RAiO

RA8815

128x33 图形/文字

LCD 驱动控制器

应 用 手 册

Version 1.4

March 2, 2006, 2006

RAiO Technology Inc.

RA8815 中文文字/图形 LCD 控制器应用手册改版说明		
版　　本	日　　期	说　　明
1.0	November 3, 2004	第一版
1.1	January 26, 2005	修正图 3-1
1.2	June 22, 2005	修正图 13-1 增加附录 D. 字型与字码表(BIG-5)
1.3	January 3, 2006	图 13-1：冷光驱动信号与电路 附录 E. 字型使用说明
1.4	March 2, 2006	修正图 4-3：串行界面硬件设定方式 修正图 4-4B : 6800 MPU 时序图 修正第 6 章: 5x4 键盘扫描 修正图 7-1 : I/O 驱动 LED 范例

章 节 内 容 页 数

1. 简介

本应用手册主要是针对 RA8815 特有的功能做一说明，大部分基本功能的设定请参考 RA8815 规格书。以下几个章节分别以范例程序教导使用者如何去设定特定功能的相关缓存器，其中包括文字卷动与移动功能的设定方式与绘图模式卷动及移动的设定方式。另外，RA8815 也提供与微处理机之间多种连接的接口，包括并列与串行模式的选择等等。

2. 系统重置

当系统电源开启之后，微控制器(MPU)需要对 RA8815 作重置的动作，亦就是控制 RA8815 的重置输入脚(RST)为低电位至少 5 毫秒，再控制(RST)为高电位至少 350 毫秒，重置完成之后，才可对 RA8815 开始继续作其它的设定。图 2-1 为 RA8815 重置(Reset)的时序图。

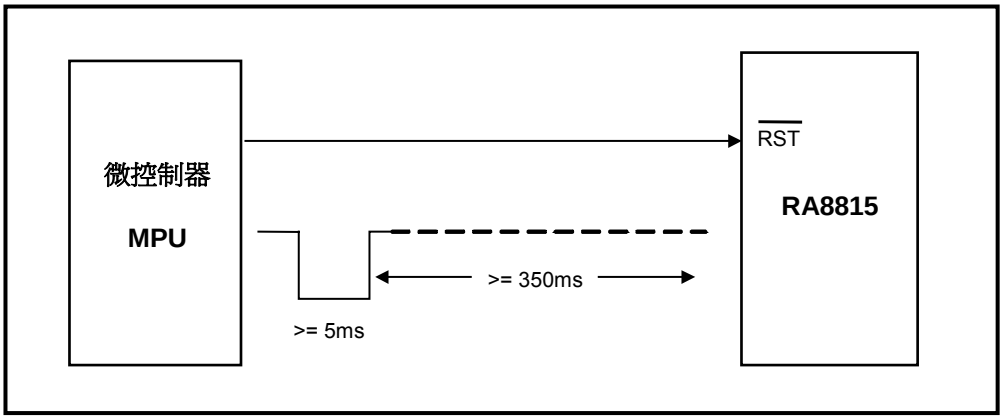


图 2-1：RA8815 重置时序

以下为 图 2-1 范例程序：

```
void LCD_Reset(void)                    //重置子程序
{
  //LCD_RST = 1;
  //delay(100);
  LCD_RST = 0;                          //控制/RST 为低电位
  delay(5ms);                           //延迟 5 毫秒
  LCD_RST = 1;                          //控制/RST 为高电位
  delay(350ms);                         //延迟 350 毫秒
}
```

3. 系统频率

RA8815 系统频率的产生主要可分为：

1. 透过内部 RC 振荡电路产生系统频率
2. 外部直接输入系统频率

其硬件初始设定方式如图 3-1：

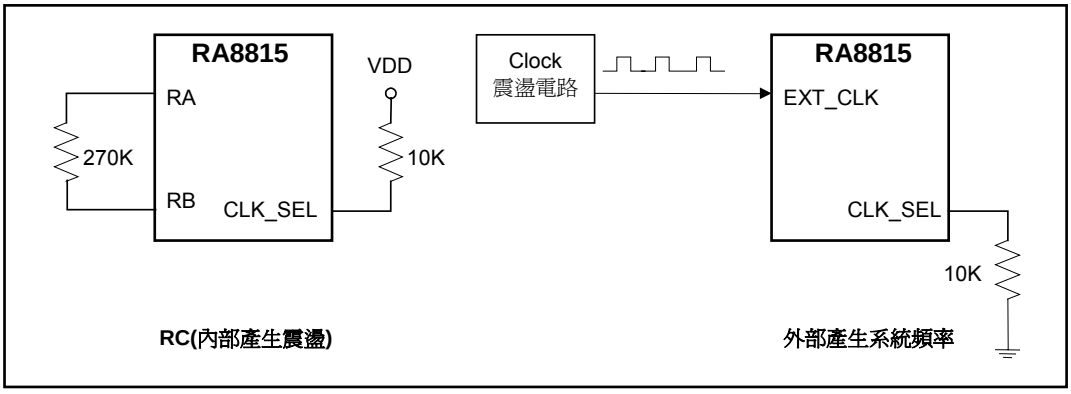


图 3-1：RA8815 系统频率的设定

4. MPU 界面

RA8815 提供七种与 MPU 沟通的接口包括：

1. 8080 MPU (8 位与 4 位数据传输宽度)并列连接界面，如图 4-1。
2. 6800 MPU (8 位与 4 位数据传输宽度)并列连接界面，如图 4-2。
3. 三线式及四线式(Type A 与 Type B)的串行连接界面，如图 4-3。

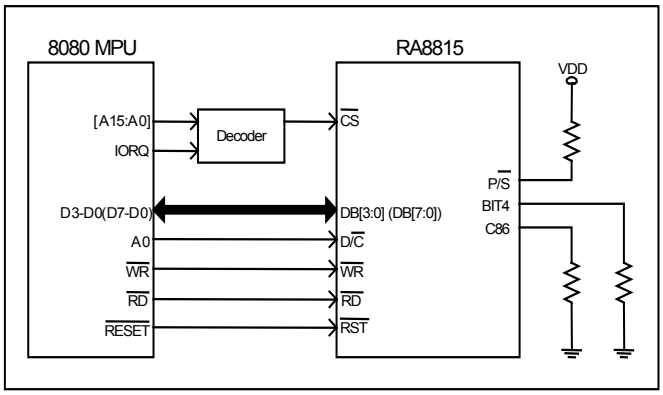


图 4-1 : 8080 MPU 8/4 位硬件设定方式

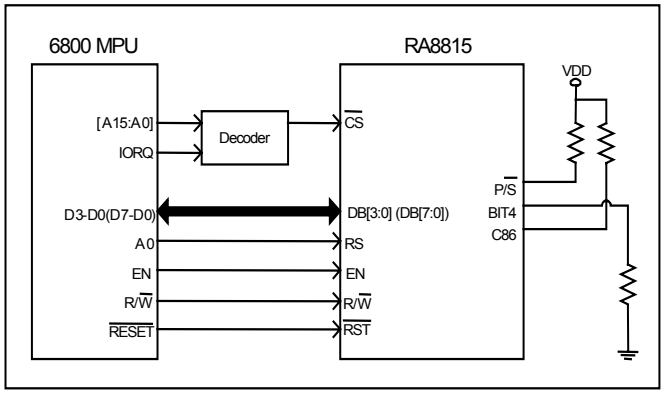


图 4-2 : 6800 MPU 8/4 位硬件设定方式

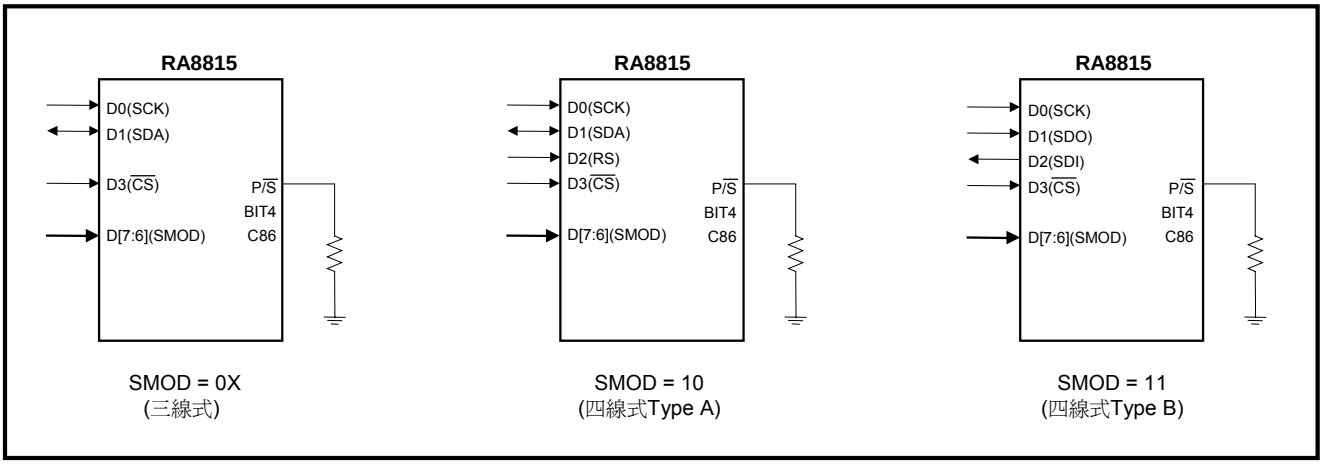


图 4-3 : 串行界面硬件设定方式

对于并列界面，现在的使用者对 6800 MPU 较不熟悉，且一般使用者比较熟悉 8051 系列的 MPU，也常用 8051 做为系统开发，因此建议直接使用 8080 的 MPU 接口，避免因不熟悉 6800 界面而产生使用上的困扰。

