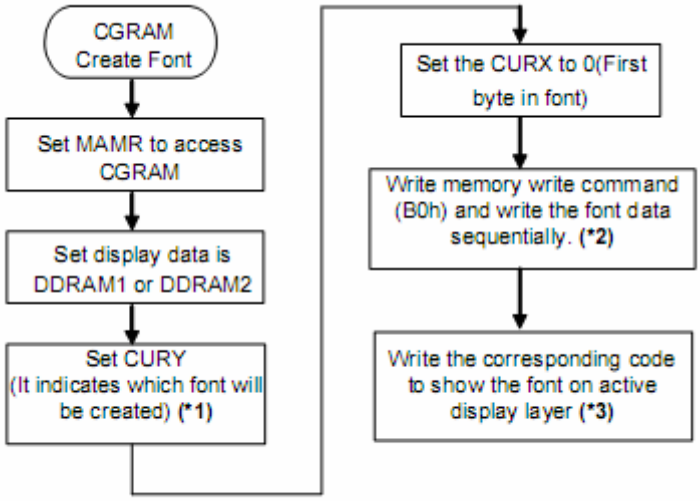


图 6-53 : 在 CGRAM 创造字型的流程图

注：

1. 此 CGRAM 大小为 512 bytes 且包括了 16 个 16x16 个全型或是 32 个 8x16 个半型使用者创造字。每一个全型字包含 32bytes。
2. 为了创造全型字，在设定好CURY（0~15）和CURX设定为 0 后，一个使用者创造字型位对应可被 32 个内存写入周期所填满，而后CURY将增加 1。请参照下方的程序例子（1）。结果显示在图 6-52（1）。
3. 为了创造半型字，在设定好CURY（0~15）和CURY设定为 0（odd font）或是 10h（even font）后，一个使用者创造字型位对应可被 16 个内存写入周期所填满。请参照下方的程序例子（2），结果显示在图 6-52（2）。
4. CGRAM字型码半型字为 8F00~8F1F，而全型字为 9F00~9F0Fh。请参考表 6-32。

范例程序(1):

```
// Create a full-size font in CGRAM and show on the screen.
// font_data[] = {00, 00, 07, 1F, 20, 67, 68, 04, 1A, 15, 16, 15, 12, 1F, 00, 00,
// 00, 00, E0, F8, 04, E6, 16, 90, 58, A8, 68, A8, 48, F8, 00, 00}
Access_CGRAM();                     // MAMR[1:0] = 00b
Only_Show_DDRAM1();                 // MAMR[6:4] = 001b
LCD_CMDWrite( CURY );               // Create 6th full-size character
LCD_DataWrite( 0x05 );
LCD_CMDWrite( CURX );               // Write font-data from 1st byte
LCD_DataWrite( 0x00 );
LCD_CMDWrite( MWrite );             // B0h, Memory write command
for( i = 0; i < 32;  i++ )          // 32 continuous write for font bit-map
  {  LCD_DataWrite( font_data[i] );
     Delay2us(50);
  }
LCD_GotoXY( 5 , 5 );                // Set Cursor position ( 5 , 5 )
LCD_CMDWrite( MWrite );             // Corresponding font code( 9F05h )
LCD_DataWrite( 0x9F );
LCD_DataWrite( 0x05 );              // Show as (1) of 图 6-52
```

范例程序(2):

```
// Create a half-size font in CGRAM and show on the screen.
// font_data[] = {00, 82, 44, 28, FE, 10, 10, 10, FE, 10, 10, 38, 54, 92, 10, 00}
Access_CGRAM();                     // MAMR[1:0] = 00b
Only_Show_DDRAM1();                 // MAMR[6:4] = 001b
LCD_CMDWrite( CURY );               // Create 6th half-size char
LCD_DataWrite( 0x02 );
LCD_CMDWrite( CURX );               // Write font-data from 1st byte
LCD_DataWrite( 0x10 );
LCD_CMDWrite( MWrite );             // B0h, Memory write command
for( i = 0; i < 16;  i++ )          // 16 continuous write for font bit-map
  {
     LCD_DataWrite( font_data[i] );
     Delay2us(50);
  }
LCD_GotoXY( 5 , 5 );                // Set Cursor position ( 5 , 5 )
LCD_CMDWrite( MWrite );             // Corresponding font code( 8F05h )
LCD_DataWrite( 0x8F );
LCD_DataWrite( 0x05 );              // Show as (2) of 图 6-52
```

6-11-2 在显示内存里创造字型

显示内存为 9.6KByte RAM，所以它可以创造 600 个半型字或 300 个全型字。表 6-36 为半型字在显示内存里的字型码对照表，

表 6-37 为全型字在显示内存里的字型码对照表，这些字型码在RA8806 也为限定的。此时缓存器CURX和CURY是用来指定位对应数据(bit-map data)来创造字型的地址。

在显示内存（DDRAM）里使用者创造字型缓存器的相关设定，请参照下面的表格：

表 6-35

Reg.	Bit_Num	说　明	缓存器编号
MAMR	Bit 7	光标自动增加致能。	REG[12h]
	Bit [6:4]	显示组合模式。	
	Bit [1:0]	写入周期内存选择。	
CURX	Bit [5:0]	此二缓存器用来指示哪一个字型码将被创造。	REG[60h]
CURY	Bit [7:0]		REG[70h]

表 6-36 : DDRAM1 与 DDRAM2 造半型字时的字型码对映表

	CURX / CURY	0	1	2	. . .	26	27
0	00	8000	8001	8002	. . .	8026	8027
1	10	8100	8101	8102	. . .	8126	8127
2	20	8200	8201	8202	. . .	8226	8227
3	30	8300	8301	8302	. . .	8326	8327
4	40	8400	8401	8402	. . .	8426	8427
5	50	8500	8501	8502	. . .	8526	8527
6	60	8600	8601	8602	. . .	8626	8627
7	70	8700	8701	8702	. . .	8726	8727
8	80	8800	8801	8802	. . .	8826	8827
9	90	8900	8901	8902	. . .	8926	8927
10	A0	8A00	8A01	8A02	. . .	8A26	8A27
11	B0	8B00	8B01	8B02	. . .	8B26	8B27
12	C0	8C00	8C01	8C02	. . .	8C26	8C27
13	D0	8D00	8D01	8D02	. . .	8D26	8D27
14	E0	8E00	8E01	8E02	. . .	8E26	8E27

表 6-37：DDRAM1 与 DDRAM2 造全型字时的字型码对映表

CURY \ CURX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20	21	22	23	24	25	26	27
0	00	9000	9001	9002	9003	9004	9005	9006	9007	9008	9009	900A	900B	900C	900D	900E	900F	9010	9011	9012	9013																		
1	10	9100	9101	9102	9103	9104	9105	9106	9107	9108	9109	910A	910B	910C	910D	910E	910F	9110	9111	9112	9113																		
2	20	9200	9201	9202	9203	9204	9205	9206	9207	9208	9209	920A	920B	920C	920D	920E	920F	9210	9211	9212	9213																		
3	30	9300	9301	9302	9303	9304	9305	9306	9307	9308	9309	930A	930B	930C	930D	930E	930F	9310	9311	9312	9313																		
4	40	9400	9401	9402	9403	9404	9405	9406	9407	9408	9409	940A	940B	940C	940D	940E	940F	9410	9411	9412	9413																		
5	50	9500	9501	9502	9503	9504	9505	9506	9507	9508	9509	950A	950B	950C	950D	950E	950F	9510	9511	9512	9513																		
6	60	9600	9601	9602	9603	9604	9605	9606	9607	9608	9609	960A	960B	960C	960D	960E	960F	9610	9611	9612	9613																		
7	70	9700	9701	9702	9703	9704	9705	9706	9707	9708	9709	970A	970B	970C	970D	970E	970F	9710	9711	9712	9713																		
8	80	9800	9801	9802	9803	9804	9805	9806	9807	9808	9809	980A	980B	980C	980D	980E	980F	9810	9811	9812	9813																		
9	90	9900	9901	9902	9903	9904	9905	9906	9907	9908	9909	990A	990B	990C	990D	990E	990F	9910	9911	9912	9913																		
10	A0	9A00	9A01	9A02	9A03	9A04	9A05	9A06	9A07	9A08	9A09	9A0A	9A0B	9A0C	9A0D	9A0E	9A0F	9A10	9A11	9A12	9A13																		
11	B0	9B00	9B01	9B02	9B03	9B04	9B05	9B06	9B07	9B08	9B09	9B0A	9B0B	9B0C	9B0D	9B0E	9B0F	9B10	9B11	9B12	9B13																		
12	C0	9C00	9C01	9C02	9C03	9C04	9C05	9C06	9C07	9C08	9C09	9C0A	9C0B	9C0C	9C0D	9C0E	9C0F	9C10	9C11	9C12	9C13																		
13	D0	9D00	9D01	9D02	9D03	9D04	9D05	9D06	9D07	9D08	9D09	9D0A	9D0B	9D0C	9D0D	9D0E	9D0F	9D10	9D11	9D12	9D13																		
14	E0	9E00	9E01	9E02	9E03	9E04	9E05	9E06	9E07	9E08	9E09	9E0A	9E0B	9E0C	9E0D	9E0E	9E0F	9E10	9E11	9E12	9E13																		

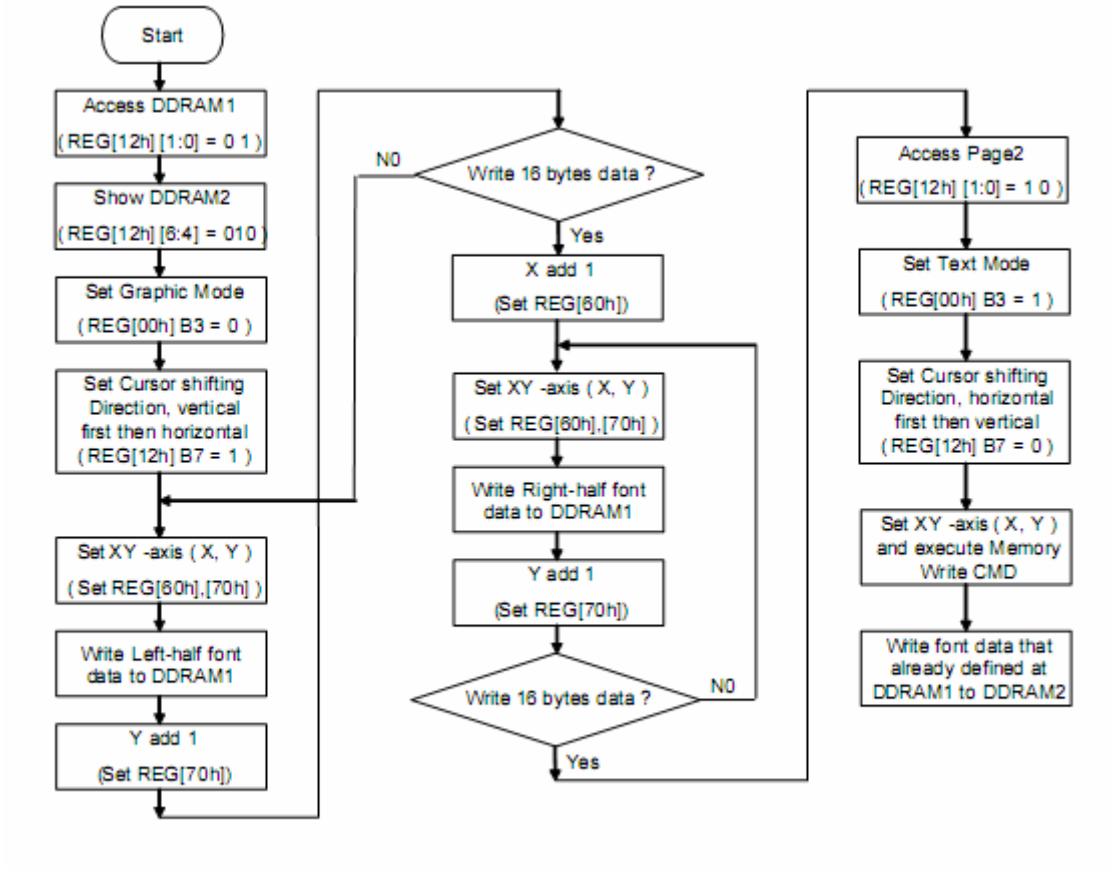


图 6-54：在 DDRAM1 创造字型的流程图

上方的流程为在DDRAM1 创造一个全型字和显示这一新的字型在DDRAM2 再屏幕上的一个例子。首先，使用者必须先设定CURX和CURY来定义哪一字型码将被创造，然后写 32Bytes位对应数据到DDRAM1。因为全型字的左边和右边各有 16Bytes数据，CURX需设定二次。请参照程序例子（3），此例我们假设想要创造如图 6-52（1）的全型字，同时对应到
表 6-37 中的码 "9205h"，其结果会在屏幕上显示如图 6-52（1）的电话字。

范例程序(3):

```
// Create a full-size font in DDRAM1 and show on the screen.
// font_data[] = {00, 00, 07, 1F, 20, 67, 68, 04, 1A, 15, 16, 15, 12, 1F, 00, 00,
// 00, 00, E0, F8, 04, E6, 16, 90, 58, A8, 68, A8, 48, F8, 00, 00}

Access_DDRAM1();                // MAMR[1:0] = 01
Only_Show_DDRAM2();             // MAMR[6:4] = 010

LCD_Graphic();                  // WLCR. Bit-3 = 0
Cursor_Shift_Direct_VH();       // MAMR. Bit-7 = 1
                                // Set the cursor moving in vertical
LCD_GotoXY(0x0A, 0x20);         // Write the left part of full-size font
LCD_CMDWrite( MWrite );         // Memory write command
for(i=0;i<16;i++)
{
    LCD_DataWrite(font_data_L[i]);
    Delay2us(50);
}

LCD_GotoXY(0x0B, 0x20);         // Write the right part of full-size font
LCD_CMDWrite( MWrite );         // Memory write command
for(i=16;i<32;i++)
{
    LCD_DataWrite(font_data_R[i]);
    Delay2us(50);
}

Access_DDRAM2();                // MAMR[1:0] = 10
LCD_Text();                     // WLCR.Bit-3 = 1
Cursor_Shift_Direct_HV();       // MAMR. Bit-7 = 0

LCD_GotoXY( 3 , 3 );            // set coordinate to ( 3 , 3 )
LCD_CMDWrite( MWrite );         // Memory write command
LCD_DataWrite(0x92);            // Write the code(9205h) of user-defined font
LCD_DataWrite(0x05);            // Show as (1) of 图 6-52
Delay2us(50);
```

6-11-3 创造符号

RA8806 提供文字创造功能让使用者可以设计和创造出新的字型，特别的符号（Symbol）和标帜（Pattern）且储存进它内建的CGRAM或DDRAM里。使用者自创字型定义为全型字（16x16）或半型字（8x16）的位对应格式，所以 24x16、40x16、16x32 或是更大的字型也可能依此格式被创造出。图 6-55 呈现一个 24x16 大小的使用者自创符号。

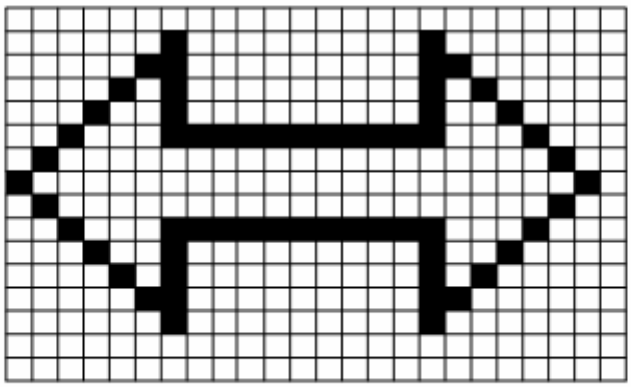


图 6-55：一个 24x16 大小的符号范例

为了创造出新的符号，除了写命令到 CGRAM 或 DDRAM 外，在试着写文字到显示之前，字型码的设定要注意。参考创造此 24x16 符号于 DDRAM 接下来的二个例子：

- 当数据写进DDRAM里的第一个地址为（0, 0）时，此符号的字型码为 0x8000、0x8001 和 0x8002。如下方图 6-56 所示。
- 数据写进DDRAM里的第一个地址为（2, 10h）时，此符号的字型码为 0x8105、0x8106 和 0x8107。请参照下方程式例子（4）。其结果显示于图 6-55。

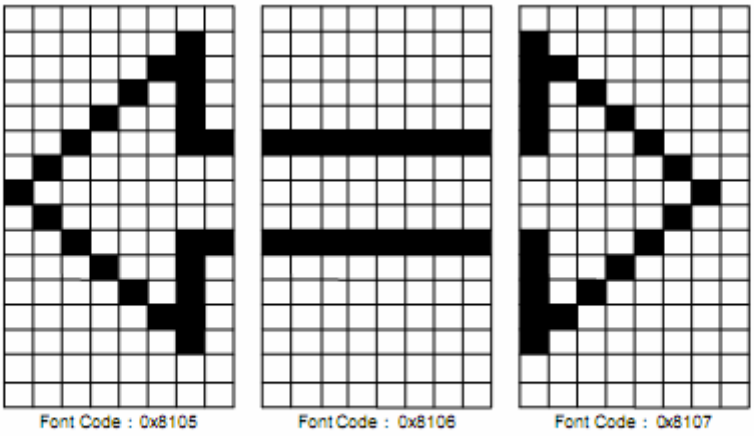


图 6-56：范例程序（4）使用的自创符号字型码

范例程序(4):

```
// Create a 24x16 symbol in DDRAM1 and show on the screen.
// font_data[] = {00, 02, 06, 0A, 12, 23, 40, 80, 40, 23, 12, 0A, 06, 02, 00, 00,
// 00, 00, 00, 00, 00, FF, 00, 00, 00, FF, 00, 00, 00, 00, 00, 00,
// 00, 80, C0, A0, 90, 88, 04, 02, 04, 88, 90, A0, C0, 80, 00, 00}

Access_DDRAM1();                // MAMR[1:0] = 01
Only_Show_DDRAM2();             // MAMR[6:4] = 010

LCD_Graphic();                  // WLCR. Bit-3 = 0
Cursor_Shift_Direct_VH();       // MAMR. Bit-7 = 1
                                // Set the cursor moving in vertical
LCD_GotoXY(0x02, 0x10);         // Write the left part of symbol
LCD_CMDWrite( MWrite );         // Memory write command
for(i=0;i<16;i++)
{
    LCD_DataWrite(font_data_L[i]);
    Delay2us(50);
}

LCD_GotoXY(0x03, 0x10);         // Write the middle part of symbol
LCD_CMDWrite( MWrite );         // Memory write command
for(i=16;i<32;i++)
{
    LCD_DataWrite(font_data_R[i]);
    Delay2us(50);
}

LCD_GotoXY(0x04, 0x10);         // Write the right part of symbol
LCD_CMDWrite( MWrite );         // Memory write command
for(i=32;i<48;i++)
{
    LCD_DataWrite(font_data_R[i]);
    Delay2us(50);
}


Access_DDRAM2();                // MAMR[1:0] = 10
LCD_Text();                     // WLCR.Bit-3 = 1
Cursor_Shift_Direct_HV();       // MAMR. Bit-7 = 0

LCD_GotoXY( 3 , 3 );            // set coordinate to ( 3 , 3 )
LCD_CMDWrite( MWrite );         // Memory write command
LCD_DataWrite(0x81);            // Write the code(8105h, 8102, 8103) of symbol
LCD_DataWrite(0x02);            // Show as 图 6-56
LCD_DataWrite(0x81);
LCD_DataWrite(0x03);
LCD_DataWrite(0x81);
LCD_DataWrite(0x04);
Delay2us(50);
```