以下图 4-4 到 4-7 为各种界面的时序图：

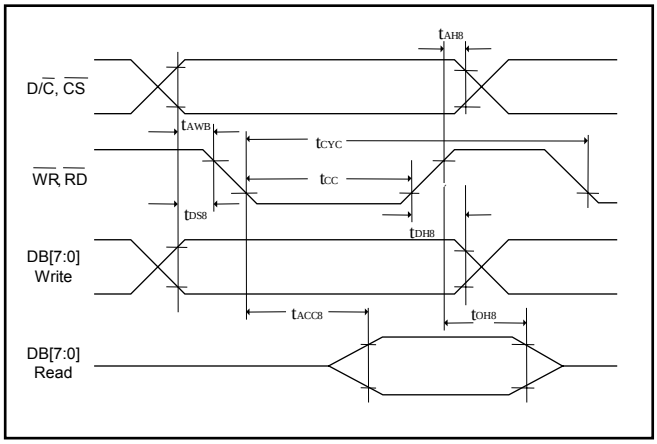


图 4-4A : 8080 MPU 时序图

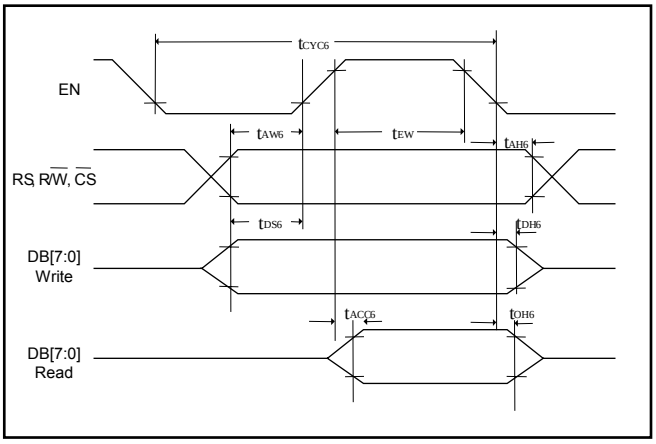


图 4-4B : 6800 MPU 时序图

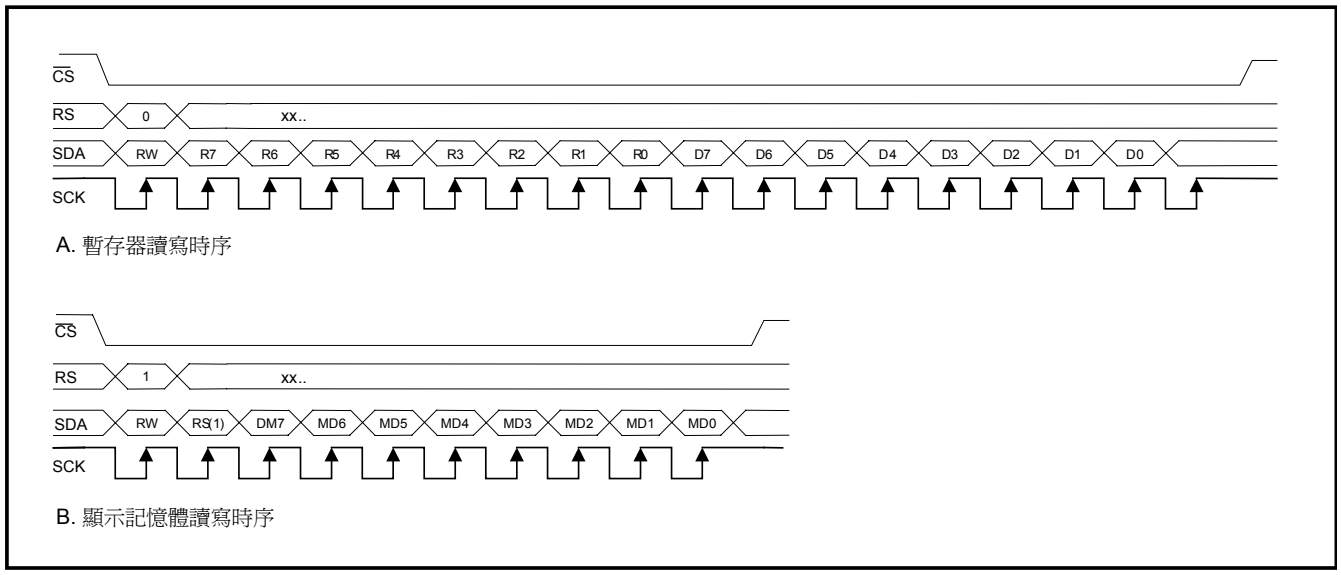


图 4-5： 四线式 Type A 串行界面时序图

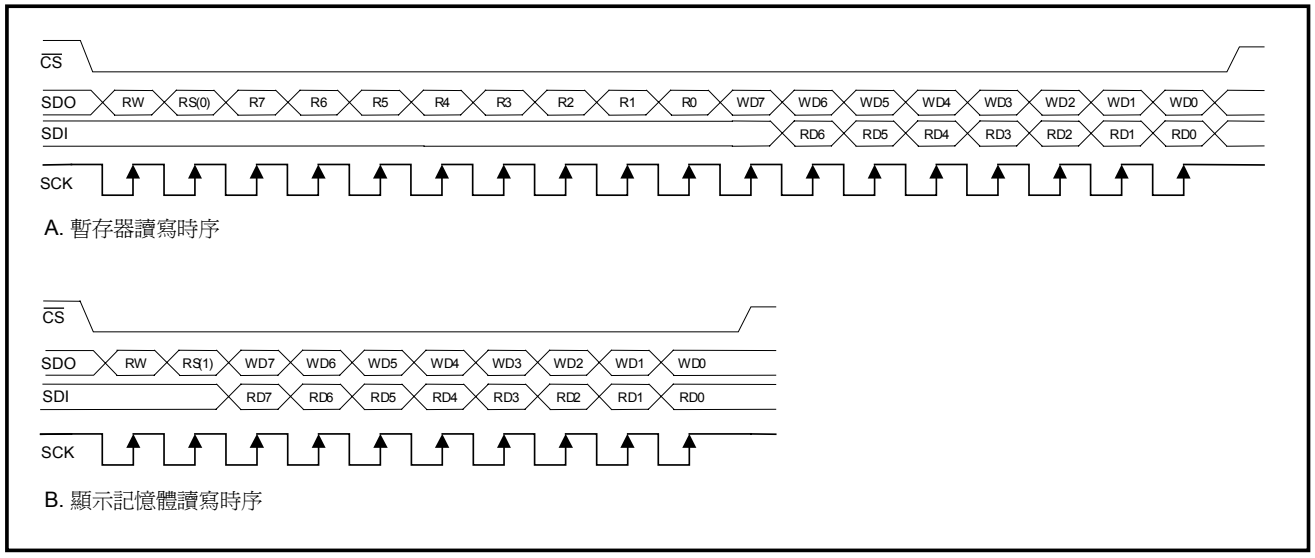


图 4-6： 四线式 Type B 串行界面时序图

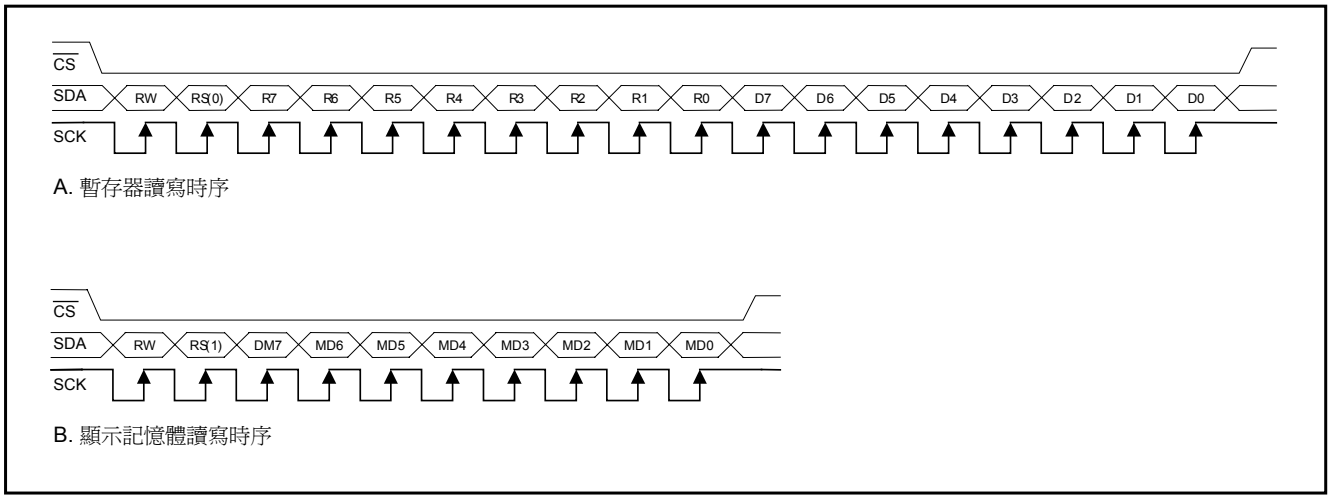


图 4-7： 三线式串行界面时序图

5. LCD 驱动电压设定

RA8815 的 LCD 驱动电压可由 IC 内部电路直接产生，或是经由外部电路供给，详细驱动原理请参考规格书 6-4 节的 LCD 驱动器与电压供应电路。根据以下的设定方式即可将内部的升压电路启动与输出，但实际显示效果需针对各种使用者的 LCD 做最后的调整。

以下为程序范例：

(a)使用内部升压电路方式：

```
LCD_CmdWrite(0x11,0xF0);        // Bit7:  使用内部升压电路
                                // Bit6:  使用内部产生的参考电压给内部电压调整器
                                // Bit5:  使用内部电压调整器电路产生”VREG”
                                // Bit4:  使用内部分压电路产生 LCD Bias 电压(V0~V4)

LCD_CmdWrite(0x12,0x17);        // Bit[7:6]内部升压电路硬件震荡频率选择
                                // Bit[5:3]电压调整器输出倍数调整
                                // Bit[2:0]输出驱动电流的大小调整

LCD_CmdWrite(0x10,0x5C);        // Bit[7:5]设定 1/5Bias
                                // Bit[4:0]LCD 显示对比的调整
```

6. 5x4 键盘扫描

RA8815 内建 5x4 键盘扫描功能，使用者只要透过缓存器设定即可很容易的读取按键数值。以下说明键盘扫描实际应用时程序的撰写方式。

6-1 写入键盘扫描控制缓存器

缓存器设定：

	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
REG[07h]	KSB	KDB1	KDB0	KSTB_SEL	K_AUTO	IRE	KF1/KSTB1	KF0/KSTB0

Bit	Description	Default	Access
7	键盘扫描功能设定。0 ➔ 键盘扫描功能关闭，1 ➔ 键盘扫描功能开启。	0h	W
6-5	键盘扫描为自动模式(Auto-Mode)时，设定消除弹跳电路(De-bounce)的扫描次数。(每次代表键盘扫描一次的时间) KDB1 / KDB0 / 扫描次数 0 / 0 / 8 0 / 1 / 16 1 / 0 / 32 1 / 1 / 64	0h	W
4	当键盘扫描为非自动模式时，0 ➔ 此缓存器的 DB[1:0]定义为 KF[1:0]，1 ➔ 此缓存器的 DB[1:0]定义为 KSTB[1:0] 。当键盘扫描为自动模式时，此缓存器的 DB[1:0]定义为 KF[1:0]。	0h	W
3	键盘扫描模式设定，1➔ 选择自动模式(Auto-Mode)，RA8815 会自动判断被按下的键，并存在 AKD[6:0]以供 MPU 读取。0➔ 选择非自动模式(Non-Auto-Mode)，RA8815 不会自动判断按下的键，但软件可经由 KSTB[1:0]与 KSD[4:0]判断目前所按下的键，可处理多键同时按的功能。	0h	W
2	键盘扫描之硬件中断设定。0 ➔ 当键盘被按下时不会产生硬件中断，1 ➔ 当键盘被按下时会产生硬件中断。	0h	W