除此之外，注意每个字型码被显示于 LDC 屏幕上的显示地址，尤其在 90 旋转换文字放大模式下。

12， 卷动功能

6-12-1 水平卷动

RA8806 提供一水平卷动功能，使用者可以透过 BGSG 和 EDSG 两个缓存器来指定卷动范围。一旦开启水平卷动功能，被指定的区域范围将直进行循环的单步水平卷动，其每次水平卷动的单位宽度为 8 个显示点数。

有关水平卷动的缓存器设定，请参考下列说明。

表 6-38

Reg.	Bit_Num	说　明	缓存器编号
ADSR	Bit 2	SCR_DIR（卷动方向）。	REG[03h]
	Bit 1	SCR_HV（垂直或水平卷动设定）。	
	Bit 0	SCR_EN（卷动致能）。	
BGSG	Bit [5:0]	设定卷动模式 Segment 的起始位置。	REG[61h]
EDSG	Bit [5:0]	设定卷动模式 Segment 的结束位置。	REG[62h]
BGCM	Bit [7:0]	设定卷动模式 Common 的起始位置。	REG[71h]
EDCM	Bit [7:0]	设定卷动模式 Common 的结束位置。	REG[72h]

(1) 流程图： 水平卷动功能的流程图说明

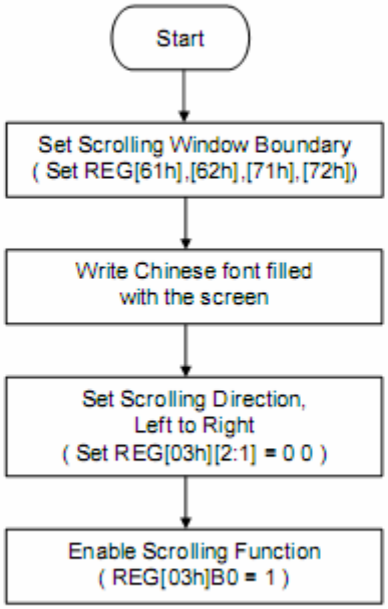


图 6-57：水平卷动流程图

(2) 范例程序:

```
LCD_Text();                    // Text Mode
//================================================
// Set Scrolling Window
//================================================
LCD_CmdWrite(0x61);        // SEG Start Position of Scrolling Mode
LCD_DataWrite(0x05);
LCD_CmdWrite(0x62);        // SEG End Position of Scrolling Mode
LCD_DataWrite(0x22);

LCD_CmdWrite(0x71);        // COM Start Position of Scrolling Mode
LCD_DataWrite(0x20);
LCD_CmdWrite(0x72);        // COM End Position of Scrolling Mode
LCD_DataWrite(0xd0);

for( i = 0 ; i < 300 ; i++ )        // Write the font to fill the screen
{
     LCD_DataWrite( RAiO [ i ] );
     Delay2us(50);
}

//================================================
// Set Scrolling Direction and Enable scroll function
//================================================
LCD_CmdWrite(0x03);
LCD_DataWrite(0x01);        // L → R
// LCD_DataWrite(0x05);     // R → L
```

6-12-2 垂直卷动

RA8806 亦提供垂直卷动功能，此卷动功能可以经由 ADSR Bit-2 来启动。一旦开启垂直卷动功能，整个显示屏将一直进行循环的单步垂直卷动，其每次垂直卷动的单位为 1 个显示点数。

(1) 流程图：垂直卷动功能的流程图说明

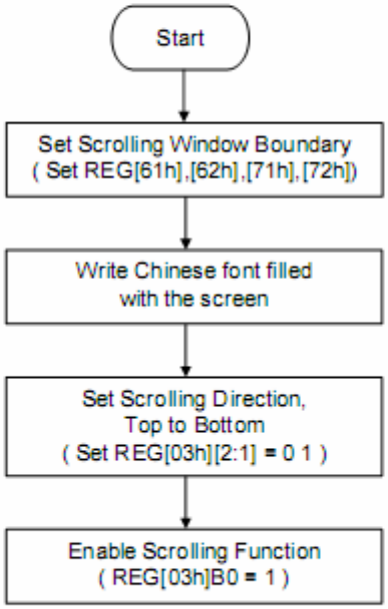


图 6-58：垂直卷动流程图

(2) 范例程序:

```
LCD_Text();                 // Text Mode

LCD_CmdWrite(0x61);         // SEG Start Position of Scrolling Mode
LCD_DataWrite(0x05);
LCD_CmdWrite(0x62);         // SEG End Position of Scrolling Mode
LCD_DataWrite(0x22);

LCD_CmdWrite(0x71);         // COM Start Position of Scrolling Mode
LCD_DataWrite(0x20);
LCD_CmdWrite(0x72);         // COM End Position of Scrolling Mode
LCD_DataWrite(0xd0);

for( i = 0 ; i < 300 ; i++ )        // Fill Chinese font on the screen
{
     LCD_DataWrite( RAiO [ i ] );
     Delay2us(50);
}

LCD_CmdWrite(0x03);         // Scrolling Enable
LCD_DataWrite(0x03);        // T → B
// LCD_DataWrite(0x07);     // B → T
```

13， 游标闪烁

使用者可以根据其使用需求，选定光标显示或不显示，亦可开启光标闪烁功能。光标的闪烁时间，可经由缓存器 "[80h] BTMR" 来调整。

- 闪烁时间 = BTMR[80h] Bit[7:0] x (1/Frame_Rate)

6-13-4 游标的宽度和高度

RA8806 的光标高度可按照使用需求，经由缓存器 CHWI Bit[7:4]来做调整，其高度调整范围为 1~16 pixels。

在一般的显示字型条件下，光标的宽度是固定在 1 个byte（8 pixels）。游标的高度则是从 1~16 pixels，其高度是依据缓存器CHWI Bit[7:4]来设定。在垂直显示模式下，光标的高度是被固定在 16 pixels，而其宽度是可以依照使用需求被设定为 1~8 个pixels，此时光标的宽度可以透过缓存器CHWI Bit[6:4]来做调整。详细说明请参考图 6-64 以及图 6-65 的说明。

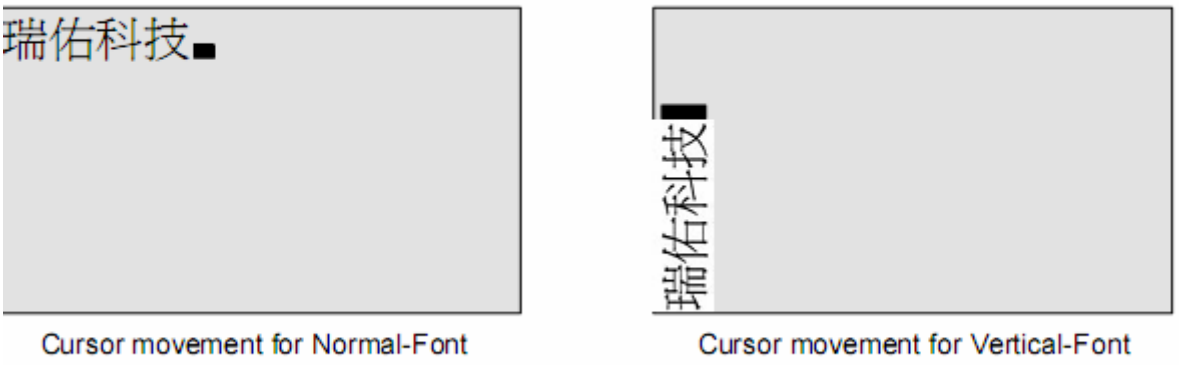


图 6-64 : 游标位移说明

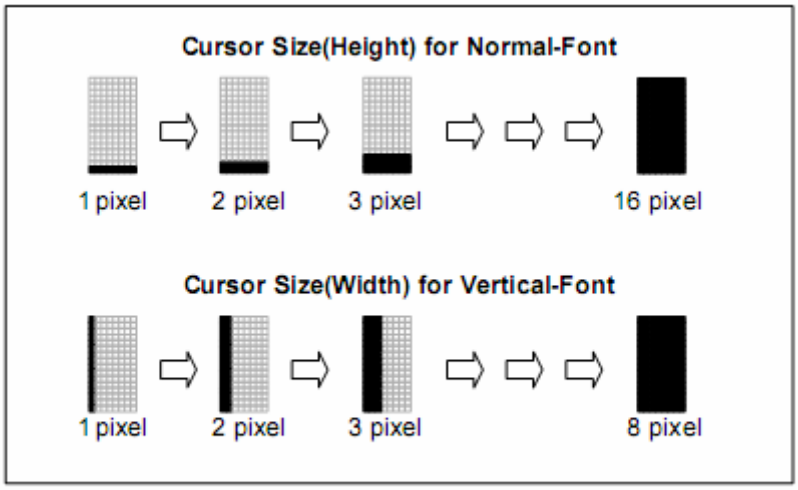


图 6-65 : 游标大小

附录C. 字码表 - ASCII

当缓存器 "FNCR" 的Bit-7 设为 "0" 时，ASCII字码表 1-4 包含之内容将如下表C- 1 ~ 表C- 4 所示。

表 C- 1: ASCII 字码表 1

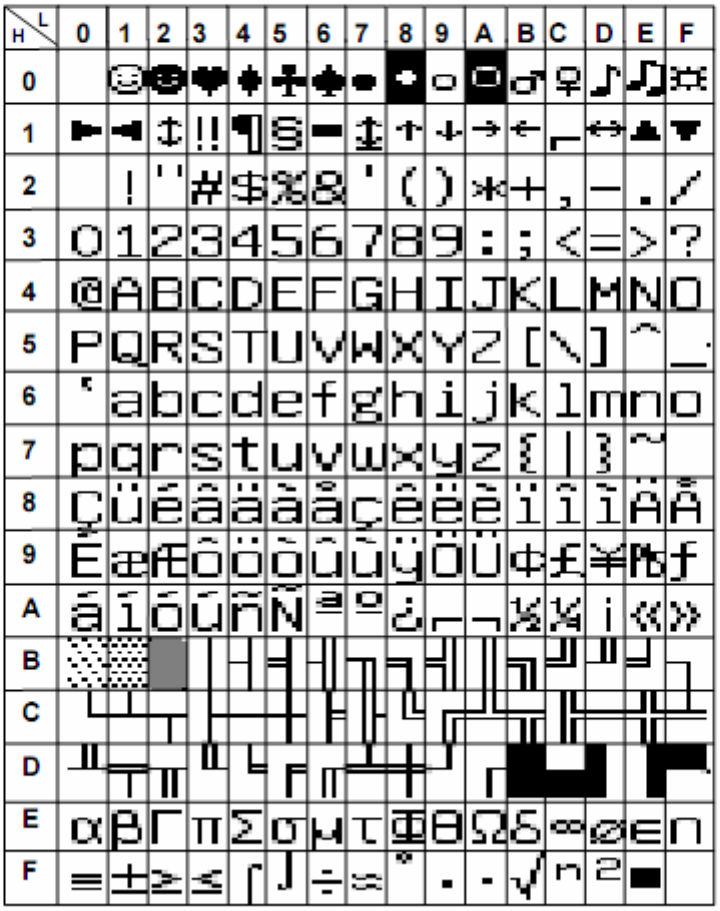


表 C- 2: ASCII 字码表 2

H \ L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	€		‚	ƒ	„	…	†	‡	ˆ	‰	Š	‹	Œ		Ž	
1		‘	’	“	”	•	–	—	˜	™	š	›	œ		ž	Ÿ
2		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬		®	¯
3	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
4	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
5	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
6	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
7	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ
8																
9																
A		Ą	˘	Ł	¤	Ľ	Ś	§	¨	Š	Ş	Ť	Ź		Ž	Ż
B	°	ą	˛	ł	´	ľ	ś	ˇ	¸	š	ş	ť	ź	˝	ž	ż
C	Ŕ	Á	Â	Ă	Ä	Ĺ	Ć	Ç	Č	É	Ę	Ë	Ě	Í	Î	Ď
D	Đ	Ń	Ň	Ó	Ô	Ő	Ö	×	Ř	Ů	Ú	Ű	Ü	Ý	Ţ	ß
E	ŕ	á	â	ă	ä	ĺ	ć	ç	č	é	ę	ë	ě	í	î	ď
F	đ	ń	ň	ó	ô	ő	ö	÷	ř	ů	ú	ű	ü	ý	ţ	˙

表 C- 3: ASCII 字码表 3

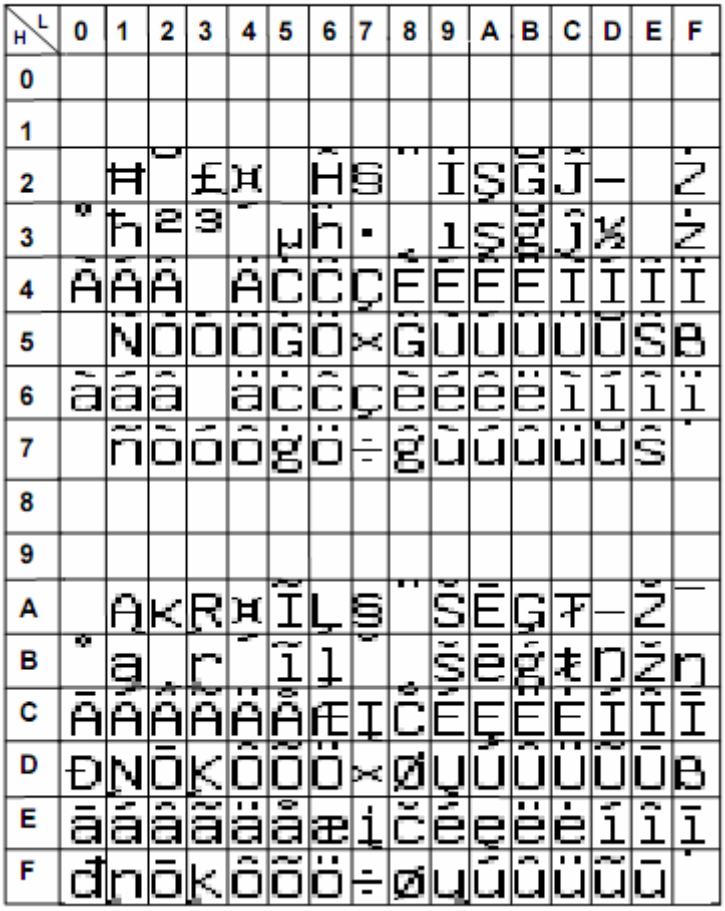
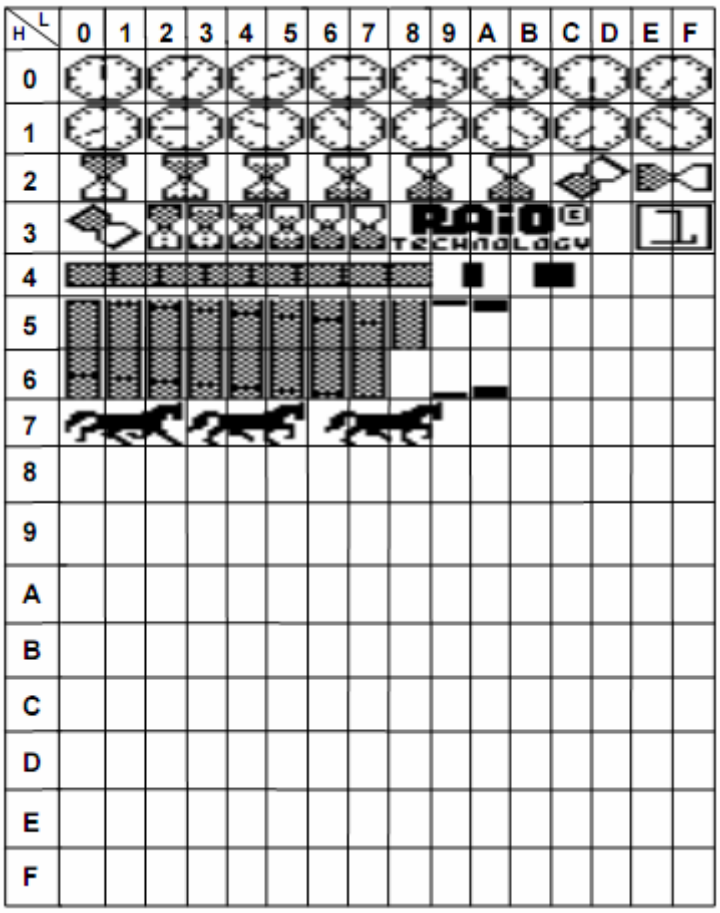


表 C- 4: ASCII 字码表 4



当缓存器 "FNCR" 的Bit-7 设为 "1" 时，ASCII字码表 1-4 包含之内容将如下表C- 5 ~ 表C- 8 所示。

表C- 5 内含符合ISO/IEC 8859-1 标准的字集，ISO是国际标准化组织的简称。ISO 8859-1 又称Latin-1 或「西欧语言」，是国际标准化组织内ISO/IEC 8859 的第一个 8 位字符集。它以ASCII为基础，在空置的0xA0 ~ 0xFF的范围内，加入 192 个字母及符号，藉以供使用变音符号的拉丁字母语言使用。此字符集支持部分于欧洲使用的语言，包括阿尔巴尼亚语、巴斯克语、布列塔尼语、加泰罗尼亚语、丹麦语、荷兰语、法罗语、弗里斯语（Frisian）、加利西亚语、德语、格陵兰语、冰岛语、爱尔兰盖尔语、意大利语、拉丁语、卢森堡语、挪威语、葡萄牙语、里托罗曼斯语、苏格兰盖尔语、西班牙语及瑞典语。

英语虽然没有重音字母，但仍会标明为 ISO 8859-1 编码。除此之外，欧洲以外的部分语言，如南非荷兰语、斯瓦希里语、印度尼西亚语及马来语、菲律宾他加洛语（Tagalog）也可使用 ISO 8859-1 编码。

表 C- 5: ASCII 字码表 1（ISO 8859-1）

H \ L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		☺	☻	♥	♦	♣	♠	●	◘	○	◙	♂	♀	♪	♫	☼
1	►	◄	↕	‼	¶	§	▬	↨	↑	↓	→	←	∟	↔	▲	▼
2	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
8																
9																
A		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬		®	¯
B	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

表C- 6 内为 ISO/IEC 8859-2 的标准字集，又称 Latin-2 或「中欧语言」，是国际标准化组织内 ISO/IEC 8859 的第二个 8 位字符集。此字符集主要支持以下文字：克罗埃西亚语、捷克语、匈牙利语、波兰语、斯洛伐克语、斯洛维尼亚语、索布语。而阿尔巴尼亚语、英语、德语、拉丁语也可用此字符集显示。芬兰语中只有于外来语才有 á 字符，若不考虑此字符，ISO/IEC 8859-2 也可用于瑞士及芬兰语。

表 C- 6: ASCII 字码表 2（ISO 8859-2）

H \ L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		☺	☻	♥	♦	♣	♠	●	◘	○	◙	♂	♀	♪	♫	☼
1	►	◄	↕	!!	¶	§	▬	↨	↑	↓	→	←	⌐	↔	▲	▼
2	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
8																
9																
A	SP	Ą	˘	Ł	¤	Ľ	Ś	§	¨	Š	Ş	Ť	Ź		Ž	Ż
B	°	ą	˛	ł	´	ľ	ś	ˇ	¸	š	ş	ť	ź	˝	ž	ż
C	Ŕ	Á	Â	Ă	Ä	Ĺ	Ć	Ç	Č	É	Ę	Ë	Ě	Í	Î	Ď
D	Đ	Ń	Ň	Ó	Ô	Ő	Ö	×	Ř	Ů	Ú	Ű	Ü	Ý	Ţ	ß
E	ŕ	á	â	ă	ä	ĺ	ć	ç	č	é	ę	ë	ě	í	î	ď
F	đ	ń	ň	ó	ô	ő	ö	÷	ř	ů	ú	ű	ü	ý	ţ	˙

表C- 7 内为ISO/IEC 8859-3 之标准字集，又称Latin-3 或「南欧语言」，是国际标准化组织内ISO/IEC 8859 的第三个 8 位字符集。它原先设计来表示土耳其语及马耳他语文字，但土耳其语已改用ISO/IEC 8859-9 显示，现时只有世界语及马耳他语仍使用此字符集。此字符集同时能支持以下文字：英语、德语、意大利语、拉丁语及葡萄牙语。

表 C- 7: ASCII 字码表 3（ISO 8859-3）

H \ L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		☺	☻	♥	♦	♣	♠	●	◘	○	◙	♂	♀	♪	♫	☼
1	►	◄	↕	!!	¶	§	▬	↨	↑	↓	→	←	⌐	↔	▲	▼
2	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
8																
9																
A	SP	Ħ	˘	£	¤		Ĥ	§	¨	İ	Ş	Ğ	Ĵ			Ż
B	°	ħ	²	³	´	µ	ĥ	·	¸	ı	ş	ğ	ĵ	½		ż
C	À	Á	Â		Ä	Ċ	Ĉ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D		Ñ	Ò	Ó	Ô	Ġ	Ö	×	Ĝ	Ù	Ú	Û	Ü	Ŭ	Ŝ	ß
E	à	á	â		ä	ċ	ĉ	ç	è	é	ê	ë	ì	í	î	ï
F		ñ	ò	ó	ô	ġ	ö	÷	ĝ	ù	ú	û	ü	ŭ	ŝ	˙

表C- 8 内为ISO/IEC 8859-4 之标准字集，又称 Latin-4 或「北欧语言」，是国际标准化组织内 ISO/IEC 8859 的第四个 8 位字符集，它设计来表示爱沙尼亚语、格陵兰语、拉脱维亚语、立陶宛语及部分萨米语（Sámi）文字，此字符集同时能支持以下文字：丹麦语、英语、芬兰语、德语、拉丁语、挪威语、斯洛维尼亚语及瑞典语。

表 C- 8: ASCII 字码表 4（ISO 8859-4）

H \ L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		☺	☻	♥	♦	♣	♠	●	◘	○	◙	♂	♀	♪	♫	☼
1	►	◄	↕	‼	¶	§	▬	↨	↑	↓	→	←	∟	↔	▲	▼
2	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
8																
9																
A	SP	Ą	ĸ	Ŗ	¤	Ĩ	Ļ	§	¨	Š	Ē	Ģ	Ŧ		Ž	¯
B	°	ą	˛	ŗ	´	ĩ	ļ	ˇ	¸	š	ē	ģ	ŧ	Ŋ	ž	ŋ
C	Ā	Á	Â	Ã	Ä	Å	Æ	Į	Č	É	Ę	Ë	Ė	Í	Î	Ī
D	Đ	Ņ	Ō	Ķ	Ô	Õ	Ö	×	Ø	Ų	Ú	Û	Ü	Ũ	Ū	ß
E	ā	á	â	ã	ä	å	æ	į	č	é	ę	ë	ė	í	î	ī
F	đ	ņ	ō	ķ	ô	õ	ö	÷	ø	ų	ú	û	ü	ũ	ū	˙

附录D. 字码表 - GB Code

A1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A			、	。	·	‐	ˇ	¨	〃	々	—	～	‖	…	‘	’
B	“	”	〔	〕	〈	〉	《	》	「	」	『	』	〖	〗	【	】
C	±	×	÷	∶	∧	∨	∑	∏	∪	∩	∈	∷	√	⊥	∥	∠
D	⌒	⊙	∫	∮	≡	≌	≈	∽	∝	≠	≮	≯	≤	≥	∞	∵
E	∴	♂	♀	°	′	″	℃	$	¤	￠	￡	‰	§	№	☆	★
F	○	●	◎	◇	◆	□	■	△	▲	※	→	←	↑	↓	〓	

A2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A																
B		1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.	13.	14.	15.
C	16.	17.	18.	19.	20.	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)
D	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(20)	①	②	③	④	⑤	⑥	⑦
E	⑧	⑨	⑩			㈠	㈡	㈢	㈣	㈤	㈥	㈦	㈧	㈨	㈩	
F		Ⅰ	Ⅱ	Ⅲ	Ⅳ	Ⅴ	Ⅵ	Ⅶ	Ⅷ	Ⅸ	Ⅹ	Ⅺ	Ⅻ			

A3	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		!	"	#	￥	%	&	'	(	)	*	+	,	—	.	/
B	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
C	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
D	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
E	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
F	p	q	r	s	t	u	v	w	x	y	z	{	|	}	￣	

A3	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		！	＂	＃	￥	％	＆	＇	（	）	＊	＋	，	－	．	／
B	０	１	２	３	４	５	６	７	８	９	：	；	＜	＝	＞	？
C	＠	Ａ	Ｂ	Ｃ	Ｄ	Ｅ	Ｆ	Ｇ	Ｈ	Ｉ	Ｊ	Ｋ	Ｌ	Ｍ	Ｎ	Ｏ
D	Ｐ	Ｑ	Ｒ	Ｓ	Ｔ	Ｕ	Ｖ	Ｗ	Ｘ	Ｙ	Ｚ	［	＼	］	＾	＿
E	｀	ａ	ｂ	ｃ	ｄ	ｅ	ｆ	ｇ	ｈ	ｉ	ｊ	ｋ	ｌ	ｍ	ｎ	ｏ
F	ｐ	ｑ	ｒ	ｓ	ｔ	ｕ	ｖ	ｗ	ｘ	ｙ	ｚ	｛	｜	｝	￣	

A4	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		ぁ	あ	ぃ	い	ぅ	う	ぇ	え	ぉ	お	か	が	き	ぎ	く
B	ぐ	け	げ	こ	ご	さ	ざ	し	じ	す	ず	せ	ぜ	そ	ぞ	た
C	だ	ち	ぢ	っ	つ	づ	て	で	と	ど	な	に	ぬ	ね	の	は
D	ば	ぱ	ひ	び	ぴ	ふ	ぶ	ぷ	へ	べ	ぺ	ほ	ぼ	ぽ	ま	み
E	む	め	も	ゃ	や	ゅ	ゆ	ょ	よ	ら	り	る	れ	ろ	ゎ	わ
F	ゐ	ゑ	を	ん												

A5	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		ァ	ア	ィ	イ	ゥ	ウ	ェ	エ	ォ	オ	カ	ガ	キ	ギ	ク
B	グ	ケ	ゲ	コ	ゴ	サ	ザ	シ	ジ	ス	ズ	セ	ゼ	ソ	ゾ	タ
C	ダ	チ	ヂ	ッ	ツ	ヅ	テ	デ	ト	ド	ナ	ニ	ヌ	ネ	ノ	ハ
D	バ	パ	ヒ	ビ	ピ	フ	ブ	プ	ヘ	ベ	ペ	ホ	ボ	ポ	マ	ミ
E	ム	メ	モ	ャ	ヤ	ュ	ユ	ョ	ヨ	ラ	リ	ル	レ	ロ	ヮ	ワ
F	ヰ	ヱ	ヲ	ン	ヴ	ヵ	ヶ									

A6	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι	Κ	Λ	Μ	Ν	Ξ	Ο
B	Π	Ρ	Σ	Τ	Υ	Φ	Χ	Ψ	Ω							
C		α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο
D	π	ρ	σ	τ	υ	φ	χ	ψ	ω							
E																
F																

A7	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н
B	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э
C	Ю	Я														
D		а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н
E	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э
F	ю	я														

A8	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		ō	ó	ǒ	ò	ē	é	ě	è	ī	ǐ	í	ì	ō	ó	ǒ
B	ò	ū	ú	ǔ	ù	ǖ	ǘ	ǚ	ǜ	ü	ê	ɑ	ḿ	ń	ň	ǹ
C	ɡ					ㄅ	ㄆ	ㄇ	ㄈ	ㄉ	ㄊ	ㄋ	ㄌ	ㄍ	ㄎ	ㄏ
D	ㄐ	ㄑ	ㄒ	ㄓ	ㄔ	ㄕ	ㄖ	ㄗ	ㄘ	ㄙ	ㄚ	ㄛ	ㄜ	ㄝ	ㄞ	ㄟ
E	ㄠ	ㄡ	ㄢ	ㄣ	ㄤ	ㄥ	ㄦ	ㄧ	ㄨ	ㄩ						
F																

A9	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A					—	—	|	|	┄	┅	┆	┇	┈	┉	┊	┋
B	┌	┌	┌	┌	┐	┐	┐	┐	└	└	└	└	┘	┘	┘	┘
C	├	├	├	├	├	├	├	├	┤	┤	┤	┤	┤	┤	┤	┤
D	┬	┬	┬	┬	┬	┬	┬	┬	┴	┴	┴	┴	┴	┴	┴	┴
E	┼	┼	┼	┼	┼	┼	┼	┼	┼	┼	┼	┼	┼	┼	┼	┼
F																

AA	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A																
B																
C																
D																
E																
F																

AB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A																
B																
C																
D																
E																
F																

AC	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A																
B																
C																
D																
E																
F																

AD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A																
B																
C																
D																
E																
F																

AE	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A																
B																
C																
D																
E																
F																

AF	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A																
B																
C																
D																
E																
F																

B0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		啊	阿	埃	挨	哎	唉	哀	皑	癌	蔼	矮	艾	碍	爱	隘
B	鞍	氨	安	俺	按	暗	岸	胺	案	肮	昂	盎	凹	敖	熬	翱
C	袄	傲	奥	懊	澳	芭	捌	扒	叭	吧	笆	八	疤	巴	拔	跋
D	靶	把	耙	坝	霸	罢	爸	白	柏	百	摆	佰	败	拜	稗	斑
E	班	搬	扳	般	颁	板	版	扮	拌	伴	瓣	半	办	绊	邦	帮
F	梆	榜	膀	绑	棒	磅	蚌	镑	傍	谤	苞	胞	包	褒	剥	

B1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		薄	雹	保	堡	饱	宝	抱	报	暴	豹	鲍	爆	杯	碑	悲
B	卑	北	辈	背	贝	钡	倍	狈	备	惫	焙	被	奔	苯	本	笨
C	崩	绷	甭	泵	蹦	迸	逼	鼻	比	鄙	笔	彼	碧	蓖	蔽	毕
D	毙	毖	币	庇	痹	闭	敝	弊	必	辟	壁	臂	避	陛	鞭	边
E	编	贬	扁	便	变	卞	辨	辩	辫	遍	标	彪	膘	表	鳖	憋
F	别	瘪	彬	斌	濒	滨	宾	摈	兵	冰	柄	丙	秉	饼	炳	

B2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		病	并	玻	菠	播	拨	钵	波	博	勃	搏	铂	箔	伯	帛
B	舶	脖	膊	渤	泊	驳	捕	卜	哺	补	埠	不	布	步	簿	部
C	怖	擦	猜	裁	材	才	财	睬	踩	采	彩	菜	蔡	餐	参	蚕
D	残	惭	惨	灿	苍	舱	仓	沧	藏	操	糙	槽	曹	草	厕	策
E	侧	册	测	层	蹭	插	叉	茬	茶	查	碴	搽	察	岔	差	诧
F	拆	柴	豺	搀	掺	蝉	馋	谗	缠	铲	产	阐	颤	昌	猖	

B3	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		场	尝	常	长	偿	肠	厂	敞	畅	唱	倡	超	抄	钞	朝
B	嘲	潮	巢	吵	炒	车	扯	撤	掣	彻	澈	郴	臣	辰	尘	晨
C	忱	沉	陈	趁	衬	撑	称	城	橙	成	呈	乘	程	惩	澄	诚
D	承	逞	骋	秤	吃	痴	持	匙	池	迟	弛	驰	耻	齿	侈	尺
E	赤	翅	斥	炽	充	冲	虫	崇	宠	抽	酬	畴	踌	稠	愁	筹
F	仇	绸	瞅	丑	臭	初	出	橱	厨	躇	锄	雏	滁	除	楚	

B4	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		础	储	矗	搐	触	处	揣	川	穿	椽	传	船	喘	串	疮
B	窗	幢	床	闯	创	吹	炊	捶	锤	垂	春	椿	醇	唇	淳	纯
C	蠢	戳	绰	疵	茨	磁	雌	辞	慈	瓷	词	此	刺	赐	次	聪
D	葱	囱	匆	从	丛	凑	粗	醋	簇	促	蹿	篡	窜	摧	崔	催
E	脆	瘁	粹	淬	翠	村	存	寸	磋	撮	搓	措	挫	错	搭	达
F	答	瘩	打	大	呆	歹	傣	戴	带	殆	代	贷	袋	待	逮	

B5	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		怠	耽	担	丹	单	郸	掸	胆	旦	氮	但	惮	淡	诞	弹
B	蛋	当	挡	党	荡	档	刀	捣	蹈	倒	岛	祷	导	到	稻	悼
C	道	盗	德	得	的	蹬	灯	登	等	瞪	凳	邓	堤	低	滴	迪
D	敌	笛	狄	涤	翟	嫡	抵	底	地	蒂	第	帝	弟	递	缔	颠
E	掂	滇	碘	点	典	靛	垫	电	佃	甸	店	惦	奠	淀	殿	碉
F	叮	雕	调	刁	掉	吊	钓	调	跌	爹	碟	蝶	迭	谍	叠	

B6	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		丁	盯	叮	钉	顶	鼎	锭	定	订	丢	东	冬	董	懂	动
B	栋	侗	恫	冻	洞	兜	抖	斗	陡	豆	逗	痘	都	督	毒	犊
C	独	读	堵	睹	赌	杜	镀	肚	度	渡	妒	端	短	锻	段	断
D	缎	堆	兑	队	对	墩	吨	蹲	敦	顿	囤	钝	盾	遁	掇	哆
E	多	夺	垛	躲	朵	跺	舵	剁	惰	堕	蛾	峨	鹅	俄	额	讹
F	娥	恶	厄	扼	遏	鄂	饿	恩	而	儿	耳	尔	饵	洱	二	

B7	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		贰	发	罚	筏	伐	乏	阀	法	珐	藩	帆	番	翻	樊	矾
B	钒	繁	凡	烦	反	返	范	贩	犯	饭	泛	坊	芳	方	肪	房
C	防	妨	仿	访	纺	放	菲	非	啡	飞	肥	匪	诽	吠	肺	废
D	沸	费	芬	酚	吩	氛	分	纷	坟	焚	汾	粉	奋	份	忿	愤
E	粪	丰	封	枫	蜂	峰	锋	风	疯	烽	逢	冯	缝	讽	奉	凤
F	佛	否	夫	敷	肤	孵	扶	拂	辐	幅	氟	符	伏	俘	服	

B8	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		浮	涪	福	袱	弗	甫	抚	辅	俯	釜	斧	脯	腑	府	腐
B	赴	副	覆	赋	复	傅	付	阜	父	腹	负	富	讣	附	妇	缚
C	咐	噶	嘎	该	改	概	钙	盖	溉	干	甘	杆	柑	竿	肝	赶
D	感	秆	敢	赣	冈	刚	钢	缸	肛	纲	岗	港	杠	篙	皋	高
E	膏	羔	糕	搞	镐	稿	告	哥	歌	搁	戈	鸽	胳	疙	割	革
F	葛	格	蛤	阁	隔	铬	个	各	给	根	跟	耕	更	庚	羹	