<table>
<tr>
<td>1-0</td>
<td>当 Bit3 键盘扫描设定为非自动模式，且 Bit4=”1”时，设定欲读回按键数据(KSD[4:0])所相对应的列数。
<table>
<tr><td>0</td><td>0</td><td>：扫描 KST 第一列</td></tr>
<tr><td>0</td><td>1</td><td>：扫描 KST 第二列</td></tr>
<tr><td>1</td><td>0</td><td>：扫描 KST 第三列</td></tr>
<tr><td>1</td><td>1</td><td>：扫描 KST 第四列</td></tr>
</table>
当 Bit3 键盘扫描设定为自动模式，或是 Bit4=”0”时，当做键盘扫描频率设定。
<table>
<tr><th>KF1</th><th>KF0</th><th>键盘扫描脉波宽度</th><th>键盘扫描周期 (4x5)</th></tr>
<tr><td>0</td><td>0</td><td>256us</td><td>1.024ms</td></tr>
<tr><td>0</td><td>1</td><td>512us</td><td>2.048ms</td></tr>
<tr><td>1</td><td>0</td><td>1.024ms</td><td>4.096ms</td></tr>
<tr><td>1</td><td>1</td><td>2.048ms</td><td>9.182ms</td></tr>
</table>
</td>
<td>0h</td>
<td>W</td>
</tr>
</table>

6-2 读取键盘扫描数据缓存器(Non-Auto Mode)

缓存器设定：

REG[07h]	***DB7***	***DB6***	***DB5***	***DB4***	***DB3***	***DB2***	***DB1***	***DB0***
	SIRQ	KSTB1	KSTB0	KSD4	KSD3	KSD2	KSD1	KSD0

Bit	Description	Default	Access
7	键盘扫瞄中断旗标 0：无按键发生 1：有按键发生 (设定 REG[0Fh]-Bit1=0，可清除此中断旗标为 0)	0h	R
6-5	读取目前设定扫描的列数，用来表示脚位 KST[3:0]是哪一个在动作。 0 0 ：扫描 KST 第一列 0 1 ：扫描 KST 第二列 1 0 ：扫描 KST 第三列 1 1 ：扫描 KST 第四列	0h	R

<table>
<tr>
<td>4-0</td>
<td>读取目前扫描的行数，可以由 MPU 读取 KSD[4:0] 与 KSTB[1:0] 来判断哪一个键被按下。
<table>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>：KIN 第一行有按键输入</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>：KIN 第二行有按键输入</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td><td>：KIN 第三行有按键输入</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td><td>：KIN 第四行有按键输入</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>：KIN 第五行有按键输入</td></tr>
</table>
</td>
<td>1Fh</td>
<td>R</td>
</tr>
</table>

6-3 键盘扫描硬件电路

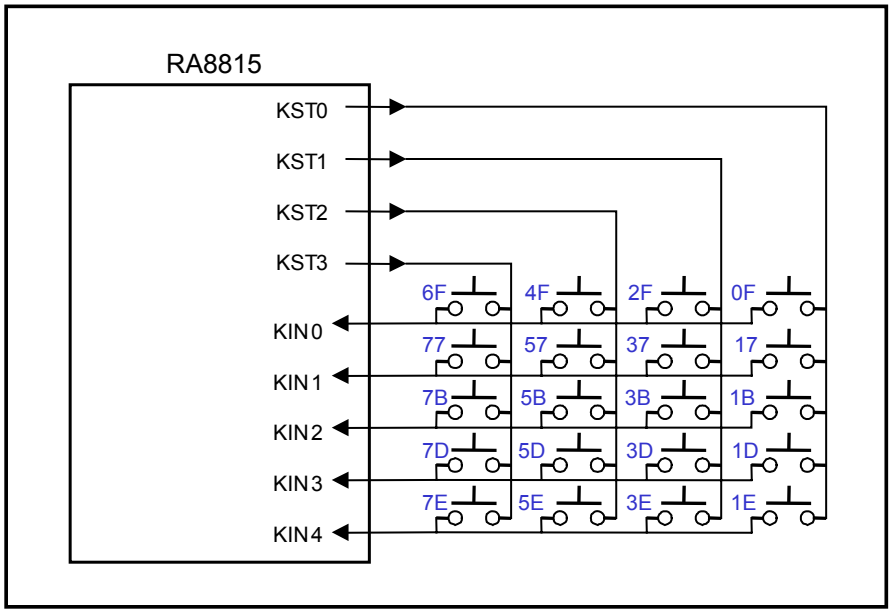


图 6-1: 键盘扫描(5x4)硬件电路

6-4 程序范例

(a)键盘扫瞄(非自动模式)

```
LCD_Initial ();
LCD_Clear();                          // 清除所有显示
LCD_CmdWrite (0x03,0x03);             // 设定绘图显示模式
LCD_CmdWrite (0x04,0x74);             // 关闭游标
LCD_CmdWrite (0x01,0x02);             // 开启 LCD 显示
GotoXY (0,0);                         // 设定光标地址
```

```
PrintStr ("Key Scan：手动",1);                          //  LCD 显示字符串
LCD_CmdWrite(0x07,0x80);                                // 启动键盘为非自动模式扫描
LCD_CmdWrite(0x0F,0x00);                                // 清除所有中断旗标
while(1)
{
  while((LCD_CmdRead_SPI3(0x07) & 0x80) == 0x80)        // 判断是否有任何按键输入
  {
    GotoXY_SPI3(5,16);
    PutHEX(Check_Key_Number(0));                        // 显示 KOUT0 第一列扫描 KIN0~KIN4 的数值
    PutHEX(Check_Key_Number(1));                        // 显示 KOUT1 第二列扫描 KIN0~KIN4 的数值
    PutHEX(Check_Key_Number(2));                        // 显示 KOUT2 第三列扫描 KIN0~KIN4 的数值
    PutHEX(Check_Key_Number(3));                        // 显示 KOUT3 第四列扫描 KIN0~KIN4 的数值
  }
}


unsigned Check_Key_Number(unsigned char row)
{
  unsigned char next_status[4],KSTB[4] = {0,0,0,0};     // 定义使用者暂存数据的变量
  static char prev_ststus[4]= {0,0,0,0};                //

  LCD_CmdWrite(0x07,0x90);                              // 设定扫描第一列
    next_status[0] = LCD_CmdRead(0x07) & 0x7f;          // 读取 KIN0~KIN4 的按键数值
  LCD_CmdWrite(0x07,0x91);                              // 设定扫描第二列
    next_status[1] = LCD_CmdRead(0x07) & 0x7f;          // 读取 KIN0~KIN4 的按键数值
  LCD_CmdWrite(0x07,0x92);                              // 设定扫描第三列
    next_status[2] = LCD_CmdRead(0x07) & 0x7f;          // 读取 KIN0~KIN4 的按键数值
  LCD_CmdWrite(0x07,0x93);                              // 设定扫描第四列
    next_status[3] = LCD_CmdRead(0x07) & 0x7f;          // 读取 KIN0~KIN4 的按键数值

  if(next_status[0] != 0x1F )                           // 判断按键之后是否已经放开
    prev_ststus[0] = next_status[0];                    // 若是放开按键，读取第一列扫描数值
  Else
    KSTB[0] = prev_ststus[0];                           // 若是未放开按键，将第一列扫描数值设定为 0
```

```
    if(next_status[1] != 0x3F )                    // 判断按键之后是否已经放开
      prev_ststus[1] = next_status[1];             // 若是放开按键，读取第二列扫描数值
    Else
      KSTB[1] = prev_ststus[1];                    // 若是未放开按键，将第二列扫描数值设定为 0

    if(next_status[2] != 0x5F )                    // 判断按键之后是否已经放开
      prev_ststus[2] = next_status[2];             // 若是放开按键，读取第三列扫描数值
    Else
      KSTB[2] = prev_ststus[2];                    // 若是未放开按键，将第三列扫描数值设定为 0

    if(next_status[3] != 0x7F )                    // 判断按键之后是否已经放开
      prev_ststus[3] = next_status[3];             // 若是放开按键，读取第四列扫描数值
    Else
     KSTB[3] = prev_ststus[3];                     // 若是未放开按键，将第四列扫描数值设定为 0

    LCD_CmdWrite(0x0F,LCD_CmdRead(0x0F) &          // 清除键盘扫瞄中断旗标
  0xFD);
    return KSTB[row];                              // 回传按键数值
  }
```

(b)键盘扫瞄(自动模式)

```
  LCD_Initial ();
  LCD_Clear();                                     // 清除所有显示
  LCD_CmdWrite (0x03,0x03);                        // 设定绘图显示模式

  LCD_CmdWrite (0x04,0x74);                        // 关闭游标
  LCD_CmdWrite (0x01,0x02);                        // 开启 LCD 显示
  GotoXY (0,0);                                    // 设定光标地址
  PrintStr("Key Scan：自动",1);                    //  LCD 显示字符串
  LCD_CmdWrite (0x07,0x88);                        // 启动键盘为非自动模式扫描
  LCD_CmdWrite (0x0F,0x00);                        // 清除所有中断旗标
  while(1)
  {
    while((LCD_CmdRead_SPI3(0x07) & 0x80) == 0x80)  // 判断是否有任何按键输入
```

```
    {
      key_number = Check_Key_Number();          // 若有按键输入，读取扫描数值
      GotoXY(0,16);                             // 设定光标地址
      PutHEX(key_number);                       // 显示按键数值
    }
}


unsigned Check_Key_Number(void)
{
   unsigned char next_status,key_number = 0xff;  // 设定使用者变量
   static char prev_ststus = 0xff;

   next_status = LCD_CmdRead(0x07) & 0x7f;       // 读取键盘扫瞄数值

   if(next_status != 0x42)                       // 判对按键是否放开
      prev_ststus = next_status;                 // 若是未放开，继续扫描
   else
      key_number = prev_ststus;                  // 若已经放开按键，将回传真正按键的数值
   LCD_CmdWrite(0x0F,0x00);                      // 清除键盘扫瞄中断旗标
   return key_number;
}
```

7. 双向 IO 埠控制

RA8815 提供八个可双向控制的输出入 I/O 埠，使用者可任意的改变每一个 I/O 埠为输入或输出，而当设定为输出埠时，I/O 也可直接驱动 LED，连接方式如图 7-1，缓存器设定方式如下:

缓存器设定：(I/O 埠方向设定)

REG[14h]	***DB7***	***DB6***	***DB5***	***DB4***	***DB3***	***DB2***	***DB1***	***DB0***
	OE7	OE6	OE5	OE4	OE3	OE2	OE1	OE0

缓存器设定：(I/O 端口数据读写缓存器)

REG[15h]	***DB7***	***DB6***	***DB5***	***DB4***	***DB3***	***DB2***	***DB1***	***DB0***
	IOD7	IOD6	IOD5	IOD4	IOD3	IOD2	IOD1	IOD0