B9	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		埂	耿	梗	工	攻	功	恭	龚	供	躬	公	宫	弓	巩	汞
B	拱	贡	共	钩	勾	沟	苟	狗	垢	构	购	够	辜	菇	咕	箍
C	估	沽	孤	姑	鼓	古	蛊	骨	谷	股	故	顾	固	雇	刮	瓜
D	剐	寡	挂	褂	乖	拐	怪	棺	关	官	冠	观	管	馆	罐	惯
E	灌	贯	光	广	逛	瑰	规	圭	硅	归	龟	闺	轨	鬼	诡	癸
F	桂	柜	跪	贵	刽	辊	滚	棍	锅	郭	国	果	裹	过	哈	

BA	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		骸	孩	海	氦	亥	害	骇	酣	憨	邯	韩	含	涵	寒	函
B	喊	罕	翰	撼	捍	旱	憾	悍	焊	汗	汉	夯	杭	航	壕	嚎
C	豪	毫	郝	好	耗	号	浩	呵	喝	荷	菏	核	禾	和	何	合
D	盒	貉	阂	河	涸	赫	褐	鹤	贺	嘿	黑	痕	很	狠	恨	哼
E	亨	横	衡	恒	轰	哄	烘	虹	鸿	洪	宏	弘	红	喉	侯	猴
F	吼	厚	候	后	呼	乎	忽	瑚	壶	葫	胡	蝴	狐	糊	湖	

BB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		弧	虎	唬	护	互	沪	户	花	哗	华	猾	滑	画	划	化
B	话	槐	徊	怀	淮	坏	欢	环	桓	还	缓	换	患	唤	痪	豢
C	焕	涣	宦	幻	荒	慌	黄	磺	蝗	簧	皇	凰	惶	煌	晃	幌
D	恍	谎	灰	挥	辉	徽	恢	蛔	回	毁	悔	慧	卉	惠	晦	贿
E	秽	会	烩	汇	讳	诲	绘	荤	昏	婚	魂	浑	混	豁	活	伙
F	火	获	或	惑	霍	货	祸	击	圾	基	机	畸	稽	积	箕	

BC	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		肌	饥	迹	激	讥	鸡	姬	绩	缉	吉	极	棘	辑	籍	集
B	及	急	疾	汲	即	嫉	级	挤	几	脊	己	蓟	技	冀	季	伎
C	祭	剂	悸	济	寄	寂	计	记	既	忌	际	妓	继	纪	嘉	枷
D	夹	佳	家	加	荚	颊	贾	甲	钾	假	稼	价	架	驾	嫁	歼
E	监	坚	尖	笺	间	煎	兼	肩	艰	奸	缄	茧	检	柬	碱	硷
F	拣	捡	简	俭	剪	减	荐	槛	鉴	践	贱	见	键	箭	件	

BD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		健	舰	剑	饯	渐	溅	涧	建	僵	姜	将	浆	江	疆	蒋
B	桨	奖	讲	匠	酱	降	蕉	椒	礁	焦	胶	交	郊	浇	骄	娇
C	嚼	搅	铰	矫	侥	脚	狡	角	饺	缴	绞	剿	教	酵	轿	较
D	叫	窖	揭	接	皆	秸	街	阶	截	劫	节	桔	杰	捷	睫	竭
E	洁	结	解	姐	戒	藉	芥	界	借	介	疥	诫	届	巾	筋	斤
F	金	今	津	襟	紧	锦	仅	谨	进	靳	晋	禁	近	烬	浸	

BE	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		尽	劲	荆	兢	茎	睛	晶	鲸	京	惊	精	粳	经	井	警
B	景	颈	静	境	敬	镜	径	痉	靖	竟	竞	净	炯	窘	揪	究
C	纠	玖	韭	久	灸	九	酒	厩	救	旧	臼	舅	咎	就	疚	鞠
D	拘	狙	疽	居	驹	菊	局	咀	矩	举	沮	聚	拒	据	巨	具
E	距	踞	锯	俱	句	惧	炬	剧	捐	鹃	娟	倦	眷	卷	绢	撅
F	攫	抉	掘	倔	爵	觉	决	诀	绝	均	菌	钧	军	君	峻	

BF	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		俊	竣	浚	郡	骏	喀	咖	卡	咯	开	揩	楷	凯	慨	刊
B	堪	勘	坎	砍	看	康	慷	糠	扛	抗	亢	炕	考	拷	烤	靠
C	坷	苛	柯	棵	磕	颗	科	壳	咳	可	渴	克	刻	客	课	肯
D	啃	垦	恳	坑	吭	空	恐	孔	控	抠	口	扣	寇	枯	哭	窟
E	苦	酷	库	裤	夸	垮	挎	跨	胯	块	筷	侩	快	宽	款	匡
F	筐	狂	框	矿	眶	旷	况	亏	盔	岿	窥	葵	奎	魁	傀	

C0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		馈	愧	溃	坤	昆	捆	困	括	扩	廓	阔	垃	拉	喇	蜡
B	腊	辣	啦	莱	来	赖	蓝	婪	栏	拦	篮	阑	兰	澜	谰	揽
C	览	懒	缆	烂	滥	琅	榔	狼	廊	郎	朗	浪	捞	劳	牢	老
D	佬	姥	酪	烙	涝	勒	乐	雷	镭	蕾	磊	累	儡	垒	擂	肋
E	类	泪	棱	楞	冷	厘	梨	犁	黎	篱	狸	离	漓	理	李	里
F	鲤	礼	莉	荔	吏	栗	丽	厉	励	砾	历	利	傈	例	俐	

C1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		痢	立	粒	沥	隶	力	璃	哩	俩	联	莲	连	镰	廉	怜
B	涟	帘	敛	脸	链	恋	炼	练	粮	凉	梁	粱	良	两	辆	量
C	晾	亮	谅	撩	聊	僚	疗	燎	寥	辽	潦	了	撂	镣	廖	料
D	列	裂	烈	劣	猎	琳	林	磷	霖	临	邻	鳞	淋	凛	赁	吝
E	拎	玲	菱	零	龄	铃	伶	羚	凌	灵	陵	岭	领	另	令	溜
F	琉	榴	硫	馏	留	刘	瘤	流	柳	六	龙	聋	咙	笼	窿	

C2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		隆	垄	拢	陇	楼	娄	搂	篓	漏	陋	芦	卢	颅	庐	炉
B	掳	卤	虏	鲁	麓	碌	露	路	赂	鹿	潞	禄	录	陆	戮	驴
C	吕	铝	侣	旅	履	屡	缕	虑	氯	律	率	滤	绿	峦	挛	孪
D	滦	卵	乱	掠	略	抡	轮	伦	仑	沦	纶	论	萝	螺	罗	逻
E	锣	箩	骡	裸	落	洛	骆	络	妈	麻	玛	码	蚂	马	骂	嘛
F	吗	埋	买	麦	卖	迈	脉	瞒	馒	蛮	满	蔓	曼	慢	漫	

C3	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		谩	芒	茫	盲	氓	忙	莽	猫	茅	锚	毛	矛	铆	卯	茂
B	冒	帽	貌	贸	么	玫	枚	梅	酶	霉	煤	没	眉	媒	镁	每
C	美	昧	寐	妹	媚	门	闷	们	萌	蒙	檬	盟	锰	猛	梦	孟
D	眯	醚	靡	糜	迷	谜	弥	米	秘	觅	泌	蜜	密	幂	棉	眠
E	绵	冕	免	勉	娩	缅	面	苗	描	瞄	藐	秒	渺	庙	妙	蔑
F	灭	民	抿	皿	敏	悯	闽	明	螟	鸣	铭	名	命	谬	摸	

C4	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		摹	蘑	模	膜	磨	摩	魔	抹	末	莫	墨	默	沫	漠	寞
B	陌	谋	牟	某	拇	牡	亩	姆	母	墓	暮	幕	募	慕	木	目
C	睦	牧	穆	拿	哪	呐	钠	那	娜	纳	氖	乃	奶	耐	奈	南
D	男	难	囊	挠	脑	恼	闹	淖	呢	馁	内	嫩	能	妮	霓	倪
E	泥	尼	拟	你	匿	腻	逆	溺	蔫	拈	年	碾	撵	捻	念	娘
F	酿	鸟	尿	捏	聂	孽	啮	镊	镍	涅	您	柠	狞	凝	宁	

C5	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		拧	泞	牛	扭	钮	纽	脓	浓	农	弄	奴	努	怒	女	暖
B	虐	疟	挪	懦	糯	诺	哦	欧	鸥	殴	藕	呕	偶	沤	啪	趴
C	爬	帕	怕	琶	拍	排	牌	徘	湃	派	攀	潘	盘	磐	盼	畔
D	判	叛	乓	庞	旁	耪	胖	抛	咆	刨	炮	袍	跑	泡	呸	胚
E	培	裴	赔	陪	配	佩	沛	喷	盆	砰	抨	烹	澎	彭	蓬	棚
F	硼	篷	膨	朋	鹏	捧	碰	坯	砒	霹	批	披	劈	琵	毗	

C6	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		啤	脾	疲	皮	匹	痞	僻	屁	譬	篇	偏	片	骗	飘	漂
B	瓢	票	撇	瞥	拼	频	贫	品	聘	乒	坪	苹	萍	平	凭	瓶
C	评	屏	坡	泼	颇	婆	破	魄	迫	粕	剖	扑	铺	仆	莆	葡
D	菩	蒲	埔	朴	圃	普	浦	谱	曝	瀑	期	欺	栖	戚	妻	七
E	凄	漆	柒	沏	其	棋	奇	歧	畦	崎	脐	齐	旗	祈	祁	骑
F	起	岂	乞	企	启	契	砌	器	气	迄	弃	汽	泣	讫	掐	

C7	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		恰	洽	牵	扦	钎	铅	千	迁	签	仟	谦	乾	黔	钱	钳
B	前	潜	遣	浅	谴	堑	嵌	欠	歉	枪	呛	腔	羌	墙	蔷	强
C	抢	橇	锹	敲	悄	桥	瞧	乔	侨	巧	鞘	撬	翘	峭	俏	窍
D	切	茄	且	怯	窃	钦	侵	亲	秦	琴	勤	芹	擒	禽	寝	沁
E	青	轻	氢	倾	卿	清	擎	晴	氰	情	顷	请	庆	琼	穷	秋
F	丘	邱	球	求	囚	酋	泅	趋	区	蛆	曲	躯	屈	驱	渠	

C8	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		取	娶	龋	趣	去	圈	颧	权	醛	泉	全	痊	拳	犬	券
B	劝	缺	炔	瘸	却	鹊	榷	确	雀	裙	群	然	燃	冉	染	瓤
C	壤	攘	嚷	让	饶	扰	绕	惹	热	壬	仁	人	忍	韧	任	认
D	刃	妊	纫	扔	仍	日	戎	茸	蓉	荣	融	熔	溶	容	绒	冗
E	揉	柔	肉	茹	蠕	儒	孺	如	辱	乳	汝	入	褥	软	阮	蕊
F	瑞	锐	闰	润	若	弱	撒	洒	萨	腮	鳃	塞	赛	三	叁	

C9	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		伞	散	桑	嗓	丧	搔	骚	扫	嫂	瑟	色	涩	森	僧	莎
B	砂	杀	刹	沙	纱	傻	啥	煞	筛	晒	珊	苫	杉	山	删	煽
C	衫	闪	陕	擅	赡	膳	善	汕	扇	缮	墒	伤	商	赏	晌	上
D	尚	裳	梢	捎	稍	烧	芍	勺	韶	少	哨	邵	绍	奢	赊	蛇
E	舌	舍	赦	摄	射	慑	涉	社	设	砷	申	呻	伸	身	深	娠
F	绅	神	沈	审	婶	甚	肾	慎	渗	声	生	甥	牲	升	绳	

CA	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		省	盛	剩	胜	圣	师	失	狮	施	湿	诗	尸	虱	十	石
B	拾	时	什	食	蚀	实	识	史	矢	使	屎	驶	始	式	示	士
C	世	柿	事	拭	誓	逝	势	是	嗜	噬	适	仕	侍	释	饰	氏
D	市	恃	室	视	试	收	手	首	守	寿	授	售	受	瘦	兽	蔬
E	枢	梳	殊	抒	输	叔	舒	淑	疏	书	赎	孰	熟	薯	暑	曙
F	署	蜀	黍	鼠	属	术	述	树	束	戍	竖	墅	庶	数	漱	

CB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		恕	刷	耍	摔	衰	甩	帅	栓	拴	霜	双	爽	谁	水	睡
B	税	吮	瞬	顺	舜	说	硕	朔	烁	斯	撕	嘶	思	私	司	丝
C	死	肆	寺	嗣	四	伺	似	饲	巳	松	耸	怂	颂	送	宋	讼
D	诵	搜	艘	擞	嗽	苏	酥	俗	素	速	粟	僳	塑	溯	宿	诉
E	肃	酸	蒜	算	虽	隋	随	绥	髓	碎	岁	穗	遂	隧	祟	孙
F	损	笋	蓑	梭	唆	缩	琐	索	锁	所	塌	他	它	她	塔	

CC	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		獭	挞	蹋	踏	胎	苔	抬	台	泰	酞	太	态	汰	坍	摊
B	贪	瘫	滩	坛	檀	痰	潭	谭	谈	坦	毯	袒	碳	探	叹	炭
C	汤	塘	搪	堂	棠	膛	唐	糖	倘	躺	淌	趟	烫	掏	涛	滔
D	绦	萄	桃	逃	淘	陶	讨	套	特	藤	腾	疼	誊	梯	剔	踢
E	锑	提	题	蹄	啼	体	替	嚏	惕	涕	剃	屉	天	添	填	田
F	甜	恬	舔	腆	挑	条	迢	眺	跳	贴	铁	帖	厅	听	烃	

CD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		汀	廷	停	亭	庭	挺	艇	通	桐	酮	瞳	同	铜	彤	童
B	桶	捅	筒	统	痛	偷	投	头	透	凸	秃	突	图	徒	途	涂
C	屠	土	吐	兔	湍	团	推	颓	腿	蜕	褪	退	吞	屯	臀	拖
D	托	脱	鸵	陀	驮	驼	椭	妥	拓	唾	挖	哇	蛙	洼	娃	瓦
E	袜	歪	外	豌	弯	湾	玩	顽	丸	烷	完	碗	挽	晚	皖	惋
F	宛	婉	万	腕	汪	王	亡	枉	网	往	旺	望	忘	妄	威	

CE	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		巍	微	危	韦	违	桅	围	唯	惟	为	潍	维	苇	萎	委
B	伟	伪	尾	纬	未	蔚	味	畏	胃	喂	魏	位	渭	谓	尉	慰
C	卫	瘟	温	蚊	文	闻	纹	吻	稳	紊	问	嗡	翁	瓮	挝	蜗
D	涡	窝	我	斡	卧	握	沃	巫	呜	钨	乌	污	诬	屋	无	芜
E	梧	吾	吴	毋	武	五	捂	午	舞	伍	侮	坞	戊	雾	晤	物
F	勿	务	悟	误	昔	熙	析	西	硒	矽	晰	嘻	吸	锡	牺	

CF	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		稀	息	希	悉	膝	夕	惜	熄	烯	溪	汐	犀	檄	袭	席
B	习	媳	喜	铣	洗	系	隙	戏	细	瞎	虾	匣	霞	辖	暇	峡
C	侠	狭	下	厦	夏	吓	掀	锨	先	仙	鲜	纤	咸	贤	衔	舷
D	闲	涎	弦	嫌	显	险	现	献	县	腺	馅	羡	宪	陷	限	线
E	相	厢	镶	香	箱	襄	湘	乡	翔	祥	详	想	响	享	项	巷
F	橡	像	向	象	萧	硝	霄	削	哮	嚣	销	消	宵	淆	晓	

D0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		小	孝	校	肖	啸	笑	效	楔	些	歇	蝎	鞋	协	挟	携
B	邪	斜	胁	谐	写	械	卸	蟹	懈	泄	泻	谢	屑	薪	芯	锌
C	欣	辛	新	忻	心	信	衅	星	腥	猩	惺	兴	刑	型	形	邢
D	行	醒	幸	杏	性	姓	兄	凶	胸	匈	汹	雄	熊	休	修	羞
E	朽	嗅	锈	秀	袖	绣	墟	戌	需	虚	嘘	须	徐	许	蓄	酗
F	叙	旭	序	畜	恤	絮	婿	绪	续	轩	喧	宣	悬	旋	玄	

D1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		选	癣	眩	绚	靴	薛	学	穴	雪	血	勋	熏	循	旬	询
B	寻	驯	巡	殉	汛	训	讯	逊	迅	压	押	鸦	鸭	呀	丫	芽
C	牙	蚜	崖	衙	涯	雅	哑	亚	讶	焉	咽	阉	烟	淹	盐	严
D	研	蜒	岩	延	言	颜	阎	炎	沿	奄	掩	眼	衍	演	艳	堰
E	燕	厌	砚	雁	唁	彦	焰	宴	谚	验	殃	央	鸯	秧	杨	扬
F	佯	疡	羊	洋	阳	氧	仰	痒	养	样	漾	邀	腰	妖	瑶	

D2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		摇	尧	遥	窑	谣	姚	咬	舀	药	要	耀	椰	噎	耶	爷
B	野	冶	也	页	掖	业	叶	曳	腋	夜	液	一	壹	医	揖	铱
C	依	伊	衣	颐	夷	遗	移	仪	胰	疑	沂	宜	姨	彝	椅	蚁
D	倚	已	乙	矣	以	艺	抑	易	邑	屹	亿	役	臆	逸	肄	疫
E	亦	裔	意	毅	忆	义	益	溢	诣	议	谊	译	异	翼	翌	绎
F	茵	荫	因	殷	音	阴	姻	吟	银	淫	寅	饮	尹	引	隐	

D3	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		印	英	樱	婴	鹰	应	缨	莹	萤	营	荧	蝇	迎	赢	盈
B	影	颖	硬	映	哟	拥	佣	臃	痈	庸	雍	踊	蛹	咏	泳	涌
C	永	恿	勇	用	幽	优	悠	忧	尤	由	邮	铀	犹	油	游	酉
D	有	友	右	佑	釉	诱	又	幼	迂	淤	于	盂	榆	虞	愚	舆
E	余	俞	逾	鱼	愉	渝	渔	隅	予	娱	雨	与	屿	禹	宇	语
F	羽	玉	域	芋	郁	吁	遇	喻	峪	御	愈	欲	狱	育	誉	

D4	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		浴	寓	裕	预	豫	驭	鸳	渊	冤	元	垣	袁	原	援	辕
B	园	员	圆	猿	源	缘	远	苑	愿	怨	院	曰	约	越	跃	钥
C	岳	粤	月	悦	阅	耘	云	郧	匀	陨	允	运	蕴	酝	晕	韵
D	孕	匝	砸	杂	栽	哉	灾	宰	载	再	在	咱	攒	暂	赞	赃
E	脏	葬	遭	糟	凿	藻	枣	早	澡	蚤	躁	噪	造	皂	灶	燥
F	责	择	则	泽	贼	怎	增	憎	曾	赠	扎	喳	渣	札	轧	

D5	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		铡	闸	眨	栅	榨	咋	乍	炸	诈	摘	斋	宅	窄	债	寨
B	瞻	毡	詹	粘	沾	盏	斩	辗	崭	展	蘸	栈	占	战	站	湛
C	绽	樟	章	彰	漳	张	掌	涨	杖	丈	帐	账	仗	胀	瘴	障
D	招	昭	找	沼	赵	照	罩	兆	肇	召	遮	折	哲	蛰	辙	者
E	锗	蔗	这	浙	珍	斟	真	甄	砧	臻	贞	针	侦	枕	疹	诊
F	震	振	镇	阵	蒸	挣	睁	征	狰	争	怔	整	拯	正	政	

D6	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		帧	症	郑	证	芝	枝	支	吱	蜘	知	肢	脂	汁	之	织
B	职	直	植	殖	执	值	侄	址	指	止	趾	只	旨	纸	志	挚
C	掷	至	致	置	帜	峙	制	智	秩	稚	质	炙	痔	滞	治	窒
D	中	盅	忠	钟	衷	终	种	肿	重	仲	众	舟	周	州	洲	诌
E	粥	轴	肘	帚	咒	皱	宙	昼	骤	珠	株	蛛	朱	猪	诸	诛
F	逐	竹	烛	煮	拄	瞩	嘱	主	著	柱	助	蛀	贮	铸	筑	

D7	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		住	注	祝	驻	抓	爪	拽	专	砖	转	撰	赚	篆	桩	庄
B	装	妆	撞	壮	状	椎	锥	追	赘	坠	缀	谆	准	捉	拙	卓
C	桌	琢	茁	酌	啄	着	灼	浊	兹	咨	资	姿	滋	淄	孜	紫
D	仔	籽	滓	子	自	渍	字	鬃	棕	踪	宗	综	总	纵	邹	走
E	奏	揍	租	足	卒	族	祖	诅	阻	组	钻	纂	嘴	醉	最	罪
F	尊	遵	昨	左	佐	柞	做	作	坐	座						

D8	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		亍	丌	兀	丐	廿	卅	丕	亘	丞	鬲	孬	噩	丨	禺	丿
B	匕	乇	夭	爻	卮	氐	囟	胤	馗	毓	睾	鼗	丶	亟	鼐	乜
C	乩	亓	芈	孛	啬	嘏	仄	厍	厝	厣	厥	厮	靥	赝	匚	叵
D	匦	匮	匾	赜	卦	卣	刂	刈	刎	刭	刳	刿	剀	剌	剞	剡
E	剜	蒯	剽	劂	劁	劐	劓	冂	罔	亻	仃	仉	仂	仨	仡	仫
F	仞	伛	仳	伢	佤	仵	伥	伧	伉	伫	佞	佧	攸	佚	佝	

D9	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		佟	佗	伲	伽	佶	佴	侑	侉	侃	侏	佾	佻	侪	佼	侬
B	侔	俦	俨	俪	俅	俚	俣	俜	俑	俟	俸	倩	偌	俳	倬	倏
C	倮	倭	俾	倜	倌	倥	倨	偾	偃	偕	偈	偎	偬	偻	傥	傧
D	傩	傺	僖	儆	僭	僬	僦	僮	儇	儋	仝	氽	佘	佥	俎	龠
E	汆	籴	兮	巽	黉	馘	冁	夔	勹	匍	訇	匐	凫	夙	兕	亠
F	兖	亳	衮	袤	亵	脔	裒	禀	嬴	蠃	羸	冫	冱	冽	冼	

DA	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		凇	冖	冢	冥	讠	讦	讧	讪	讴	讵	讷	诂	诃	诋	诏
B	诎	诒	诓	诔	诖	诘	诙	诜	诟	诠	诤	诨	诩	诮	诰	诳
C	诶	诹	诼	诿	谀	谂	谄	谇	谌	谏	谑	谒	谔	谕	谖	谙
D	谛	谘	谝	谟	谠	谡	谥	谧	谪	谫	谮	谯	谲	谳	谵	谶
E	卩	卺	阝	阢	阡	阱	阪	阽	阼	陂	陉	陔	陟	陧	陬	陲
F	陴	隈	隍	隗	隰	邗	邛	邝	邙	邬	邡	邴	邳	邶	邺	

DB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		邸	邰	郏	郅	邾	郐	郄	郇	郓	郦	郢	郜	郗	郛	郫
B	郯	郾	鄄	鄢	鄞	鄣	鄱	鄯	鄹	酃	酆	刍	奂	劢	劬	劭
C	劾	哿	勐	勖	勰	叟	燮	矍	廴	凵	凼	鬯	厶	弁	畚	巯
D	坌	垩	垡	塾	墼	壅	壑	圩	圬	圪	圳	圹	圮	圯	坜	圻
E	坂	坩	垅	坫	垆	坼	坻	坨	坭	坶	坳	垭	垤	垌	垲	埏
F	垧	垴	垓	垠	埕	埘	埚	埙	埒	垸	埴	埯	埸	埤	埝	

DC	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		堋	堍	埽	埭	堀	堞	堙	塄	堠	塥	塬	墁	墉	墚	墀
B	馨	鼙	懿	艹	艽	艿	芏	芊	芨	芄	芎	芑	芗	芙	芫	芸
C	芾	芰	苈	苊	苣	芘	芷	芮	苋	苌	苁	芩	芴	芡	芪	芟
D	苄	苎	芤	苡	茉	苷	苤	茏	茇	苜	苴	苒	苘	茌	苻	苓
E	茑	茚	茆	茔	茕	苠	苕	茜	荑	荛	荜	茈	莒	茼	茴	茱
F	莛	荞	茯	荏	荇	荃	荟	荀	茗	荠	茭	茺	茳	荦	荥	

DD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		荨	茛	荩	荬	荪	荭	荮	莰	荸	莳	莴	莠	莪	莓	莜
B	莅	荼	莶	莩	荽	莸	荻	莘	莞	莨	莺	莼	菁	萁	菥	菘
C	堇	萘	萋	菝	菽	菖	萜	萸	萑	萆	菔	菟	萏	萃	菸	菹
D	菪	菅	菀	萦	菰	菡	葜	葑	葚	葙	葳	蒇	蒈	葺	蒉	葸
E	萼	葆	葩	葶	蒌	蒎	萱	葭	蓁	蓍	蓐	蓦	蒽	蓓	蓊	蒿
F	蒺	蓠	蒡	蒹	蒴	蒗	蓥	蓣	蔌	甍	蔸	蓰	蔹	蔟	蔺	

DE	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		蕖	蔻	蓿	蓼	蕙	蕈	蕨	蕤	蕞	蕺	瞢	蕃	蕲	蕻	薤
B	薨	薇	薏	蕹	薮	薜	薅	薹	薷	薰	藓	藁	藜	藿	蘧	蘅
C	蘩	蘖	蘼	廾	弈	夼	奁	耷	奕	奚	奘	匏	尢	尥	尬	尴
D	扌	扪	抟	抻	拊	拚	拗	拮	挢	拶	挹	捋	捃	掭	揶	捱
E	捺	掎	掴	捭	掬	掊	捩	掮	掼	揲	揸	揠	揿	揄	揞	揎
F	摒	揆	掾	摅	摁	搋	搛	搠	搌	搦	搡	摞	撄	摭	撖	

DF	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		摺	撷	撸	撙	撺	擀	擐	擗	擤	擢	攉	攥	攮	弋	忒
B	甙	弑	卟	叱	叽	叩	叨	叻	吒	吖	吆	呋	呒	呓	呔	呖
C	呃	吡	呗	呙	吣	吲	咂	咔	呷	呱	呤	咚	咛	咄	呶	呦
D	咝	哐	咭	哂	咴	哒	咧	咦	哓	哔	呲	咣	哕	咻	咿	哌
E	哙	哚	哜	咩	咪	咤	哝	哏	哞	唛	哧	唠	哽	唔	哳	唢
F	唣	唏	唑	唧	唪	啧	喏	喵	啉	啭	啁	啕	唿	啐	唼	

E0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		唷	啖	啵	啶	啷	唳	唰	啜	喋	嗒	喃	喱	喹	喈	喁
B	喟	啾	嗖	喑	啻	嗟	喽	喾	喔	喙	嗪	嗷	嗉	嘟	嗑	嗫
C	嗬	嗔	嗦	嗝	嗄	嗯	嗥	嗲	嗳	嗌	嗍	嗨	嗵	嗤	辔	嘞
D	嘈	嘌	嘁	嘤	嘣	嗾	嘀	嘧	嘭	噘	嘹	噗	嘬	噍	噢	噙
E	噜	噌	噔	嚆	噤	噱	噫	噻	噼	嚅	嚓	嚯	囔	囗	囝	囡
F	囵	囫	囹	囿	圄	圊	圉	圜	帏	帙	帔	帑	帱	帻	帼	

E1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		帷	幄	幔	幛	幞	幡	岌	屺	岍	岐	岖	岈	岘	岙	岑
B	岚	岜	岵	岢	岽	岬	岫	岱	岣	峁	岷	峄	峒	峤	峋	峥
C	崂	崃	崧	崦	崮	崤	崞	崆	崛	嵘	崾	崴	崽	嵬	嵛	嵯
D	嵝	嵫	嵋	嵊	嵩	嵴	嶂	嶙	嶝	豳	嶷	巅	彳	彷	徂	徇
E	徉	後	徕	徙	徜	徨	徭	徵	徼	衢	彡	犭	犰	犴	犷	犸
F	狃	狁	狎	狍	狒	狨	狯	狩	狲	狴	狷	猁	狳	猃	狺	

E2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		狻	猗	猓	猡	猊	猞	猝	猕	猢	猹	猥	猬	猸	猱	獐
B	獍	獗	獠	獬	獯	獾	舛	夥	飧	夤	夂	饣	饧	饨	饩	饪
C	饫	饬	饴	饷	饽	馀	馄	馇	馊	馍	馐	馑	馓	馔	馕	庀
D	庑	庋	庖	庥	庠	庹	庵	庾	庳	赓	廒	廑	廛	廨	廪	膺
E	忄	忉	忖	忏	怃	忮	怄	忡	忤	忾	怅	怆	忪	忭	忸	怙
F	怵	怦	怛	怏	怍	怩	怫	怊	怿	怡	恸	恹	恻	恺	恂	

E3	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		恪	恽	悖	悚	悭	悝	悃	悒	悌	悛	惬	悻	悱	惝	惘
B	惆	惚	悴	愠	愦	愕	愣	惴	愀	愎	愫	慊	慵	憬	憔	憧
C	憷	懔	懵	忝	隳	闩	闫	闱	闳	闵	闶	闼	闾	阃	阄	阆
D	阈	阊	阋	阌	阍	阏	阒	阕	阖	阗	阙	阚	丬	爿	戕	氵
E	汔	汜	汊	沣	沅	沐	沔	沌	汨	汩	汴	汶	沆	沩	泐	泔
F	沭	泷	泸	泱	泗	沲	泠	泖	泺	泫	泮	沱	泓	泯	泾	

E4	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		洹	洧	洌	浃	浈	洇	洄	洙	洎	洫	浍	洮	洵	洚	浏
B	浒	浔	洳	涑	浯	涞	涠	浞	涓	涔	浜	浠	浼	浣	渚	淇
C	淅	淞	渎	涿	淠	渑	淦	淝	淙	渖	涫	渌	涮	渫	湮	湎
D	湫	溲	湟	溆	湓	湔	渲	渥	湄	滟	溱	溘	滠	漭	滢	溥
E	溧	溽	溻	溷	滗	溴	滏	溏	滂	溟	潢	潆	潇	漤	漕	滹
F	漯	漶	潋	潴	漪	漉	漩	澉	澍	澌	潸	潲	潼	潺	濑	

E5	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		潍	澧	澹	澶	濂	濡	濮	濞	濠	濯	瀚	瀣	瀛	瀹	瀵
B	灏	灞	宀	宄	宕	宓	宥	宸	甯	骞	搴	寤	寮	褰	寰	蹇
C	謇	辶	迓	迕	迥	迮	迤	迩	迦	迳	迨	逅	逄	逋	逦	逑
D	逍	逖	逡	逵	逶	逭	逯	遄	遑	遒	遐	遨	遘	遢	遛	暹
E	遴	遽	邂	邈	邃	邋	彐	彗	彖	彘	尻	咫	屐	屙	孱	屣
F	屦	羼	弪	弩	弭	艴	弼	鬻	屮	妁	妃	妍	妩	妪	妣	

E6	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		妗	姊	妫	妞	妤	姒	妲	妯	姗	妾	娅	娆	姝	娈	姣
B	姘	姹	娌	娉	娲	娴	娑	娣	娓	婀	婧	婊	婕	娼	婢	婵
C	胬	媪	媛	婷	婺	媾	嫫	媲	嫒	嫔	媸	嫠	嫣	嫱	嫖	嫦
D	嫘	嫜	嬉	嬗	嬖	嬲	嬷	孀	尕	尜	孚	孥	孳	孑	孓	孢
E	驵	驷	驸	驺	驿	驽	骀	骁	骅	骈	骊	骐	骒	骓	骖	骘
F	骛	骜	骝	骟	骠	骢	骣	骥	骧	纟	纡	纣	纥	纨	纩	

E7	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		纭	纰	纾	绀	绁	绂	绉	绋	绌	绐	绔	绗	绛	绠	绡
B	绨	绫	绮	绯	绱	绲	缍	绶	绺	绻	绾	缁	缂	缃	缇	缈
C	缋	缌	缏	缑	缒	缗	缙	缜	缛	缟	缡	缢	缣	缤	缥	缦
D	缧	缪	缫	缬	缭	缯	缰	缱	缲	缳	缵	幺	畿	巛	甾	邕
E	玎	玑	玮	玢	玟	珏	珂	珑	玷	玳	珀	珉	珈	珥	珙	顼
F	琊	珩	珧	珞	玺	珲	琏	琪	瑛	琦	琥	琨	琰	琮	琬	

E8	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		琛	琚	瑁	瑜	瑗	瑕	瑙	瑷	瑭	瑾	璜	璎	璀	璁	璇
B	璋	璞	璨	璩	璐	璧	瓒	璺	韪	韫	韬	杌	杓	杞	杈	杩
C	枥	枇	杪	杳	枘	枧	杵	枨	枞	枭	枋	杷	杼	柰	栉	柘
D	栊	柩	枰	栌	柙	枵	柚	枳	柝	栀	柃	枸	柢	栎	柁	柽
E	栲	栳	桠	桡	桎	桢	桄	桤	梃	栝	桕	桦	桁	桧	桀	栾
F	桊	桉	栩	梵	梏	桴	桷	梓	桫	棂	楮	棼	椟	椠	棹	

E9	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		椤	椪	椋	椁	楗	棣	椐	楱	椹	楠	楂	楝	榄	楫	榀
B	榘	楸	椴	槌	榇	榈	槎	榉	楦	楣	楹	榛	榧	榻	榫	榭
C	槔	榱	槁	槊	槟	榕	槠	榍	槿	樯	槭	樗	樘	橥	槲	橄
D	樾	檠	橐	橛	樵	檎	橹	樽	樨	橘	橼	檑	檐	檩	檗	檫
E	猷	獒	殁	殂	殇	殄	殒	殓	殍	殚	殛	殡	殪	轫	轭	轱
F	轲	轳	轵	轶	轸	轷	轹	轺	轼	轾	辁	辂	辄	辇	辋	

EA	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		辍	辎	辏	辘	辚	軎	戋	戗	戛	戟	戢	戡	戥	戤	戬
B	臧	瓯	瓴	瓿	甏	甑	甓	攴	旮	旯	旰	昊	昙	杲	昃	昕
C	昀	炅	曷	昝	昴	昱	昶	昵	耆	晟	晔	晁	晏	晖	晡	晗
D	晷	暄	暌	暧	暝	暾	曛	曜	曦	曩	贲	贳	贶	贻	贽	赀
E	赅	赆	赈	赉	赇	赍	赕	赙	觇	觊	觋	觌	觎	觏	觐	觑
F	牮	犟	牝	牦	牯	牾	牿	犄	犋	犍	犏	犒	挈	挲	掰	

EB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		搿	擘	耄	毪	毳	毽	毵	毹	氅	氇	氆	氍	氕	氘	氙
B	氚	氡	氩	氤	氪	氲	攵	敕	敫	牍	牒	牖	爰	虢	刖	肟
C	肜	肓	肼	朊	肽	肱	肫	肭	肴	肷	胧	胨	胩	胪	胛	胂
D	胄	胙	胍	胗	朐	胝	胫	胱	胴	胭	脍	脎	胲	胼	朕	脒
E	豚	脶	脞	脬	脘	脲	腈	腌	腓	腴	腙	腚	腱	腠	腩	腼
F	腽	腭	腧	塍	媵	膈	膂	膑	滕	膣	膪	臌	朦	臊	膻	

EC	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		臁	膦	欤	欷	欹	歃	歆	歙	飑	飒	飓	飕	飙	飚	殳
B	彀	毂	觳	斐	齑	斓	於	旆	旄	旃	旌	旎	旒	旖	炀	炜
C	炖	炝	炻	烀	炷	炫	炱	烨	烊	焐	焓	焖	焯	焱	煳	煜
D	煨	煅	煲	煊	煸	煺	熘	熳	熵	熨	熠	燠	燔	燧	燹	爝
E	爨	灬	焘	煦	熹	戾	戽	扃	扈	扉	礻	祀	祆	祉	祛	祜
F	祓	祚	祢	祗	祠	祯	祧	祺	禅	禊	禚	禧	禳	忑	忐	