范例程序：

```
LCD_CmdWrite(0x14,0xFF);          // 设定 8 个 I/O 端口为输出模式
LCD_CmdWrite(0x15,0x55);          // 设定 8 个 I/O 输出为”01010101”
```

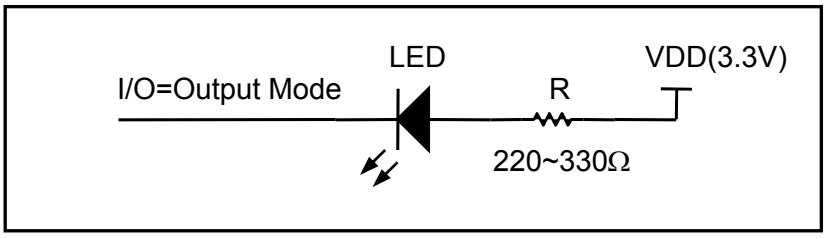


图 7-1 : I/O 驱动 LED 范例

8. 基本显示功能设定

RA8815 中文显示除了一般正常字型显示之外，也可将字型设定为粗体显示或是将字型作反白的显示。

8-1 文字模式-字型正常显示

可由缓存器 REG-[03h]之 BIT[1：0]设定，此时 RA8815 会将被写入的数据译码为全型字型的显示。

REG[03h]	***DB7***	***DB6***	***DB5***	***DB4***	***DB3***	***DB2***	***DB1***	***DB0***
	BMOD1	BMOD0	BIEN	ASCS	BOLD	INV	**MD1**	**MD0**

Bit	Description	Default	Access
1-0	显示模式选择 0 0 ：绘图模式 0 1 ：小 ASCII 模式 1 0 ：大 ASCII 模式 1 1 ：全角模式(16x16 文字)	0h	R/W

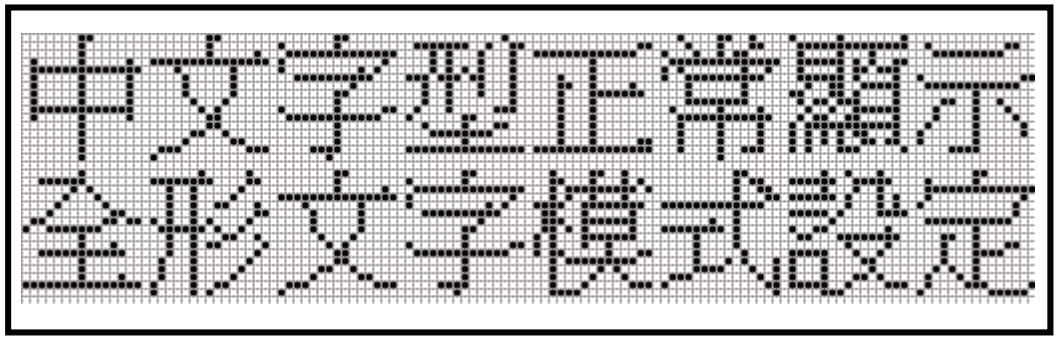


图 8-1： 文字模式-的正常显示

范例程序：

```
LCD_Clear_LCD                          // 清除所有显示
LCD_CmdWrite(0x01,0x02);               // 开启 LCD 显示

LCD_CmdWrite(0x03,0x03);               // 文字模式
GotoXY(0,0);                           // 设定光标地址
// PrintStr("中文字型正常显示",1);      // 显示中文字符串
```

```
LCD_DataWrite(0xA4);                    // 中
LCD_DataWrite(0xA4);

LCD_DataWrite(0xA4);                    // 文
LCD_DataWrite(0xE5);

LCD_DataWrite(0xA6);                    // 字
LCD_DataWrite(0x72);

LCD_DataWrite(0xAB);                    // 型
LCD_DataWrite(0xAC);

LCD_DataWrite(0xA5);                    // 正
LCD_DataWrite(0xBF);

LCD_DataWrite(0xB1);                    // 常
LCD_DataWrite(0x60);

LCD_DataWrite(0xC5);                    // 显
LCD_DataWrite(0xE3);

LCD_DataWrite(0xA5);                    // 示
LCD_DataWrite(0xDC);

GotoXY(0,16);                           // 设定光标地址
PrintStr("全角文字模式设定",1);          // 显示中文字符串
While(1);
```

8-2 文字模式-粗体显示

可由缓存器 REG-[03h]之 BIT[3]设定，将原来正常的全型字转变为粗体字型再写入显示内存。

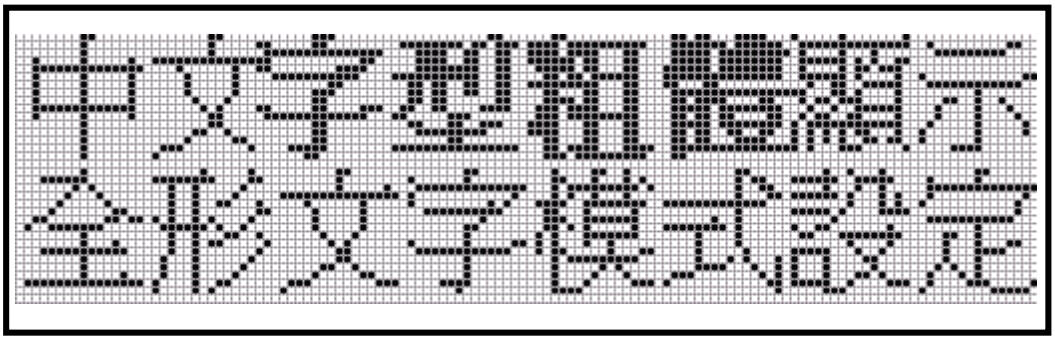


图 8-2： 文字模式-的粗体显示

范例程序：

```
.
LCD_CmdWrite(0x03,0x03);                    // 设定一般字型模式
GotoXY(0,0);                                // 设定光标地址
PrintStr("中文",1);                          // 显示”中文”字符串

LCD_CmdWrite(0x03,0x0B);                    // 设定粗体字型模式
PrintStr("字型粗体",1);                      // 显示”字型粗体”字符串

LCD_CmdWrite(0x03,0x03);
PrintStr("显示",1);                          // 设定一般字型模式
PrintStr("全角文字模式设定",1);               // 显示”全角文字模式设定”字符串
While(1);
:
:
:
```

8-3 文字模式-字型反白显示

可由缓存器 REG-[03h]之 BIT[2] 设定，将原来正常的全型字转变为反向字型再写入显示内存。

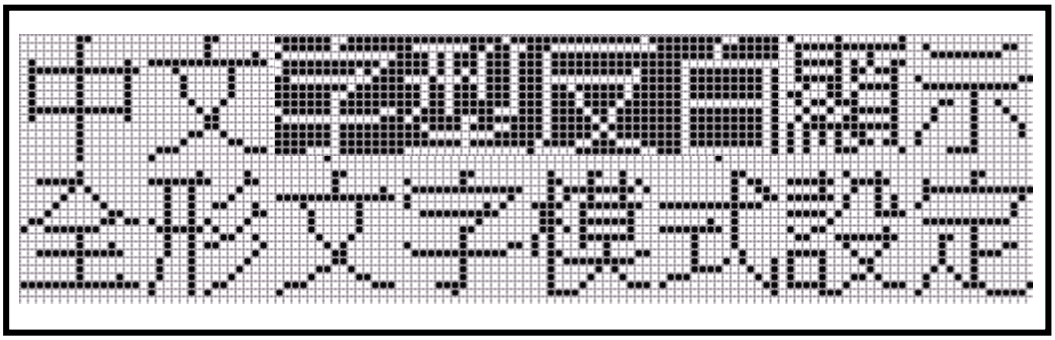


图 8-3： 文字模式-的字型反白显示

范例程序：

```
.
LCD_CmdWrite(0x03,0x03);                    // 设定一般字型模式
GotoXY(0,0);                                // 设定光标地址
PrintStr("中文",1);                         // 显示”中文”字符串

LCD_CmdWrite(0x03,0x07);                    // 设定字型反白模式
PrintStr("字型反白",1);                     // 显示”字型反白”字符串

LCD_CmdWrite(0x03,0x03);
PrintStr("显示",1);                         // 设定一般字型模式
PrintStr("全角文字模式设定",1);             // 显示”全角文字模式设定”字符串
While(1);
:
:
:
```

9. 文字模式卷动

RA8815 提供 LCD 显示画面可经由缓存器的设定指定一区块，使硬件自动做屏幕画面卷动的效果，其中缓存器的功能设定主要包刮卷动画面的方向与速度，及每次卷动画面的像数与卷动的范围，以下针对实际应用时程序撰写方式作一说明。

9-1 卷动方向

可由缓存器 REG-[0Eh]之 BIT[3：2] 设定，选择四种不同的卷动方式。

缓存器设定：

	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
REG[0Eh]	SCR_IMD1	SCR_IMD0	AUTO_SCR	SBUF	**SCR_MD1**	**SCR_MD0**	SCR_INTEN	SCR_EN

Bit	Description	Default	Access
3-2	设定移动方向： 0 0 ：由左往右卷动(水平卷动) 0 1 ：由右往左卷动(水平卷动) 1 0 ：由上往下卷动(垂直卷动) 1 1 ：由下往上卷动(垂直卷动)	0h	R/W

9-2 卷动范围设定

设定 REG-[08h]’REG-[09h]’REG-[0Ah]’REG-[0Bh]四个缓存器，可指定一区块作为卷动的范围。

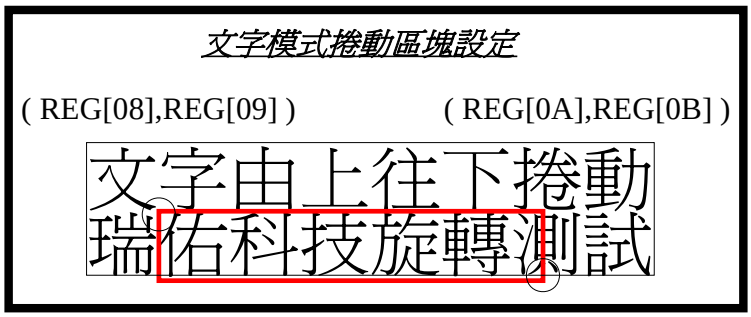


图 9-1： 设定卷动范围

(a)缓存器设定：卷动区块水平起始地址(X1)

REG[08h]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	--	--	--	SSX4	SSX3	SSX2	SSX1	SSX0

(b)缓存器设定：卷动区块垂直起始地址(Y1)

REG[09h]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	--	--	SSY5	SSY4	SSY3	SSY2	SSY1	SSY0

(c)缓存器设定：卷动区块水平卷动范围(X2)

REG[0Ah]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	--	--	--	SRX4	SRX3	SRX2	SRX1	SRX0

(d)缓存器设定：卷动区块垂直卷动范围(Y2)

REG[0Bh]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	--	--	SRY5	SRY4	SRY3	SRY2	SRY1	SRY0

9-3 卷动速度设定

(a)REG-[0Dh]之 Bit[7：4] 设定自动卷动的速度。

(b)REG-[0Dh]之 Bit[3：0] 设定每次位移的点数(每次卷动像素 1~16) 。

缓存器设定：

REG[0Dh]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	SPD3	SDP2	SPD1	SPD0	STP3	STP2	STP1	STP0

Bit	Description	Default	Access
7-4	设定自动卷动的速度： 0 0 0 0 ：最快(8 个 Frame) . . . 1 1 1 1 ：最慢(128 个 Frame)	0h	R/W
3-0	设定每次位移的像素： 0 0 0 0 ：最小 1 个位移的像素 . .	0h	R/W

	. 1 1 1 1 ：最大 16 个位移的像素		

9-4 程序范例

(a)由上往下卷动(垂直卷动)

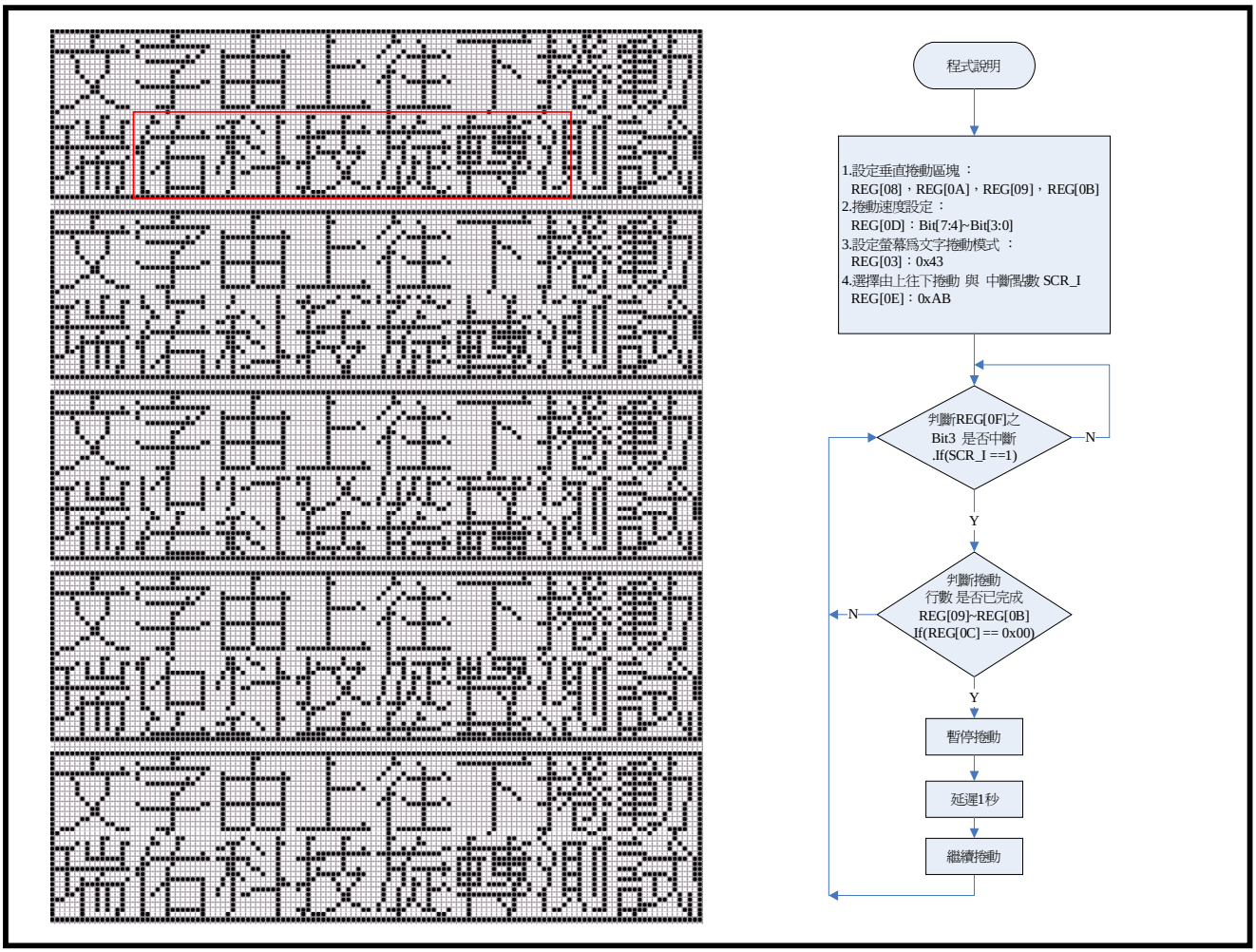


图 9-2：垂直卷动(由上往下)

以下为 图 9-2 范例程序：

```
LCD_Clear_LCD                          // 清除所有显示
LCD_CmdWrite(0x0C,0x00);               // 清除卷动向量缓存器
LCD_CmdWrite(0x0E,0x00);               // 清除卷动功能缓存器
LCD_CmdWrite(0x01,0x02);               //  LCD 显示开启
LCD_CmdWrite(0x04,0x74);               // 关闭光标显示
```

```
LCD_CmdWrite(0x03,0x03);                          // 正常文字显示模式
GotoXY(0,0);                                      // 设定光标地址
PrintStr_SPI3("文字由上往下卷动瑞佑科技旋转测试",1);   // 显示中文字符串
                                                  // 设定卷动区块
LCD_CmdWrite(0x08,2);                             // 设定卷动范围水平起始地址
LCD_CmdWrite(0x09,16);                            // 设定卷动范围垂直起始地址
LCD_CmdWrite(0x0A,10);                            // 设定卷动范围水平结束地址
LCD_CmdWrite(0x0B,31);                            // 设定卷动范围垂直结束地址
LCD_CmdWrite(0x03,0x43);                          // 设定文字卷动功能模式
LCD_CmdWrite(0x0D,0x30);                          // 卷动速度设定
LCD_CmdWrite(0x0E,0xAB);                          // 由上往下卷动时，每位移 8 点，SCR_I
                                                  // 产生中断


While(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)       // 侦测 SCR_I 是否为”1”(中断)
  {
    if(LCD_CmdRead_SPI3(0x0C) == 0x00)            // 判断是否已经卷动 16 点完成
    {
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) & 0xFE);   // 暂停卷动
      delay(1000);                                // 延迟 1 秒
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) | 0x01);   // 继续卷动
    }
  }
}
```

(b)由下往上卷动(垂直卷动)：

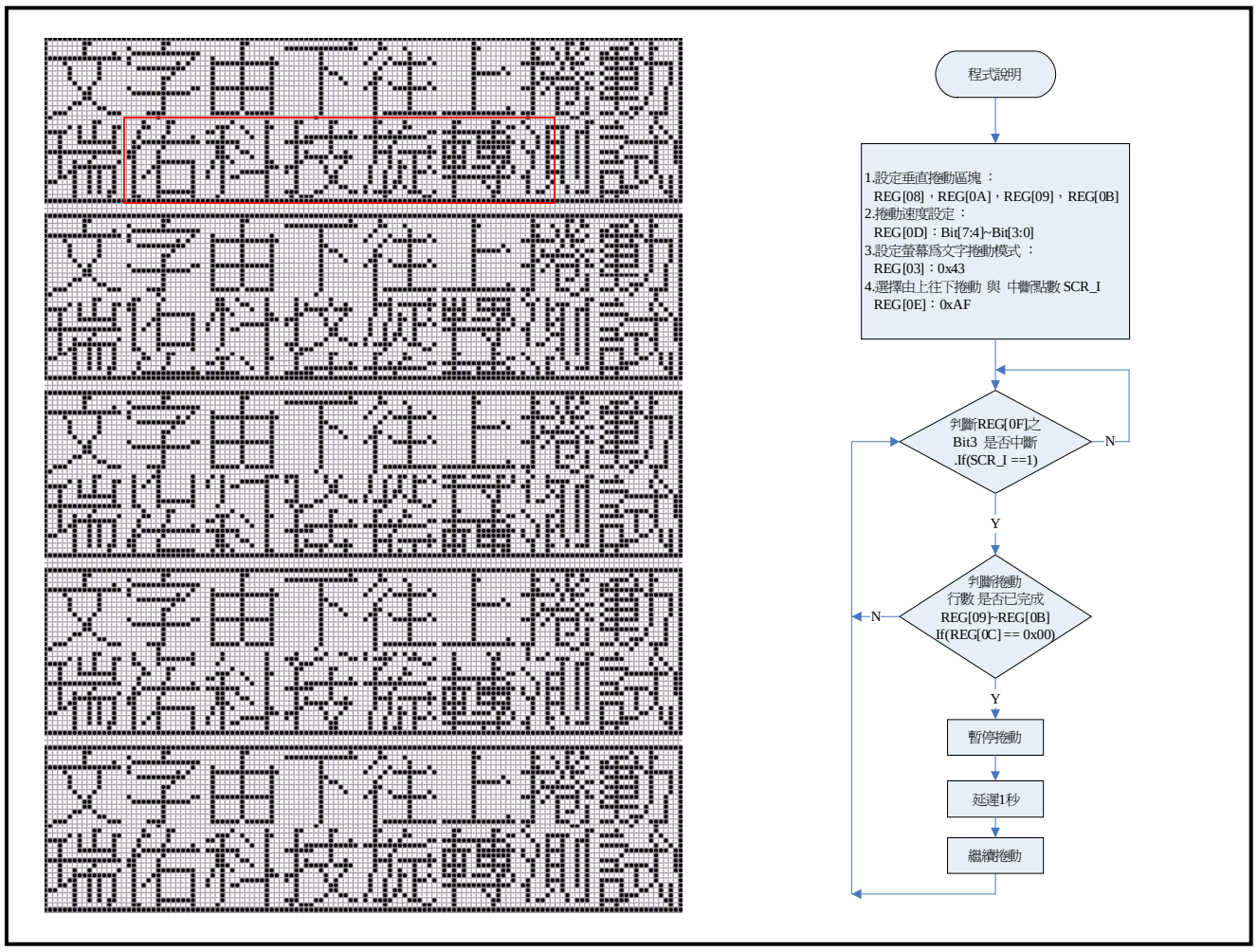


图 9-3： 垂直卷动(由下往上)