ED	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		怼	恝	恚	恧	恁	恙	恣	悫	愆	愍	慝	憩	憝	懋	懑
B	戆	肀	聿	沓	泶	淼	矶	矸	砀	砉	砗	砘	砑	斫	砭	砜
C	砝	砹	砺	砻	砟	砼	砥	砬	砣	砩	硎	硭	硖	硗	砦	硐
D	硇	硌	硪	碛	碓	碚	碇	碜	碡	碣	碲	碹	碥	磔	磙	磉
E	磬	磲	礅	磴	礓	礤	礞	礴	龛	黹	黻	黼	盱	眄	眍	盹
F	眇	眈	眚	眢	眙	眭	眦	眵	眸	睐	睑	睇	睃	睚	睨	

EE	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		睢	睥	睿	瞍	睽	瞀	瞌	瞑	瞟	瞠	瞰	瞵	瞽	町	畀
B	畎	畋	畈	畛	畲	畹	疃	罘	罡	罟	詈	罨	罴	罱	罹	羁
C	罾	盍	盥	蠲	钅	钆	钇	钋	钊	钌	钍	钏	钐	钔	钗	钕
D	钚	钛	钜	钣	钤	钫	钪	钭	钬	钯	钰	钲	钴	钶	钷	钸
E	钹	钺	钼	钽	钿	铄	铈	铉	铊	铋	铌	铍	铎	铐	铑	铒
F	铕	铖	铗	铙	铘	铛	铞	铟	铠	铢	铤	铥	铧	铨	铪	

EF	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		铩	铫	铮	铯	铳	铴	铵	铷	铹	铼	铽	铿	锃	锂	锆
B	锇	锉	锊	锍	锎	锏	锒	锓	锔	锕	锖	锘	锛	锝	锞	锟
C	锢	锪	锫	锩	锬	锱	锲	锴	锶	锷	锸	锼	锾	锿	镂	锵
D	镄	镅	镆	镉	镌	镎	镏	镒	镓	镔	镖	镗	镘	镙	镛	镞
E	镟	镝	镡	镢	镤	镥	镦	镧	镨	镩	镪	镫	镬	镯	镱	镲
F	镳	锺	矧	矬	雉	秕	秭	秣	秫	稆	嵇	稃	稂	稞	稔	

F0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		稹	稷	穑	黏	馥	穰	皈	皎	皓	皙	皤	瓞	瓠	甬	鸠
B	鸢	鸨	鸩	鸪	鸫	鸬	鸲	鸱	鸶	鸸	鸷	鸹	鸺	鸾	鹁	鹂
C	鹄	鹆	鹇	鹈	鹉	鹋	鹌	鹎	鹑	鹕	鹗	鹚	鹛	鹜	鹞	鹣
D	鹦	鹧	鹨	鹩	鹪	鹫	鹬	鹱	鹭	鹳	疒	疔	疖	疠	疝	疬
E	疣	疳	疴	疸	痄	疱	疰	痃	痂	痖	痍	痣	痨	痦	痤	痫
F	痧	瘃	痱	痼	痿	瘐	瘀	瘅	瘌	瘗	瘊	瘥	瘘	瘕	瘙	

F1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		瘛	瘼	瘢	瘠	癀	瘭	瘰	瘿	瘵	癃	瘾	瘳	癍	癞	癔
B	癜	癖	癫	癯	翊	竦	穸	穹	窀	窆	窈	窕	窦	窠	窬	窨
C	窭	窳	衤	衩	衲	衽	衿	袂	袢	裆	袷	袼	裉	裢	裎	裣
D	裥	裱	褚	裼	裨	裾	裰	褡	褙	褓	褛	褊	褴	褫	褶	襁
E	襦	襻	疋	胥	皲	皴	矜	耒	耔	耖	耜	耠	耢	耥	耦	耧
F	耩	耨	耱	耋	耵	聃	聆	聍	聒	聩	聱	覃	顸	颀	颃	

F2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		颉	颌	颍	颏	颔	颚	颛	颞	颟	颡	颢	颥	颦	虍	虔
B	虬	虮	虿	虺	虼	虻	蚨	蚍	蚋	蚬	蚝	蚧	蚣	蚪	蚓	蚩
C	蚶	蛄	蚵	蛎	蚰	蚺	蚱	蚯	蛉	蛏	蚴	蛩	蛱	蛲	蛭	蛳
D	蛐	蜓	蛞	蛴	蛟	蛘	蛑	蜃	蜇	蛸	蜈	蜊	蜍	蜉	蜣	蜻
E	蜞	蜥	蜮	蜚	蜾	蝈	蜴	蜱	蜩	蜷	蜿	螂	蜢	蝽	蝾	蝻
F	蝠	蝰	蝌	蝮	螋	蝓	蝣	蝼	蝤	蝙	蝥	螓	螯	螨	蟒	

F3	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		蟆	螈	螅	螭	螗	螃	螫	蟥	螬	螵	螳	蟋	蟓	螽	蟑
B	蟀	蟊	蟛	蟪	蟠	蟮	蠖	蠓	蟾	蠊	蠛	蠡	蠹	蠼	缶	罂
C	罄	罅	舐	竺	竽	笈	笃	笄	笕	笊	笫	笏	筇	笸	笪	笙
D	笮	笱	笠	笥	笤	笳	笾	笞	筘	筚	筅	筵	筌	筝	筠	筮
E	筻	筢	筲	筱	箐	箦	箧	箸	箬	箝	箨	箅	箪	箜	箢	箫
F	箴	篑	篁	篌	篝	篚	篥	篦	篪	簌	篾	篼	簏	簖	簋	

F4	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		簟	簪	簦	簸	籁	籀	臾	舁	舂	舄	臬	衄	舡	舢	舣
B	舭	舯	舨	舫	舸	舻	舳	舴	舾	艄	艉	艋	艏	艚	艟	艨
C	衾	袅	袈	裘	裟	襞	羝	羟	羧	羯	羰	羲	籼	敉	粑	粝
D	粜	粞	粢	粲	粼	粽	糁	糇	糌	糍	糈	糅	糗	糨	艮	暨
E	羿	翎	翕	翥	翡	翦	翩	翮	翳	糸	絷	綦	綮	繇	纛	麸
F	麴	赳	趄	趔	趑	趱	赧	赭	豇	豉	酊	酐	酎	酏	酤	

F5	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		酢	酡	酰	酩	酯	酽	酾	酲	酴	酹	醌	醅	醐	醍	醑
B	醢	醣	醪	醭	醮	醯	醵	醴	醺	豕	鹾	趸	跫	踅	蹙	蹩
C	趵	趿	趼	趺	跄	跖	跗	跚	跞	跎	跏	跛	跆	跬	跷	跸
D	跣	跹	跻	跤	踉	跽	踔	踝	踟	踬	踮	踣	踯	踺	蹀	踹
E	踵	踽	踱	蹉	蹁	蹂	蹑	蹒	蹊	蹰	蹶	蹼	蹯	蹴	躅	躏
F	躔	躐	躜	躞	豸	貂	貊	貅	貘	貔	斛	觖	觞	觚	觜	

F6	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		觥	觫	觯	訾	謦	靓	雩	雳	雯	霆	霁	霈	霏	霎	霪
B	霭	霰	霾	龀	龃	龅	龆	龇	龈	龉	龊	龌	黾	鼋	鼍	隹
C	隼	隽	雎	雒	瞿	雠	銎	銮	鋈	錾	鍪	鏊	鎏	鐾	鑫	鱿
D	鲂	鲅	鲆	鲇	鲈	稣	鲋	鲎	鲐	鲑	鲒	鲔	鲕	鲚	鲛	鲞
E	鲟	鲠	鲡	鲢	鲣	鲥	鲦	鲧	鲨	鲩	鲫	鲭	鲮	鲰	鲱	鲲
F	鲳	鲴	鲵	鲶	鲷	鲺	鲻	鲼	鲽	鳄	鳅	鳆	鳇	鳊	鳋	

F7	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		鳌	鳍	鳎	鳏	鳐	鳓	鳔	鳕	鳗	鳘	鳙	鳜	鳝	鳟	鳢
B	靼	鞅	鞑	鞒	鞔	鞯	鞫	鞣	鞲	鞴	骱	骰	骷	鹘	骶	骺
C	骼	髁	髀	髅	髂	髋	髌	髑	魅	魃	魇	魉	魈	魍	魑	飨
D	餍	餮	饕	饔	髟	髡	髦	髯	髫	髻	髭	髹	鬈	鬏	鬓	鬟
E	鬣	麽	麾	縻	麂	麇	麈	麋	麒	鏖	麝	麟	黛	黜	黝	黠
F	黟	黢	黩	黧	黥	黪	黯	鼢	鼬	鼯	鼹	鼷	鼽	鼾	齄	

4. 示范程序

```
;*******************************************************
RS              EQU  P1.1            ;L:SELECT COMMAND,H:SELECT DDRAM
WR              EQU  P3.6            ;L EFFECTIVE
RD              EQU  P3.7            ;L EFFECTIVE
CS1             EQU  P2.7            ;L EFFECITVE
BUSY            EQU  P2.0
INT             EQU  P3.2
RST             EQU  P1.4
;DB0--DB7       EQU   P0.0---P0.7
REGNAME          EQU      30H
REGDATA          EQU      31H
REGDATA1         EQU      32H
LCM_X            EQU      33H
```

```
LCM_Y           EQU     34H
LCM_DATA1       EQU     35H
LCM_DATA2       EQU     36H
COUNT1          EQU     37H
COUNT2          EQU     38H
COUNT3          EQU     39H
DP_TEMP1        EQU     3AH
DP_TEMP2        EQU     3BH
;*************************************************
        ORG     0000H
        AJMP    MAIN
        ORG     0100H
MAIN:   NOP
        MOV     SP,#5FH
        LCALL   DELAY2          ;WAIT 120MS POWER-ON
        LCALL   LCM_INIT
        LCALL   LCM_CLR
;*************************************************
LGS1:   MOV     REGNAME,#10H
        MOV     REGDATA,#00H    ;NORMAL TEXT,CURSOR OFF,AUTO-INC
        LCALL   REG_WRITE
        MOV     REGNAME,#00H
        MOV     REGDATA,#04H    ;   GRAPHIC MODE
        LCALL   REG_WRITE
        MOV     LCM_X,#00H      ;   WRITE 3 BOLD
        MOV     LCM_Y,#00H
        LCALL   WR_ZB
        MOV     A,#0B0H         ;WRITE MEMORY DATA
        LCALL   REG_WR
        MOV     COUNT1,#03H
LGS1_LPR1:  MOV     COUNT2,#28H
LGS1_LPR2:  MOV     A,#0FFH
        LCALL   DDR_WRITE
        DJNZ    COUNT2,LGS1_LPR2
        DJNZ    COUNT1,LGS1_LPR1
        MOV     COUNT1,#0EAH
LGS1_LPR3:  MOV     A,#0E0H
            LCALL   DDR_WRITE
```

```
                MOV       COUNT2,#26H
LGS1_LPR4:      MOV       A,#00H
                LCALL DDR_WRITE
                DJNZ      COUNT2,LGS1_LPR4
                MOV       A,#07H
                LCALL     DDR_WRITE
                DJNZ      COUNT1,LGS1_LPR3
                MOV       COUNT1,#03H
LGS1_LPR5:      MOV       COUNT2,#28H
LGS1_LPR6:      MOV       A,#0FFH
                LCALL     DDR_WRITE
                DJNZ      COUNT2,LGS1_LPR6
                DJNZ      COUNT1,LGS1_LPR5
                LCALL   DELAY1
;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
LGS2:           MOV     REGNAME,#00H
                MOV     REGDATA,#04H        ;GRAPHIC MODE
                LCALL   REG_WRITE
                MOV     LCM_DATA1,#0FFH
                MOV     LCM_DATA2,#00H
                LCALL   LAT_DISP
                LCALL   DELAY1
                MOV       LCM_DATA1,#00H
                MOV       LCM_DATA2,#0FFH
                LCALL     LAT_DISP
                LCALL     DELAY1
                MOV       LCM_DATA1,#0AAH
                MOV       LCM_DATA2,#0AAH
                LCALL     LAT_DISP
                LCALL     DELAY1
                MOV       LCM_DATA1,#55H
                MOV       LCM_DATA2,#055H
                LCALL     LAT_DISP
                LCALL     DELAY1
                MOV       LCM_DATA1,#0AAH
                MOV       LCM_DATA2,#55H
                LCALL     LAT_DISP
                LCALL     DELAY1
```

```
                MOV     LCM_DATA1,#55H
                MOV     LCM_DATA2,#0AAH
                LCALL   LAT_DISP
                LCALL   DELAY1
;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
 LGS3:
                MOV     REGNAME,#00H
                MOV     REGDATA,#0CH      ;   TEXT MODE
                LCALL   REG_WRITE
                MOV     DPTR,#TAB1
                LCALL   CHRT_DISP
                LCALL     DELAY1
;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
LGS4:           MOV     REGNAME,#10H
                MOV     REGDATA,#10H         ;BOLD TEXT,CURSOR OFF, AUTO-INC
                LCALL   REG_WRITE
                MOV     REGNAME,#00H
                MOV     REGDATA,#0CH        ;TEXT MODE
                LCALL   REG_WRITE
                MOV     DPTR,#TAB1
                LCALL   CHRT_DISP
                LCALL   DELAY1
                LCALL     DISCOM              ;240X320 MODE
                MOV       REGNAME,#10H
                MOV       REGDATA,#08H
                LCALL     REG_WRITE
                MOV     DPTR,#TAB1
                LCALL   CHRT_DISP
                LCALL   DELAY1
                MOV     LCM_X,#00H
                MOV     LCM_Y,#00H
                LCALL   WR_ZB
                MOV       A,#0B0H             ;WRITE MEMORY DATA
                LCALL     REG_WR
                MOV       R0,#0FH
   LGS4_R2:
                MOV       R1,#28H
   LGS4_R1:
```

```
                MOV     A,#00H
                LCALL   DDR_WRITE
                DJNZ    R1,LGS4_R1
                DJNZ    R0,LGS4_R2
                LCALL   COMOK                   ;ENLARGE TEXT MODE
                MOV     REGNAME,#10H
                MOV     REGDATA,#10H
                LCALL   REG_WRITE
                MOV     DPTR,#TAB3
                LCALL   HCHRT_DISP
                LCALL   DELAY1
                MOV     REGNAME,#0F1H       ;ENLARGE TEXT BY TWICES
                MOV     REGDATA,#050H
                LCALL   REG_WRITE
                MOV     DPTR,#TAB3
                LCALL   HCHRT_DISP
                LCALL   DELAY1
                MOV     REGNAME,#0F1H       ;ENLARGE TEXT BY THREE TIMES
                MOV     REGDATA,#0F0H
                LCALL   REG_WRITE
                MOV     DPTR,#TAB3
                LCALL   HCHRT_DISP
                LCALL   DELAY1
;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
LGS5:           MOV     REGNAME,#00H
                MOV     REGDATA,#0DH        ;TEXT MODE,REVERSE DISPLAY
                LCALL   REG_WRITE
                LCALL   DELAY1
;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
                MOV     REGNAME,#00H
                MOV     REGDATA,#04H        ;GRAPHIC MODE,NOMAL DISPLAY
                LCALL   REG_WRITE
LGS6:           MOV     DPTR,#TAB2
                MOV     DP_TEMP1,DPH
                MOV     DP_TEMP2,DPL
                LCALL PHO_DISP
                LCALL DELAY1
```

```
                LJMP    $
;*********************SUBROUTINE******************************
LCM_INIT:       MOV     COUNT1,#00H             ;INITIAL RA8806
                MOV     COUNT2,#20H
LCM_INIT1:      MOV     DPTR,#INITTAB1
                MOV     A,COUNT1
                MOVC    A,@A+DPTR
                MOV     REGNAME,A
                MOV     DPTR,#INITTAB2
                MOV     A,COUNT1
                MOVC    A,@A+DPTR
                MOV     REGDATA,A
                LCALL   REG_WRITE
                INC     COUNT1
                DJNZ    COUNT2,LCM_INIT1
                RET
;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
LCM_CLR:        MOV     REGNAME,#0E0H           ;CLEAR SCREEN
        MOV     REGDATA,#00H
        LCALL   REG_WRITE
        MOV     REGNAME,#0F0H
        LCALL   REG_READ
        MOV     A,REGDATA1
        ORLA,#08H
        MOV     REGDATA,A
        MOV     REGNAME,#0F0H
        LCALL   REG_WRITE

        MOV     REGNAME,#00H
        MOV     REGDATA,#04H
        LCALL   REG_WRITE
        MOV     LCM_X,#00H
        MOV     LCM_Y,#00H
        LCALL   WR_ZB
        MOV      A,#0B0H              ;WRITE MEMORY DATA
        LCALL    REG_WR
        MOV     A,#00H
        LCALL   DDR_WRITE
```

```
        RET
    ;*************************************************
CHRT_DISP:  MOV   LCM_X,#00H          ;DISPLAY CHINESE FONT OR TEXT ALL OVER
            MOV   LCM_Y,#00H
        LCALL WR_ZB
        MOV    A,#0B0H            ;WRITE MEMORY DATA
        LCALL  REG_WR
        MOV   COUNT1,#0FH
CHRT_DISP1:  MOV   COUNT2,#28H
CHRT_DISP2:  CLRA
        MOVC  A,@A+DPTR
        LCALL DDR_WRITE
        INC DPTR
        DJNZ  COUNT2,CHRT_DISP2
        DJNZ  COUNT1,CHRT_DISP1
        RET
    ;*************************************************
    HCHRT_DISP:  MOV   LCM_X,#00H          ;DISPLAY CHINESE FONT OR TEXT ALL
    OVER THE SCREEN
                 MOV   LCM_Y,#00H
                 LCALL WR_ZB
                 MOV    A,#0B0H            ;WRITE MEMORY DATA
                 LCALL  REG_WR
                 MOV   COUNT1,#01H
    HCHRT_DISP1: MOV   COUNT2,#01EH
    HCHRT_DISP2: CLRA
                 MOVC  A,@A+DPTR
                 LCALL DDR_WRITE
                 INC DPTR
                 DJNZ  COUNT2,HCHRT_DISP2
                 DJNZ  COUNT1,HCHRT_DISP1
                 RET
    ;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    PHO_DISP:    MOV   LCM_X,#00H       ;DISPLAY A GRAPHIC ALL OVER THSCREEN
                 MOV   LCM_Y,#00H
                 LCALL WR_ZB
                 MOV    A,#0B0H            ;WRITE MEMORY DATA
                 LCALL  REG_WR
```

```
                MOV     COUNT3,#03H
PHO_DISP0:      MOV     COUNT1,#50H
PHO_DISP1:      MOV     COUNT2,#28H
PHO_DISP2:      CLR A
                MOVC    A,@A+DPTR
                LCALL   DDR_WRITE
                INC DPTR
                DJNZ    COUNT2,PHO_DISP2
                DJNZ    COUNT1,PHO_DISP1
                MOV     DPH,DP_TEMP1
                MOV     DPL,DP_TEMP2
                DJNZ    COUNT3,PHO_DISP0
                RET
;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
LAT_DISP:       MOV LCM_X,#00H          ;DIAPLAY GRAPHIC DOTS
                MOV LCM_Y,#00H
                LCALL   WR_ZB
                MOV     A,#0B0H         ;WRITE MEMORY DATA
                LCALL   REG_WR
                MOV     COUNT1,#78H
LAT_DISP1:      MOV     COUNT2,#28H
LAT_DISP2:      MOV     A,LCM_DATA1
                LCALL   DDR_WRITE
                DJNZ    COUNT2,LAT_DISP2
                MOV     COUNT2,#28H
LAT_DISP3:      MOV     A,LCM_DATA2
                LCALL   DDR_WRITE
                DJNZ    COUNT2,LAT_DISP3
                DJNZ    COUNT1,LAT_DISP1
                RET
;**************************************************
WR_ZB:          MOV     REGNAME,#60H    ;SET CUSOR POSITION
                MOV     REGDATA,LCM_X
                LCALL   REG_WRITE
                MOV     REGNAME,#70H
                MOV     REGDATA,LCM_Y
                LCALL   REG_WRITE
                RET
```

```
;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
REG_WRITE:      MOV     A,REGNAME               ;WRITE DATA TO COMMAND REGISTER
                LCALL   REG_WR
                MOV     A,REGDATA
                LCALL   DDR_WRITE
                RET
;**************************************************
REG_WR:         MOV     P0,A
                CLRCS1
                SETB    RD
                SETB    RS
                CLRWR
                NOP
                NOP
               SETBWR
               SETBCS1
               RET
;*******************************************************
DDR_WRITE:      JNB BUSY,$              ;WRITE DATA TO DDRAM
                MOV     P0,A
                CLRCS1
                SETB    RD
                CLRRS
                CLRWR
                NOP
                NOP
                SETB    WR
                SETB    CS1
                RET
;***********************************************************
REG_READ: MOV     A,REGNAME               ;READ COMMAND REGISTER
          LCALL   REG_WR
          MOV     P0,#0FFH
          CLRCS1
          SETB    WR
          SETB    RS
          CLRRD
          NOP
```

```
        MOV     A,P0
        SETB    RD
        SETB    CS1
        MOV     REGDATA1,A
        RET
;*************************************************
DISCOM:         MOV  REGNAME,#01H
        LCALL   REG_READ
        MOV     A,REGDATA1
        ORL     A,#05H
        MOV     REGDATA,A
        MOV     REGNAME,#01H
        LCALL   REG_WRITE
                RET
;*************************************************
COMOK:  MOV     REGNAME,#01H
        LCALL   REG_READ
        MOV  A,REGDATA1
        ANL  A,#0FEH
        ORL     A,#04H
        MOV  REGDATA,A
        MOV  REGNAME,#01H
        LCALL   REG_WRITE
                RET
;*************************************************
DISSEG:  MOVREGNAME,#01H
         LCALL  REG_READ
         MOV A,REGDATA1
         ORL  A,#06H
         MOV REGDATA,A
         MOV REGNAME,#01H
         LCALL  REG_WRITE
                RET
;*************************************************
SEGOK:  MOV     REGNAME,#01H
        LCALL   REG_READ
        MOV     A,REGDATA1
        ANL     A,#0FDH
```

```
            ORL       A,#04H
            MOV       REGDATA,A
            MOV       REGNAME,#01H
            LCALL     REG_WRITE
                      RET
  ;*****************************************************
DELAY1:  MOV       R5,#16H
  DEL11:  MOV       R6,#0CFH
  DEL12:  MOV       R7,#0AFH
  DEL13:  DJNZ      R7,DEL13
            DJNZ      R6,DEL12
            DJNZ      R5,DEL11
            RET

DELAY2:  MOV       R6,#0EAH
  DEL21:  MOV       R7,#0FFH
  DEL22:  DJNZ      R7,DEL22
            DJNZ      R6,DEL21
            RET
  ;*****************************************************
  INITTAB1:
  DB 000H,001H,003H,00FH,010H,011H,012H,021H,031H,040H,050H,020H,030H,060H,061H,062H
  DB 070H,071H,072H,080H,090H,0A0H,0A1H,0A2H,0A2H,0A3H,0A4H
  DB 0D0H,0D1H,0E0H,0F0H,0F1H
  INITTAB2:
  DB 000H,004H,000H,000H,000H,000H,011H,027H,0EFH,000H,000H,027H,0EFH,000H,000H,000H
  DB 000H,000H,000H,020H,000H,000H,000H,000H,000H,000H,000H
  DB 000H,000H,000H,000H,000H

  TAB1:
  DB "            有一种爱叫做放手                    "
  DB "如果两个人的天堂象是温馨的墙囚禁你的梦想"
  DB "幸福是否象是一扇铁窗，候鸟失去了南方      "
  DB "如果你对天空向往渴望一双翅膀放手让你飞翔"
  DB "你的羽翼不该伴随玫瑰，听从凋谢的时光      "
  DB "浪漫如果变成了牵绊，我愿为你选择回到孤单"
  DB "缠绵如果变成了锁链，抛开诺言                "
  DB "有一种爱叫做放手，为爱放弃天长地久.       "
```

```
DB "我们相守若让你付出所有，让真爱带我走        "
DB "有一种爱叫做放手，为爱结束天长地久.         "
DB "我的离去若让你拥有所有让真爱带我走说分手"
DB "为了你，失去你，狠心扮演伤害你，为了你，"
DB "离开你，永远不分的离去.                     "
DB "有一种爱叫做放手，为爱结束天长地久.         "
DB "我的离去若让你拥有所有让真爱带我走说分手"

TAB3:
DB "                                          "

TAB2:
;--  调入了一幅图像：F:\梁\画图\HOCO32080.bmp  --
;--  宽度 x 高度=320x80  --
DB
0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB
0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB
0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0C0H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,003H,0C0H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,003H
DB  080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H
DB  080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H
DB  080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H
DB 080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
```

```
DB 000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H
DB 080H,03FH,0F8H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB 000H,000H,000H,001H,080H,03FH,0F8H,000H,000H,000H,000H,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,001H,080H,07FH,0FCH,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H,080H,07FH,0FCH,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H
DB 080H,0FFH,0FFH,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H,000H,080H,000H
DB 00FH,0FFH,000H,001H,080H,0FFH,0FFH,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB 001H,000H,080H,000H,00FH,0FFH,000H,001H,083H,0FFH,0FFH,080H,000H,000H,000H,000H
DB 000H,000H,000H,00CH,043H,001H,080H,000H,07FH,0FFH,0F0H,001H,083H,0FFH,0FFH,080H
DB 000H,000H,000H,000H,000H,000H,000H,00CH,043H,001H,080H,000H,07FH,0FFH,0F0H,001H
DB 087H,0FFH,0FFH,080H,000H,000H,000H,000H,000H,000H,000H,019H,0FFH,007H,000H,007H
DB 0FFH,0FFH,0FEH,001H,087H,0FFH,0FFH,080H,000H,000H,000H,000H,000H,000H,000H,019H
DB 0FFH,007H,000H,007H,0FFH,0FFH,0FEH,001H,08FH,0FFH,0FFH,0E0H,000H,000H,000H,000H
DB 000H,000H,001H,03FH,0FCH,01EH,000H,01FH,0FFH,0FFH,0FFH,081H,08FH,0FFH,0FFH,0E0H
DB 000H,000H,000H,000H,000H,000H,001H,03FH,0FCH,01EH,000H,01FH,0FFH,0FFH,0FFH,081H
DB 09FH,0FFH,0FFH,0E0H,000H,000H,000H,000H,000H,000H,001H,0BFH,0FFH,0FCH,000H,07FH
DB 0FCH,000H,07FH,0C1H,09FH,0FFH,0FFH,0E0H,000H,000H,000H,000H,000H,000H,001H,0BFH
DB 0FFH,0FCH,000H,07FH,0FCH,000H,07FH,0C1H,097H,0FFH,0FFH,0F0H,000H,000H,000H,000H
DB 000H,000H,001H,0FFH,0FFH,0F0H,000H,0FFH,0C0H,000H,00FH,0E1H,097H,0FFH,0FFH,0F0H
DB 000H,000H,000H,000H,000H,000H,001H,0FFH,0FFH,0F0H,000H,0FFH,0C0H,000H,00FH,0E1H
DB 09FH,0FFH,0FFH,0F0H,000H,000H,000H,000H,000H,000H,001H,0FFH,0FFH,0E0H,003H,0FFH
DB 0FFH,0FCH,001H,0F1H,09FH,0FFH,0FFH,0F0H,000H,000H,000H,000H,000H,000H,001H,0FFH
DB 0FFH,0E0H,003H,0FFH,0FFH,0FCH,001H,0F1H,097H,0FFH,0FFH,0F8H,000H,000H,000H,000H
DB 000H,000H,001H,0DFH,0FFH,0C0H,007H,0FFH,0FFH,0FFH,080H,0F1H,097H,0FFH,0FFH,0F8H
DB 00H,000H,000H,000H,000H,000H,001H,0DFH,0FFH,0C0H,007H,0FFH,0FFH,0FFH,080H,0F1H
DB 09FH,07FH,0FFH,0FCH,000H,000H,000H,000H,000H,000H,003H,0FFH,0FFH,000H,00FH,0FFH
DB 0FFH,0FFH,0E0H,039H,09FH,07FH,0FFH,0FCH,000H,000H,000H,000H,000H,000H,003H,0FFH
DB 0FFH,000H,00FH,0FFH,0FFH,0FFH,0E0H,039H,08BH,07FH,0FFH,0FCH,000H,000H,000H,000H
DB 000H,000H,007H,0FFH,0F8H,000H,01FH,0FFH,0F0H,003H,0F8H,039H,08BH,07FH,0FFH,0FCH
DB  000H,000H,000H,000H,000H,000H,007H,0FFH,0F8H,000H,01FH,0FFH,0F0H,003H,0F8H,039H
DB 087H,0FFH,0FFH,0FCH,000H,000H,000H,000H,000H,000H,007H,0FFH,0E0H,000H,03FH,0FFH
DB 0FFH,0C0H,07CH,019H,087H,0FFH,0FFH,0FCH,000H,000H,000H,000H,000H,000H,007H,0FFH
DB 0E0H,000H,03FH,0FFH,0FFH,0C0H,07CH,019H,08BH,0FFH,0FFH,0FEH,000H,000H,000H,000H
DB 000H,000H,004H,07FH,0F0H,000H,03FH,0FFH,0FFH,0F8H,01EH,009H,08BH,0FFH,0FFH,0FEH
```

```
DB 000H,000H,000H,000H,000H,000H,004H,07FH,0F0H,000H,03FH,0FFH,0FFH,0F8H,01EH,009H
DB 081H,0F3H,0FFH,0FEH,000H,000H,000H,000H,000H,000H,001H,0FFH,0F8H,000H,07FH,0FFH
DB 0FFH,0FEH,00FH,009H,081H,0F3H,0FFH,0FEH,000H,000H,000H,000H,000H,000H,001H,0FFH
DB 0F8H,000H,07FH,0FFH,0FFH,0FEH,00FH,009H,081H,0F7H,0FFH,0FEH,000H,000H,000H,000H
DB 000H,000H,00FH,0FFH,0FCH,000H,0FFH,0FFH,0FFH,0FFH,087H,009H,081H,0F7H,0FFH,0FEH
DB 000H,000H,000H,000H,000H,000H,00FH,0FFH,0FCH,000H,0FFH,0FFH,0FFH,0FFH,087H,009H
DB 081H,0F7H,0FFH,0FFH,000H,000H,000H,000H,000H,000H,01FH,0FFH,0FEH,000H,0FFH,0FFH
DB 0FFH,0FFH,083H,089H,081H,0F7H,0FFH,0FFH,000H,000H,000H,000H,000H,000H,01FH,0FFH
DB 0FEH,000H,0FFH,0FFH,0FFH,0FFH,083H,089H,081H,0D3H,0FFH,0FFH,0F0H,03FH,0F8H,000H
DB 00H,000H,03FH,0FFH,0FEH,001H,0FFH,0FFH,0FFH,0FFH,0C3H,089H,081H,0D3H,0FFH,0FFH
DB 0F0H,03FH,0F8H,000H,000H,000H,03FH,0FFH,0FEH,001H,0FFH,0FFH,0FFH,0FFH,0C3H,089H
DB 081H,0D3H,0FFH,0FFH,0FFH,0FFH,0FFH,000H,000H,000H,07FH,0FFH,0FEH,001H,0FFH,0FFH
DB 0FFH,0FFH,0E1H,089H,081H,0D3H,0FFH,0FFH,0FFH,0FFH,0FFH,000H,000H,000H,07FH,0FFH
DB 0FEH,001H,0FFH,0FFH,0FFH,0FFH,0E1H,089H,080H,0E3H,0FFH,0FFH,0FFH,0FFH,0FFH,080H
DB 000H,000H,0FFH,0FFH,0FEH,003H,0FFH,0FFH,0FFH,0FFH,0E1H,089H,080H,0E3H,0FFH,0FFH
DB 0FFH,0FFH,0FFH,080H,000H,000H,0FFH,0FFH,0FEH,003H,0FFH,0FFH,0FFH,0FFH,0E1H,089H
DB 080H,003H,0FFH,0FFH,0FFH,0FFH,0FFH,0C0H,000H,001H,0FFH,0FFH,0FEH,003H,0FFH,0FFH
DB 0FFH,0FFH,0F1H,089H,080H,003H,0FFH,0FFH,0FFH,0FFH,0FFH,0C0H,000H,001H,0FFH,0FFH
DB 0FEH,003H,0FFH,0FFH,0FFH,0FFH,0F1H,089H,080H,003H,0FFH,0FFH,0FFH,0FFH,0FFH,0E0H
DB 000H,001H,0FFH,0FFH,0FEH,003H,0FFH,0FFH,0FFH,0FFH,0F3H,009H,080H,003H,0FFH,0FFH
DB 0FFH,0FFH,0FFH,0E0H,000H,001H,0FFH,0FFH,0FEH,003H,0FFH,0FFH,0FFH,0FFH,0F3H,009H
DB 080H,003H,0FFH,0FFH,0FFH,0FFH,0FFH,0F0H,000H,001H,0FFH,0FFH,0FFH,007H,0FFH,0FFH
DB 0FFH,0FFH,0F2H,011H,080H,003H,0FFH,0FFH,0FFH,0FFH,0FFH,0F0H,000H,001H,0FFH,0FFH
DB 0FFH,007H,0FFH,0FFH,0FFH,0FFH,0F2H,011H,080H,003H,0FFH,0FFH,0FFH,0FFH,0FFH,0F8H
DB 000H,001H,0FFH,0FFH,0FFH,08FH,0FFH,0FFH,0FFH,0FFH,0F0H,021H,080H,003H,0FFH,0FFH
DB 0FFH,0FFH,0FFH,0F8H,000H,001H,0FFH,0FFH,0FFH,08FH,0FFH,0FFH,0FFH,0FFH,0F0H,021H
DB 080H,003H,0FFH,0FFH,0FFH,0FFH,0FFH,0F8H,000H,001H,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB 0FFH,0FFH,0F0H,001H,080H,003H,0FFH,0FFH,0FFH,0FFH,0FFH,0F8H,000H,001H,0FFH,0FFH
DB 0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0F0H,001H,080H,003H,0FFH,0FFH,0FFH,0FFH,0FFH,0F8H
DB 000H,001H,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,070H,001H,080H,003H,0FFH,0FFH
DB 0FFH,0FFH,0FFH,0F8H,000H,001H,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,070H,001H
DB 080H,001H,0FFH,0FFH,0FFH,0FFH,0FFH,0FCH,000H,001H,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB 0FEH,0FFH,070H,001H,080H,001H,0FFH,0FFH,0FFH,0FFH,0FFH,0FCH,000H,001H,0FFH,0FFH
DB 0FFH,0FFH,0FFH,0FFH,0FEH,0FFH,070H,001H,080H,000H,0FFH,0FFH,0FFH,0FFH,0FFH,0FCH
DB 000H,001H,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FEH,0FFH,070H,001H,080H,000H,0FFH,0FFH
DB 0FFH,0FFH,0FFH,0FCH,000H,001H,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FEH,0FFH,070H,001H
DB 080H,000H,0FFH,0FFH,0FFH,0FFH,0FFH,0FEH,000H,000H,0FFH,0FFH,0FFH,0FFH,0FFH,07FH
DB 0FFH,07FH,020H,001H,080H,000H,0FFH,0FFH,0FFH,0FFH,0FFH,0FEH,000H,000H,0FFH,0FFH
```

```
DB 0FFH,0FFH,0FFH,07FH,0FFH,07FH,020H,001H,080H,007H,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB 000H,000H,0FFH,0FFH,0FFH,0FFH,0FFH,0BFH,0FFH,07FH,020H,001H,080H,007H,0FFH,0FFH
DB 0FFH,0FFH,0FFH,0FFH,000H,000H,0FFH,0FFH,0FFH,0FFH,0FFH,0BFH,0FFH,07FH,020H,001H
DB 081H,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,080H,000H,07FH,0FFH,0FFH,0FFH,0FFH,0DFH
DB 0FFH,07EH,020H,001H,081H,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,080H,000H,07FH,0FFH
DB0FFH,0FFH,0FFH,0DFH,0FFH,07EH,020H,001H,081H,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB 0C0H,000H,03FH,0FFH,0FFH,0FFH,0FFH,0DFH,0FFH,03EH,040H,001H,081H,0FFH,0FFH,0FFH
DB0FFH,0FFH,0FFH,0FFH,0C0H,000H,03FH,0FFH,0FFH,0FFH,0FFH,0DFH,0FFH,03EH,040H,001H
DB083H,0FFH,0EFH,0FFH,0FFH,0FFH,0FFH,0FFH,0C0H,000H,01FH,0FFH,0FFH,0FFH,0FFH,0CFH
DB 0FFH,03CH,040H,001H,083H,0FFH,0EFH,0FFH,0FFH,0FFH,0FFH,0FFH,0C0H,000H,01FH,0FFH
DB 0FFH,0FFH,0FFH,0CFH,0FFH,03CH,040H,001H,083H,0E0H,007H,0FFH,0FFH,0FFH,0FFH,0FFH
DB 0C0H,000H,00FH,0FFH,0FFH,0FFH,0FFH,0CFH,0FEH,038H,040H,001H,083H,0E0H,007H,0FFH
DB 0FFH,0FFH,0FFH,0FFH,0C0H,000H,00FH,0FFH,0FFH,0FFH,0FFH,0CFH,0FEH,038H,040H,001H
DB 083H,0C0H,007H,0C0H,000H,03FH,0FFH,0F7H,0E0H,000H,007H,0FFH,0FFH,0FFH,0FFH,08FH
DB 0FEH,038H,040H,001H,083H,0C0H,007H,0C0H,000H,03FH,0FFH,0F7H,0E0H,000H,007H,0FFH
DB 0FFH,0FFH,0FFH,08FH,0FEH,038H,040H,001H,083H,080H,007H,0C0H,000H,03FH,0FFH,0E7H
DB 0F0H,000H,001H,0FFH,0FFH,0FFH,0FFH,08FH,0FEH,030H,040H,001H,083H,080H,007H,0C0H
DB 000H,03FH,0FFH,0E7H,0F0H,000H,001H,0FFH,0FFH,0FFH,0FFH,08FH,0FEH,030H,040H,001H
DB 083H,080H,007H,0C0H,000H,01FH,0FFH,0E3H,0F8H,000H,000H,0FFH,0FFH,0FFH,0FFH,01FH
DB 07CH,020H,040H,001H,083H,080H,007H,0C0H,000H,01FH,0FFH,0E3H,0F8H,000H,000H,0FFH
DB 0FFH,0FFH,0FFH,01FH,07CH,020H,040H,001H,083H,080H,00FH,080H,000H,01FH,0FFH,0C3H
DB 0F8H,000H,000H,07FH,0FFH,0FFH,0FEH,01EH,078H,000H,040H,001H,083H,080H,00FH,080H
DB 000H,01FH,0FFH,0C3H,0F8H,000H,000H,07FH,0FFH,0FFH,0FEH,01EH,078H,000H,040H,001H
DB 083H,000H,00FH,080H,000H,00FH,0E7H,0E1H,0FCH,000H,000H,03FH,0FFH,0FFH,0FEH,01EH
DB 0F0H,000H,000H,001H,083H,000H,00FH,080H,000H,00FH,0E7H,0E1H,0FCH,000H,000H,03FH
DB 0FFH,0FFH,0FEH,01EH,0F0H,000H,000H,001H,083H,080H,00FH,000H,000H,00FH,0F7H,0E0H
DB 0FEH,000H,000H,01FH,0FFH,0FFH,0DAH,03CH,0E0H,000H,000H,001H,083H,080H,00FH,000H
DB 000H,00FH,0F7H,0E0H,0FEH,000H,000H,01FH,0FFH,0FFH,0DAH,03CH,0E0H,000H,000H,001H
DB    087H,080H,01FH,000H,000H,007H,0F7H,0E0H,077H,000H,000H,007H,0FFH,0FFH,0F2H,030H
DB    080H,000H,000H,001H,087H,080H,01FH,000H,000H,007H,0F7H,0E0H,077H,000H,000H,007H
DB    0FFH,0FFH,0F2H,030H,080H,000H,000H,001H,087H,0C0H,01FH,000H,000H,003H,0F3H,0F0H
DB    073H,080H,000H,001H,0FFH,0FFH,0F1H,020H,000H,000H,000H,001H,087H,0C0H,01FH,000H
DB    000H,003H,0F3H,0F0H,073H,080H,000H,001H,0FFH,0FFH,0F1H,020H,000H,000H,000H,001H
DB    087H,0E0H,01EH,000H,000H,001H,0F9H,0F0H,071H,0C0H,000H,000H,0FEH,07FH,0F0H,000H
DB    000H,000H,000H,001H,087H,0E0H,01EH,000H,000H,001H,0F9H,0F0H,071H,0C0H,000H,000H
DB    0FEH,07FH,0F0H,000H,000H,000H,000H,001H,087H,0E0H,01EH,000H,000H,000H,0F8H,0F8H
DB    078H,0C0H,000H,000H,07CH,03FH,0F0H,000H,000H,000H,000H,001H,087H,0E0H,01EH,000H
DB    000H,000H,0F8H,0F8H,078H,0C0H,000H,000H,07CH,03FH,0F0H,000H,000H,000H,000H,001H
```

```
DB 083H,0E0H,01CH,000H,000H,000H,07CH,0F8H,07CH,060H,000H,000H,078H,01FH,0E0H,000H
DB 000H,000H,000H,001H,083H,0E0H,01CH,000H,000H,000H,07CH,0F8H,07CH,060H,000H,000H
DB 078H,01FH,0E0H,000H,000H,000H,000H,001H,081H,0E0H,018H,000H,000H,000H,07CH,0FCH
DB 07EH,030H,000H,000H,072H,007H,0E0H,000H,000H,000H,000H,001H,081H,0E0H,018H,000H
DB 000H,000H,07CH,0FCH,07EH,030H,000H,000H,072H,007H,0E0H,000H,000H,000H,000H,001H
DB 080H,000H,038H,000H,000H,000H,03FH,0F8H,07FH,010H,000H,000H,070H,000H,0E0H,000H
DB 000H,000H,000H,001H,080H,000H,038H,000H,000H,000H,03FH,0F8H,07FH,010H,000H,000H
DB 070H,000H,0E0H,000H,000H,000H,000H,001H,080H,000H,038H,000H,000H,000H,03FH,0E0H
DB 03CH,090H,000H,000H,060H,000H,078H,000H,000H,000H,000H,001H,080H,000H,038H,000H
DB 000H,000H,03FH,0E0H,03CH,090H,000H,000H,060H,000H,078H,000H,000H,000H,000H,001H
DB 080H,000H,078H,000H,000H,000H,03FH,080H,03CH,010H,000H,000H,060H,000H,060H,000H
DB 000H,000H,000H,001H,080H,000H,078H,000H,000H,000H,03FH,080H,03CH,010H,000H,000H
DB 060H,000H,060H,000H,000H,000H,000H,001H,080H,000H,078H,000H,000H,000H,07EH,000H
DB 03CH,010H,000H,000H,0C0H,003H,0F0H,000H,000H,000H,000H,001H,080H,000H,078H,000H
DB 000H,000H,07EH,000H,03CH,010H,000H,000H,0C0H,003H,0F0H,000H,000H,000H,000H,001H
DB 080H,000H,0F8H,000H,000H,000H,0FCH,000H,01EH,008H,000H,001H,040H,004H,0D8H,000H
DB 000H,000H,000H,001H,080H,000H,0F8H,000H,000H,000H,0FCH,000H,01EH,008H,000H,001H
DB 040H,004H,0D8H,000H,000H,000H,000H,001H,080H,001H,0F0H,000H,000H,000H,0F8H,000H
DB 007H,004H,000H,001H,000H,001H,008H,000H,000H,000H,000H,001H,080H,001H,0F0H,000H
DB 000H,000H,0F8H,000H,007H,004H,000H,001H,000H,001H,008H,000H,000H,000H,000H,001H
DB 080H,007H,0E0H,000H,000H,000H,0F0H,000H,000H,082H,000H,002H,00CH,002H,000H,000H
DB 000H,000H,000H,001H,080H,007H,0E0H,000H,000H,000H,0F0H,000H,000H,082H,000H,002H
DB 00CH,002H,000H,000H,000H,000H,000H,001H,080H,00FH,0C0H,000H,000H,001H,0F0H,000H
DB 000H,000H,000H,002H,040H,004H,000H,000H,000H,000H,000H,001H,080H,00FH,0C0H,000H
DB 000H,001H,0F0H,000H,000H,000H,000H,002H,040H,004H,000H,000H,000H,000H,000H,001H
DB 080H,00FH,0C0H,000H,000H,001H,0F0H,000H,000H,000H,000H,00CH,048H,000H,000H,000H
DB 000H,000H,000H,001H,080H,00FH,0C0H,000H,000H,001H,0F0H,000H,000H,000H,000H,00CH
DB 048H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H,000H,003H,0E0H,000H
DB 000H,000H,000H,038H,041H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H
DB 000H,003H,0E0H,000H,000H,000H,000H,038H,041H,000H,000H,000H,000H,000H,000H,001H
DB 080H,000H,000H,000H,000H,007H,0C0H,000H,000H,000H,00FH,0E2H,000H,000H,000H,000H
DB 000H,000H,000H,001H,080H,000H,000H,000H,000H,007H,0C0H,000H,000H,000H,00FH,0E2H
DB 000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H,000H,00FH,080H,000H
DB 000H,000H,000H,040H,000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H
DB 000H,00FH,080H,000H,000H,000H,000H,040H,000H,000H,000H,000H,000H,000H,000H,001H
DB 080H,000H,000H,000H,000H,01FH,080H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB 000H,000H,000H,001H,080H,000H,000H,000H,000H,01FH,080H,000H,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H,000H,01FH,000H,000H
```