以下为 图 9-3 范例程序：

```
LCD_Clear_LCD                                    // 清除所有显示
LCD_CmdWrite(0x0C,0x00);                         // 清除卷动向量缓存器
LCD_CmdWrite(0x0E,0x00);                         // 清除卷动功能缓存器
LCD_CmdWrite(0x01,0x02);                         //  LCD 显示开启
LCD_CmdWrite(0x04,0x74);                         // 关闭光标显示
LCD_CmdWrite(0x03,0x03);                         // 正常文字显示模式
GotoXY(0,0);                                     // 设定光标地址
PrintStr_SPI3("文字由下往下卷动瑞佑科技旋转测试",1);  // 显示中文字符串
                                                 // 设定卷动区块
LCD_CmdWrite(0x08,2);                            // 设定卷动范围水平起始地址
```

```
LCD_CmdWrite(0x09,16);                          // 设定卷动范围垂直起始地址
LCD_CmdWrite(0x0A,10);                          // 设定卷动范围水平结束地址
LCD_CmdWrite(0x0B,31);                          // 设定卷动范围垂直结束地址
LCD_CmdWrite(0x03,0x43);                        // 设定文字卷动功能模式
LCD_CmdWrite(0x0D,0x30);                        // 卷动速度设定
LCD_CmdWrite(0x0E,0xAF);                        // 由下往上卷动时，每位移 8 点，SCR_I 产
                                                // 生中断


While(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)     // 侦测 SCR_I 是否为”1”(中断)
  {
    if(LCD_CmdRead_SPI3(0x0C) == 0x00)          // 判断是否已经卷动 16 点完成
    {
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) & 0xFE);   // 暂停卷动
      delay(1000);                              // 延迟 1 秒
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) | 0x01);   // 继续卷动
    }
  }
}
```

(c)由左往右卷动(水平卷动)：

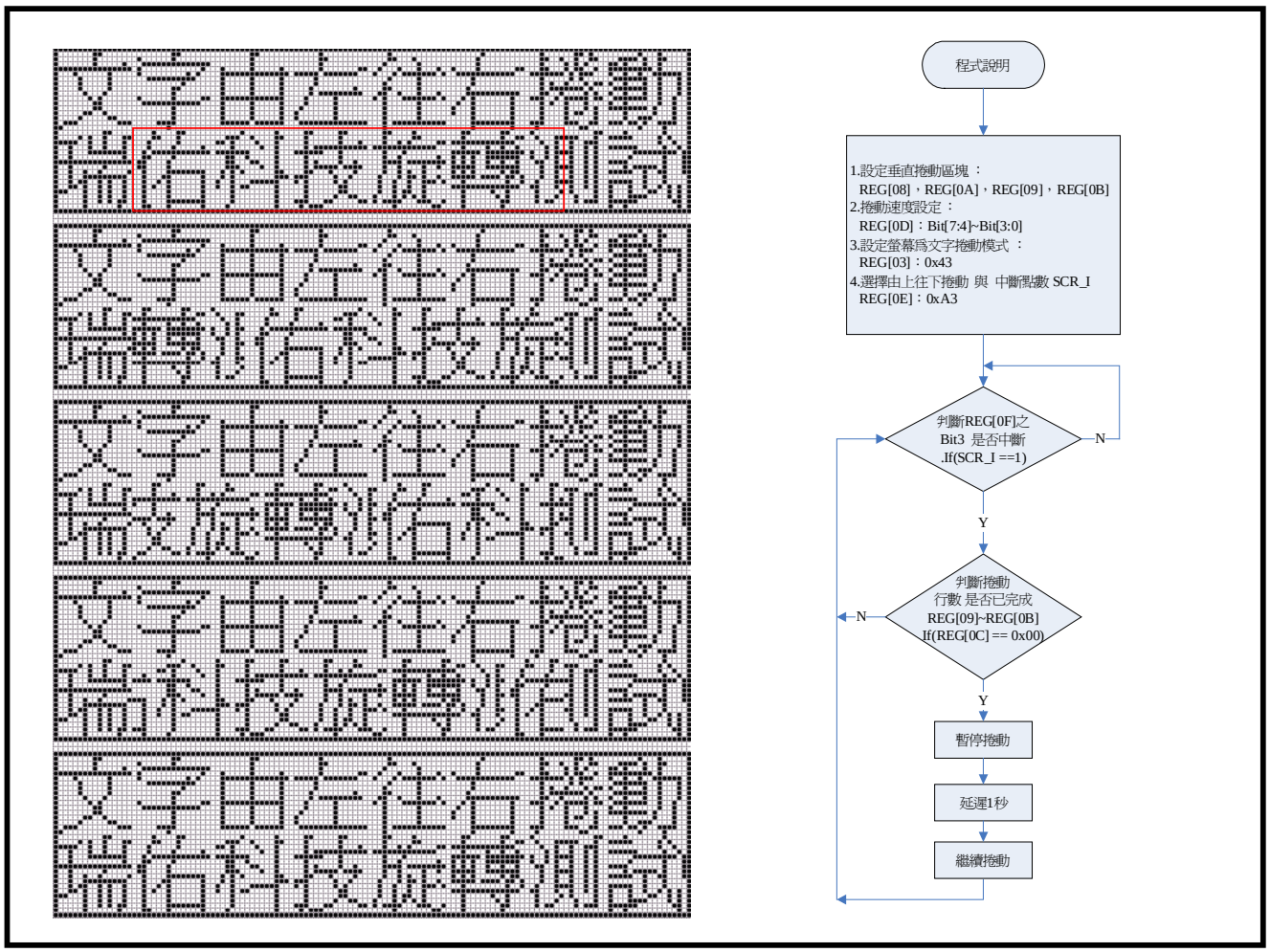


图 9-4： 水平卷动(由左往右)

以下为 图 9-4 范例程序：

```
LCD_Clear_LCD                                          // 清除所有显示
LCD_CmdWrite(0x0C,0x00);                               // 清除卷动向量缓存器
LCD_CmdWrite(0x0E,0x00);                               // 清除卷动功能缓存器
LCD_CmdWrite(0x01,0x02);                               //  LCD 显示开启
LCD_CmdWrite(0x04,0x74);                               // 关闭光标显示
LCD_CmdWrite(0x03,0x03);                               // 正常文字显示模式
GotoXY(0,0);                                           // 设定光标地址
PrintStr_SPI3("文字由左往右卷动瑞佑科技旋转测试",1);    // 显示中文字符串
                                                       // 设定卷动区块
LCD_CmdWrite(0x08,2);                                  // 设定卷动范围水平起始地址
```

```
LCD_CmdWrite(0x09,16);                          // 设定卷动范围垂直起始地址
LCD_CmdWrite(0x0A,10);                          // 设定卷动范围水平结束地址
LCD_CmdWrite(0x0B,31);                          // 设定卷动范围垂直结束地址
LCD_CmdWrite(0x03,0x43);                        // 设定文字卷动功能模式
LCD_CmdWrite(0x0D,0x30);                        // 卷动速度设定
LCD_CmdWrite(0x0E,0xA3);                        // 由左往右卷动时，每位移 8 点，SCR_I 产
                                                // 生中断


While(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)     // 侦测 SCR_I 是否为”1”(中断)
  {
    if(LCD_CmdRead_SPI3(0x0C) == 0x00)          // 判断是否已经卷动 16 点完成
    {
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) & 0xFE);   // 暂停卷动
      delay(1000);                              // 延迟 1 秒
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) | 0x01);   // 继续卷动
    }
  }
}
```

(d)由右往左卷动(水平卷动)：

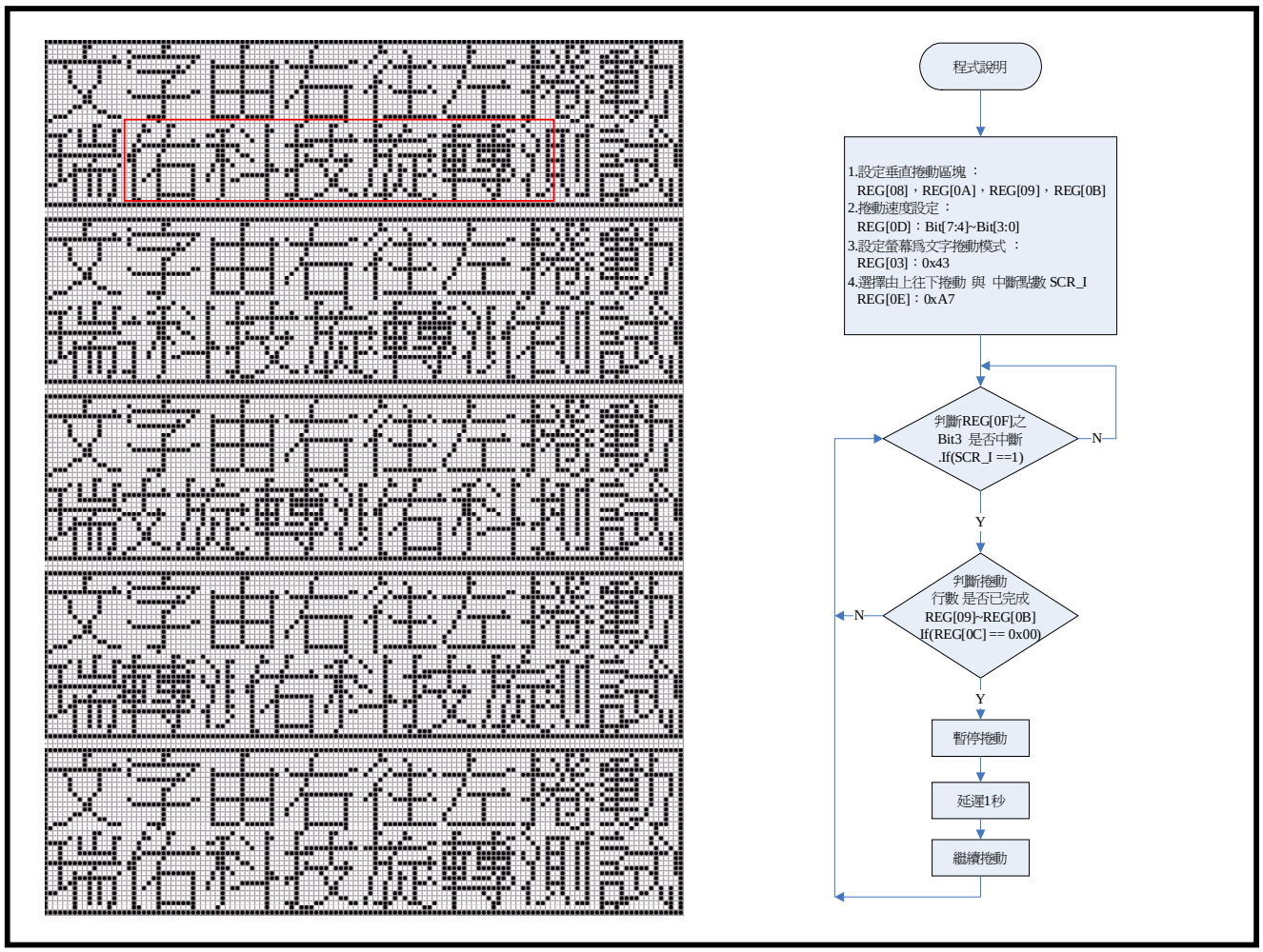


图 9-5： 水平卷动(由右往左)

以下为 图 9-5 范例程序：

```
LCD_Clear_LCD                                   // 清除所有显示
LCD_CmdWrite(0x0C,0x00);                        // 清除卷动向量缓存器
LCD_CmdWrite(0x0E,0x00);                        // 清除卷动功能缓存器
LCD_CmdWrite(0x01,0x02);                        //  LCD 显示开启
LCD_CmdWrite(0x04,0x74);                        // 关闭光标显示
LCD_CmdWrite(0x03,0x03);                        // 正常文字显示模式
GotoXY(0,0);                                    // 设定光标地址
PrintStr_SPI3("文字由右往左卷动瑞佑科技旋转测试",1);  // 显示中文字符串
                                                // 设定卷动区块
LCD_CmdWrite(0x08,2);                           // 设定卷动范围水平起始地址
```

```
LCD_CmdWrite(0x09,16);                          // 设定卷动范围垂直起始地址
LCD_CmdWrite(0x0A,10);                          // 设定卷动范围水平结束地址
LCD_CmdWrite(0x0B,31);                          // 设定卷动范围垂直结束地址
LCD_CmdWrite(0x03,0x43);                        // 设定文字卷动功能模式
LCD_CmdWrite(0x0D,0x30);                        // 卷动速度设定
LCD_CmdWrite(0x0E,0xA7);                        // 由右往左卷动时，每位移 8 点，SCR_I 产生中
                                                // 断


While(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)     // 侦测 SCR_I 是否为”1”(中断)
  {
    if(LCD_CmdRead_SPI3(0x0C) == 0x00)          // 判断是否已经卷动 16 点完成
    {
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) & 0xFE);   // 暂停卷动
      delay(1000);                              // 延迟 1 秒
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) | 0x01);   // 继续卷动
    }
  }
}
```

10. 文字模式移动

RA8815 提供 LCD 显示画面可经由缓存器的设定指定一区块，使硬件自动做屏幕画面移动的效果，其中缓存器的功能设定主要包刮移动画面的方向与速度，及每次移动画面的像数与移动的范围，以下针对实际应用时程序撰写方式作一说明。

10-1 移动方向

可由缓存器 REG-[0Eh]之 BIT[3：2]设定，选择四种不同的卷动方式

缓存器设定：

	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
REG[0Eh]	SCR_IMD1	SCR_IMD0	AUTO_SCR	SBUF	**SCR_MD1**	**SCR_MD0**	SCR_INTEN	SCR_EN

Bit	Description	Default	Access
3-2	设定移动方向： 0 0 ：由左往右移动(水平移动) 0 1 ：由右往左移动(水平移动) 1 0 ：由上往下卷动(垂直移动) 1 1 ：由下往上卷动(垂直移动)	0h	R/W

10-2 移动范围设定

设定 REG-[08h]、REG-[09h]、REG-[0Ah]、REG-[0Bh]四个缓存器，可指定一区块作为移动的范围。

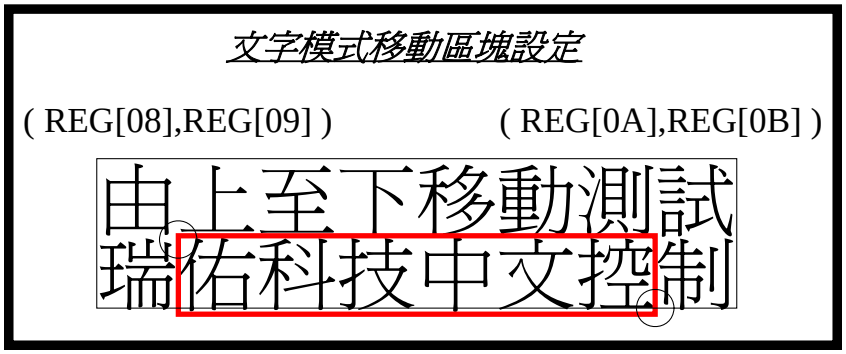


图 10-1： 设定移动范围

(a)缓存器设定：移动区块水平起始地址(X1)

REG[08h]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	--	--	--	SSX4	SSX3	SSX2	SSX1	SSX0

(b)缓存器设定：移动区块垂直起始地址(Y1)

REG[09h]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	--	--	SSY5	SSY4	SSY3	SSY2	SSY1	SSY0

(c)缓存器设定：移动区块水平移动范围(X2)

REG[0Ah]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	--	--	--	SRX4	SRX3	SRX2	SRX1	SRX0

(d)缓存器设定：移动区块垂直移动范围(Y2)

REG[0Bh]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	--	--	SRY5	SRY4	SRY3	SRY2	SRY1	SRY0

10-3 移动速度设定

(a)REG-[0Dh]之 Bit[7：4]设定自动移动的速度。

(b)REG-[0Dh]之 Bit[3：0]设定每次位移的点数(每次移动像素 1~16) 。

缓存器设定：

REG[0Dh]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	SPD3	SDP2	SPD1	SPD0	STP3	STP2	STP1	STP0

Bit	Description	Default	Access
7-4	设定自动移动的速度： 0 0 0 0 ：最快(8 个 Frame) . . . 1 1 1 1 ：最慢(128 个 Frame)	0h	R/W
3-0	设定每次位移的像素： 0 0 0 0 ：最小 1 个位移的像素 . .	0h	R/W

	. 1 1 1 1 ：最大 16 个位移的像素		

10-4 程序范例

(a)由上往下移动(垂直移动):

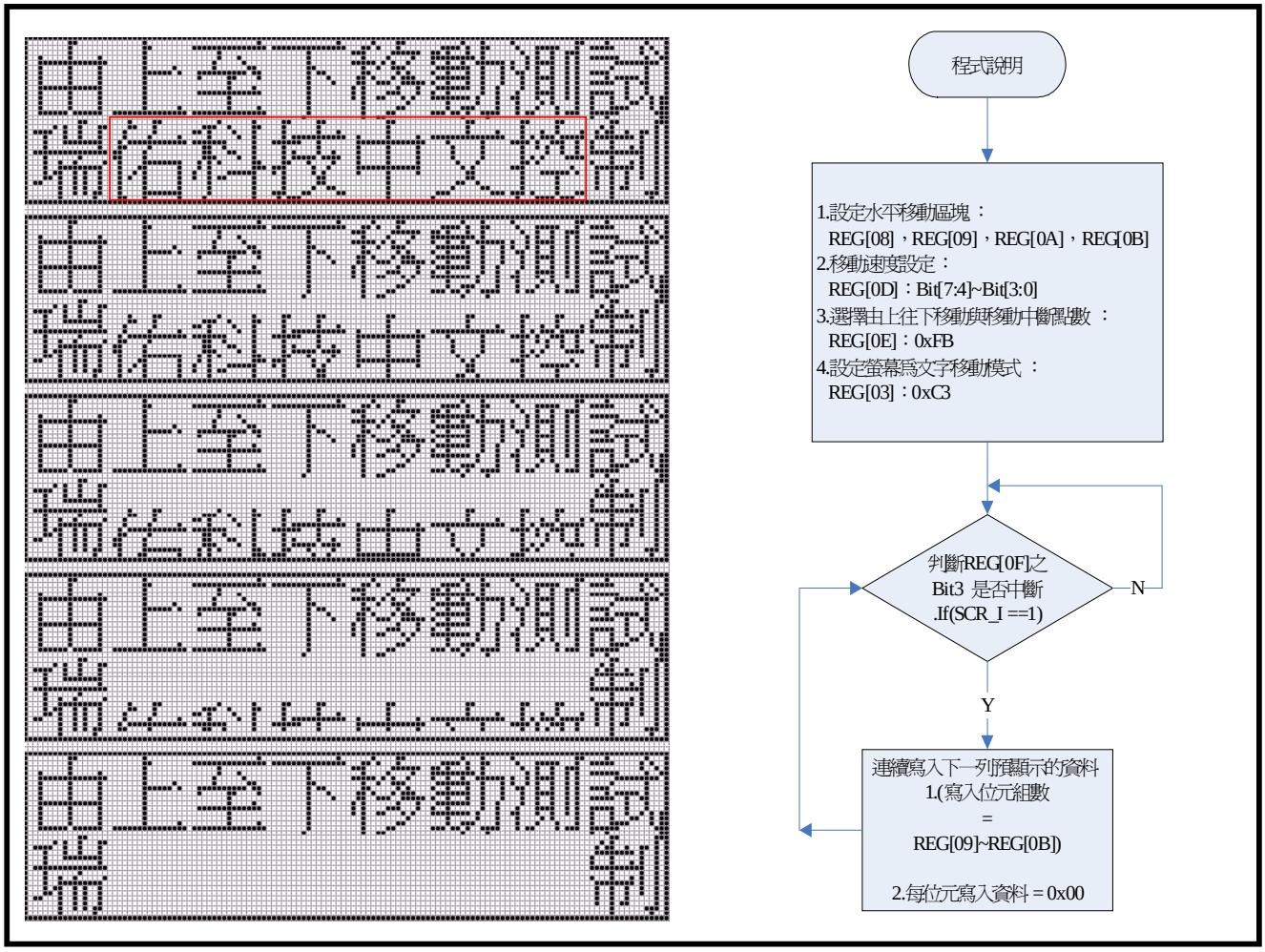


图 10-2： 垂直移动(由上往下)

以下为 图 10-2 范例程序：

```
LCD_Clear_LCD                          // 清除所有显示
LCD_CmdWrite(0x0C,0x00);               // 清除移动向量缓存器
LCD_CmdWrite(0x0E,0x00);               // 清除移动功能缓存器
LCD_CmdWrite(0x03,0x03);               // 文字模式
GotoXY(0,0);                           // 设定光标地址
```

```
PrintStr("由上往下移动测试瑞佑科技中文控制",1);        // 显示中文字符串
                                                    // 设定移动区块
LCD_CmdWrite(0x08,0x02);                            // 设定移动范围水平起始地址
LCD_CmdWrite(0x09,0x10);                            // 设定移动范围垂直起始地址
LCD_CmdWrite(0x0A,0x0B);                            // 设定移动范围水平结束地址
LCD_CmdWrite(0x0B,0x1F);                            // 设定移动范围垂直结束地址
LCD_CmdWrite(0x0D,0x50);                            // 卷动速度设定
LCD_CmdWrite(0x0E,0xFB);                            // 启动由上而下移动功能，每移动 16 行点数
                                                    // 时就发出 SCR_I 中断

LCD_CmdWrite(0x03,0xC3);                            // 设定文字移动功能模式

While(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)         // 侦测 SCR_I 是否为”1”(中断)
  {
    for(i=0 ;i<12 ;i++)
        LCD_DataWrite(0x20);                        // 连续写入空格符
    LCD_CmdWrite(0x0F,LCD_CmdRead(0x0F) &           // 清除 SCR_I 中断位为”0”
0xFB);
  }
}
```