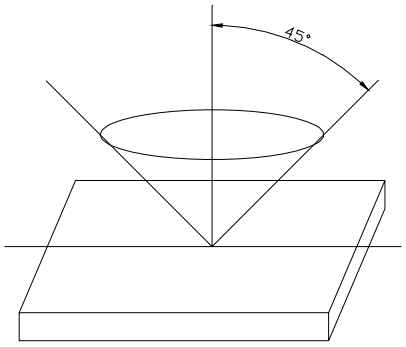
```
DB    000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H
DB    000H,01FH,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H
DB    080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H
DB    000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H
DB    080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H
DB    000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H
DB    080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H
DB    000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H
DB    080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H,080H,000H,000H,000H
DB    000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H
DB    0C0H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB    000H,000H,000H,003H,0C0H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,003H,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FF
DB0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FF

        END
```

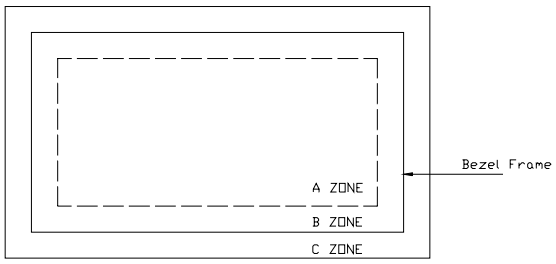
九：检测标准

9.1 检测样品的条件

玻璃应该在 40W 以内的白灯下检测，目测的距离应该在 30cm 以内。
检测样品的方向应该是在以法线为中心的 45 度以内。



9.2 应用区域的定义

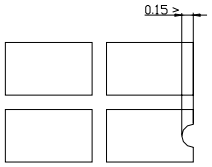


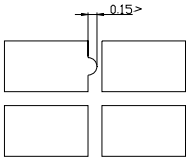
A Zone: 有效显示区域
B Zone: 铁框和有效显示区域之间的距离
C Zone: 铁框边距离
A Zone + B Zone=有效的参观区域

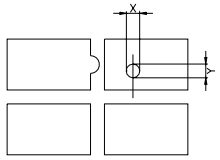
9.3 标准

序列	参数	标准

<table>
<tr><td>1</td><td>黑点和白点,
脏点</td><td>圆点
<table><tr><td>Zone
DIMENSION(MM)</td><td colspan="3">可接受的数值</td></tr><tr><td></td><td>A</td><td>B</td><td>C</td></tr><tr><td>D≤0.1</td><td>*</td><td>*</td><td>*</td></tr><tr><td>0.1<D≤0.2</td><td>5</td><td>5</td><td>*</td></tr><tr><td>0.2<D≤0.3</td><td>0</td><td>1</td><td>*</td></tr><tr><td>0.3<D</td><td>0</td><td>0</td><td>*</td></tr></table>D=(长+宽)/2 * 不考虑

长点
<table><tr><td colspan="2">Zone
X(mm) Y(mm)</td><td colspan="3">Acceptable Number</td></tr><tr><td></td><td></td><td>A</td><td>B</td><td>C</td></tr><tr><td>-</td><td>0.02≥W</td><td>*</td><td>*</td><td>*</td></tr><tr><td>2.0≥L</td><td>0.03≥W</td><td>3</td><td>3</td><td>*</td></tr><tr><td>1.0≥L</td><td>0.04≥W</td><td>1</td><td>2</td><td>*</td></tr><tr><td>1.0≥L</td><td>0.05≥W</td><td>0</td><td>2</td><td>*</td></tr><tr><td>-</td><td>0.05<W</td><td colspan="3">Not acceptable</td></tr></table>X: 长 Y: 宽 * 不考虑</td></tr>
<tr><td>2</td><td>气泡
(玻璃和偏光片
之间)</td><td><table><tr><td>Zone</td><td colspan="3">接受数值</td></tr><tr><td></td><td>A</td><td>B</td><td>C</td></tr><tr><td>D≤0.1</td><td>*</td><td>*</td><td>*</td></tr><tr><td>0.1<D≤0.2</td><td>5</td><td>5</td><td>*</td></tr><tr><td>0.2<D≤0.3</td><td>0</td><td>1</td><td>*</td></tr><tr><td>0.3<D</td><td>0</td><td>0</td><td>*</td></tr></table>*: 不考虑</td></tr>
<tr><td>3</td><td>不规格的点</td><td>(1) 点形状（缺点）
 

(2)点形状（多点）
 
</td></tr>
</table>

		不能和旁边的点阵相连. (3) 针孔  (X+Y)/2<0.2mm (小于 0.1mm 是不考虑的)
4	偏光片划伤	根据实际情况而定.
5	偏光片脏点	如果脏点是在 LCD 的表面，则不能算是不合格品.
6	玻璃彩虹	按照实际的情况而定.

十：使用注意事项

十分感谢您购买我公司的产品，在使用前请您首先仔细阅读以下注意事项，以免给您造成不必要的损失，您在使用过程中遇到困难时，请拨打我们的服务电话，我们将尽力为您提供服

务和帮助。

1．处理保护膜

在装好的模块成品表面贴有一层保护膜，以防在装配时沾污显示表面，在整机装配结束前不得撕去以免弄脏或损坏表面。

2．加装衬垫

在模块和前面板之间最好加装一块约0.1 毫米左右的衬垫。面板还应保持平整，以免在装配后产生扭曲，并可提高其抗振性能。

3．严防静电

模块中的控制、驱动电压是很低、微功耗的CMOS 电路，极易被静电击穿，静电击穿是一种不可修复的损坏，而人体有时会产生高达几十伏或上百伏的静电，所以，在操作、装配以及使用中都应极其小心，严防静电。为此：

（1） 不要用手随意去摸外引线、电路板上的电路及金属框。

（2） 如必须直接接触时，应使人体与模块保持在同一电位，或使人体良好接地。

（3） 焊接使用的烙铁及装配使用的电动工具必须良好接地，没有漏电。

（4） 不得使用真空吸尘器进行清洁处理，因为它会产生很强的静电。

（5） 空气干燥也会产生静电，因此，工作间湿度应在RH60%以上。

（6） 取出或放回包装袋或移动位置时，也需小心，防止产生静电。不要随意更换包装或舍弃原包装。

4．装配操作时的注意事项

（1） 模块是经过精心设计组装而成的，请勿随意自行加工、修整。

（2） 金属框爪不得随意扭动、拆卸。

（3） 不要随意修改加工PCB 板外形、装配孔、线路及其部件。

（4） 不得修改导电胶条。

（5） 不得修改任何内部支架。

（6） 不要碰、摔、折曲、扭动模块。

5．焊接

在焊接外引线时，应按如下规程进行操作。

（1） 烙铁头温度小于280 度。

（2） 焊接时间不超过4 秒。

（3） 焊接材料：共晶型、低熔点。

（4） 不要使用酸性助焊剂。

（5） 重复焊接不要超过三次，且每次重复需间隔5 分钟。

6．模块的使用与保养

（1） 模块的外引线决不允许接错，在您想调试液晶模块时，请注意正确接线，尤其是正负电源的接线不能接错，否则可能造成过流、过压烧电路上的芯片等对液晶模块元器件有损的现象。

（2） 模块在使用时，接入电源及断开电源，必须在正电源稳定接入以后才能输入信号电平。如在电源稳定前或断开后输入信号电平，有可能损坏模块中的IC 及电路。

（3） 点阵液晶模块显示时的对比度、视角与温度、驱动电压的关系很大，所以，如果驱动电压过高，不仅会影响显示效果，还会缩短模块的使用寿命。

（4） 因为液晶材料的物理特性，液晶的的对比度会随温度的变化而相应变化，所以，您加的负压也应随温度作相应调整。大致是温度变化10 度，电压变化1 伏。为满足这一要求，您可以做一个温度补偿电路，或者安排一个电位器，随温度调整负电压值。

（5） 不应在规定工作温度范围外使用，并且不应在超过存储极限温度的范围外存储。如果温度低于结晶温度，液晶就会结晶，如果温度过高，液晶将变成各向同性的液晶，破坏分子取向，使器件报废。

（6） 用力按显示部分，会产生异常显示。这时切断电源，稍待片刻重新上电，即恢复正常。

（7） 液晶显示器件或模块表面结雾时，不要通电工作，因为这将引起电极化学反应，产生断线。

（8） 长期用于阳光及强光下时，被遮部分会产生残留现象。

7．模块的存储

若长期（如几年以上）存储，我们推荐以下方式：

（1） 装入聚乙稀口袋（最好有防静电涂层）并将口封住

（2） 在-10° C --- +35° C 之间存储。

（3） 放在暗处，避强光。

（4） 决不能在表面压放任何物品。

（5） 严格避免在极限温度/湿度条件下存放。

保修是以上述注意事项未被忽视为先决条件的，典型的违反例子如下：

（1）断裂的液晶显示屏玻璃。

（2）线路板孔修改或损坏。

（3）线路板布线损坏。

（4）电路修改，包括元件的增加。

（5）线路板随意研磨、雕刻或油漆。

（6）焊接或更改玻璃框。

模块维修将基于双方协议下列出给顾客的清单。模块必须与防静电包装和故障详细陈述一起送回。顾客安装的连接器或电缆必须坏线路板孔，线路和引线端条件下全部移去在不破坏线路板孔，线路和引线端条件下全部移去。