(b)由下往上移动(垂直移动)：

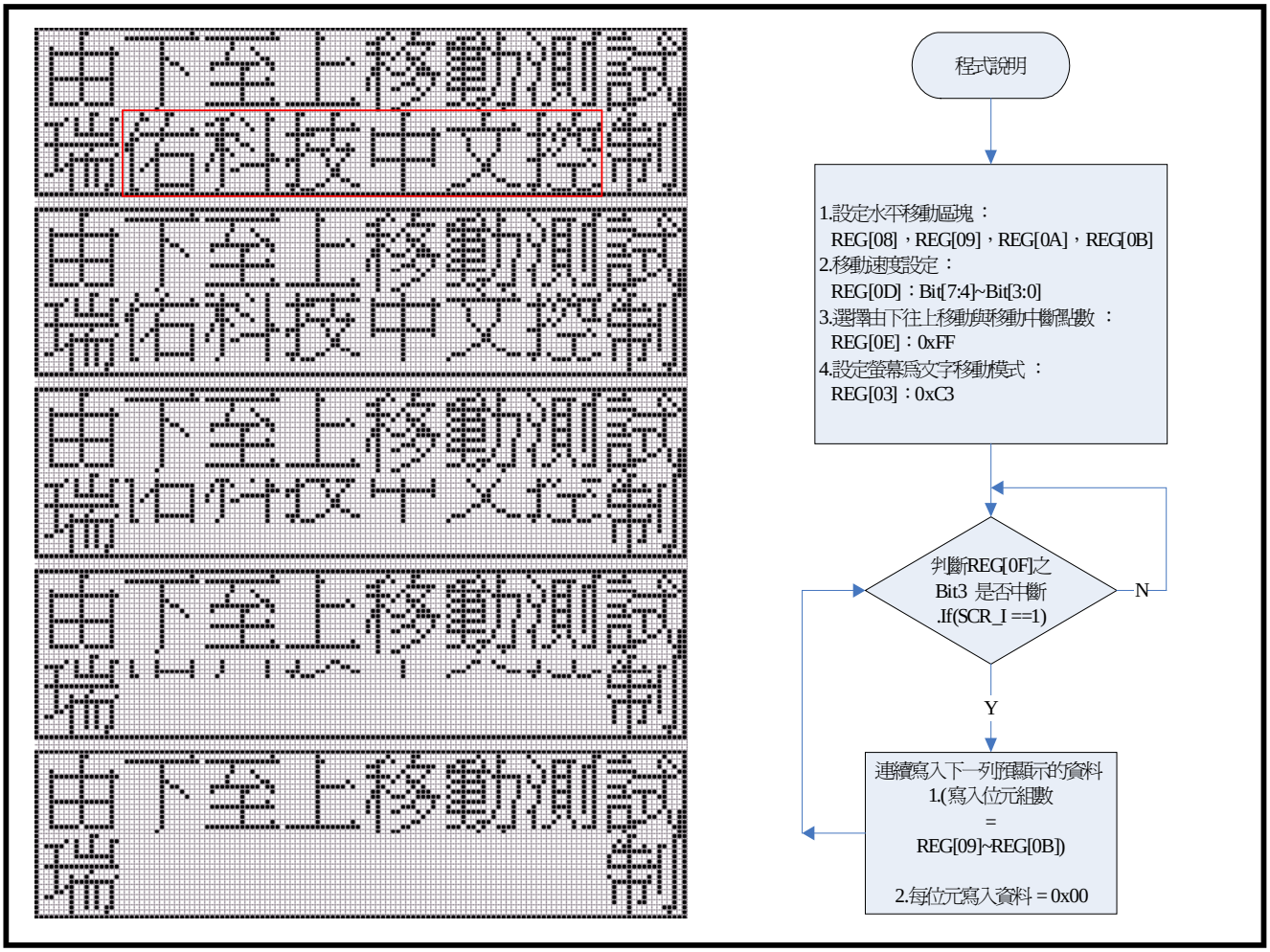


图 10-3： 垂直移动(由下往上)

以下为 图 10-3 范例程序：

```
LCD_Clear_LCD                                   // 清除所有显示
LCD_CmdWrite(0x0C,0x00);                        // 清除移动向量缓存器
LCD_CmdWrite(0x0E,0x00);                        // 清除移动功能缓存器
LCD_CmdWrite(0x03,0x03);                        // 文字模式
GotoXY(0,0);                                    // 设定光标地址
PrintStr("由下往上移动测试瑞佑科技中文控制",1);      // 显示字符串
                                                // 设定移动区块
LCD_CmdWrite(0x08,0x02);                        // 设定移动范围水平起始地址
LCD_CmdWrite(0x09,0x10);                        // 设定移动范围垂直起始地址
LCD_CmdWrite(0x0a,0x0b);                        // 设定移动范围水平结束地址
```

```
LCD_CmdWrite(0x0b,0x1f);                    // 设定移动范围垂直结束地址
LCD_CmdWrite(0x0d,0x70);                    // 卷动速度设定
LCD_CmdWrite(0x03,0xc3);                    // 设定文字移动功能模式
LCD_CmdWrite(0x0E,0xff);                    // 启动由下而上移动功能，每移动 16 行点数
                                            // 时就发出 SCR_I 中断


While(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)
  {
    for(i=0 ;i<12 ;i++)                     // 检查移动中断
      LCD_DataWrite(0x20);                  // 若是，则连续写入一列空格符…
    LCD_CmdWrite(0x0F,LCD_CmdRead(0x0F) &   // 清除 SCR_I 中断位为”0”
0xFB);
  }
}
```

(c)由左往右移动(水平移动)：

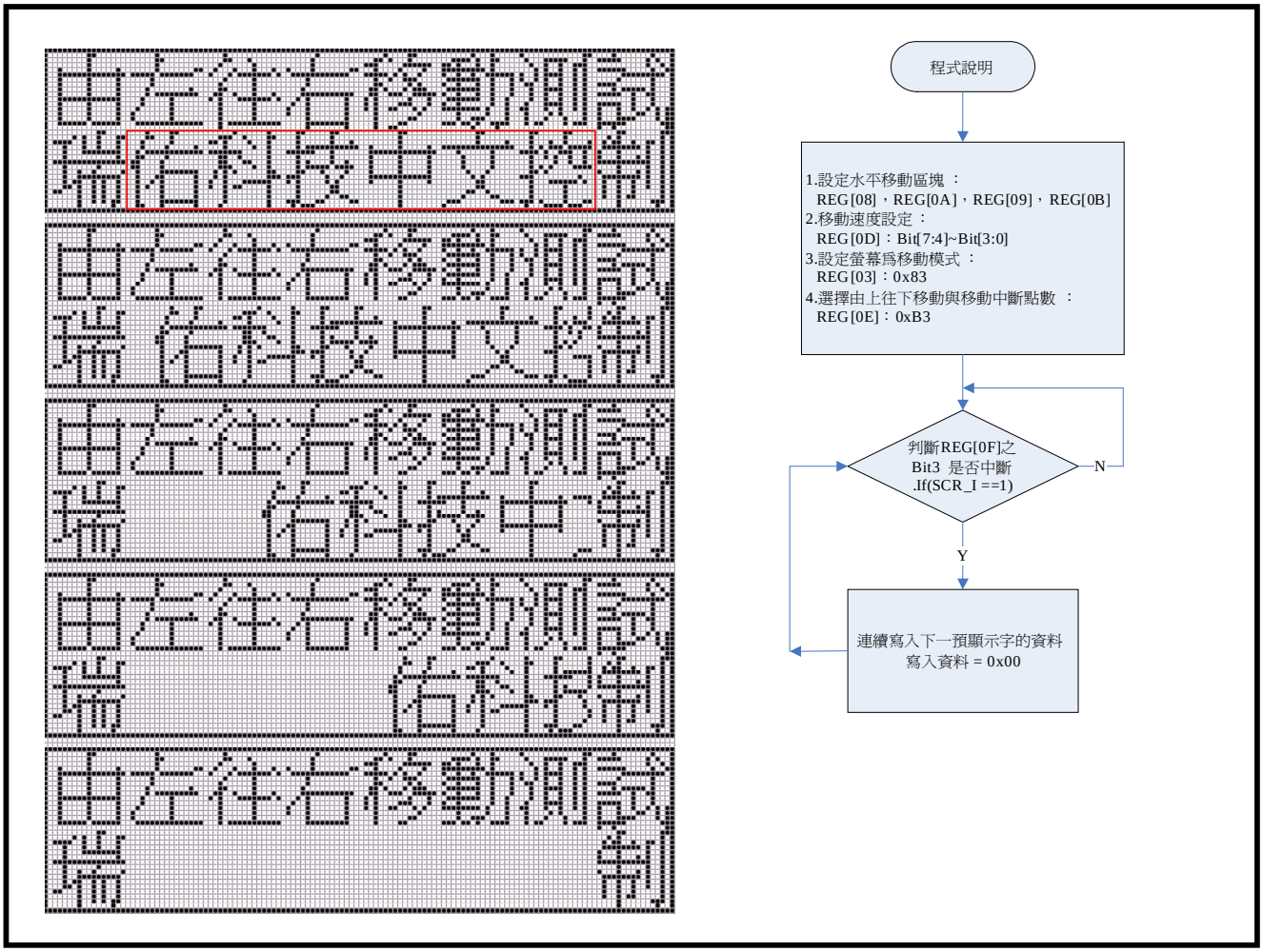


图 10-4： 水平移动(由左往右)

以下为 图 10-4 范例程序：

```
LCD_CmdWrite(0x0C,0x00);                    // 清除移动向量缓存器
LCD_CmdWrite(0x0E,0x00);                    // 清除移动功能缓存器
LCD_CmdWrite(0x03,0x03);                    // 文字模式

LCD_Clear_LCD                               // 清除所有显示
GotoXY(0,0);                                // 设定光标地址
PrintStr("由左往右移动测试瑞佑科技中文控制",1);  // 显示字符串
                                            // 设定移动区块
LCD_CmdWrite(0x08,0x02);                    // 设定移动范围水平起始地址
LCD_CmdWrite(0x09,0x10);                    // 设定移动范围垂直起始地址
```

```
LCD_CmdWrite(0x0a,0x0b);                    // 设定移动范围水平结束地址
LCD_CmdWrite(0x0b,0x1f);                    // 设定移动范围垂直结束地址
LCD_CmdWrite(0x0D,0x30);                    // 卷动速度设定
LCD_CmdWrite(0x03,0x83);                    // 设定文字移动功能模式
LCD_CmdWrite(0x0E,0xB3);                    // 启动由左往右移动功能，每移动 8 点像数时
                                            // 就发出一次 SCR_I 中断
while(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)  // 侦测 SCR_I 是否为”1”(中断)
  {
    LCD_DataWrite(0x20);                    // 若是中断产生，就写入空格符
    LCD_CmdWrite(0x0F,LCD_CmdRead(0x0F)     // 清除 SCR_I 中断位为”0”
& 0xFB);
  }
}
```

(d)由右往左移动(水平移动)：

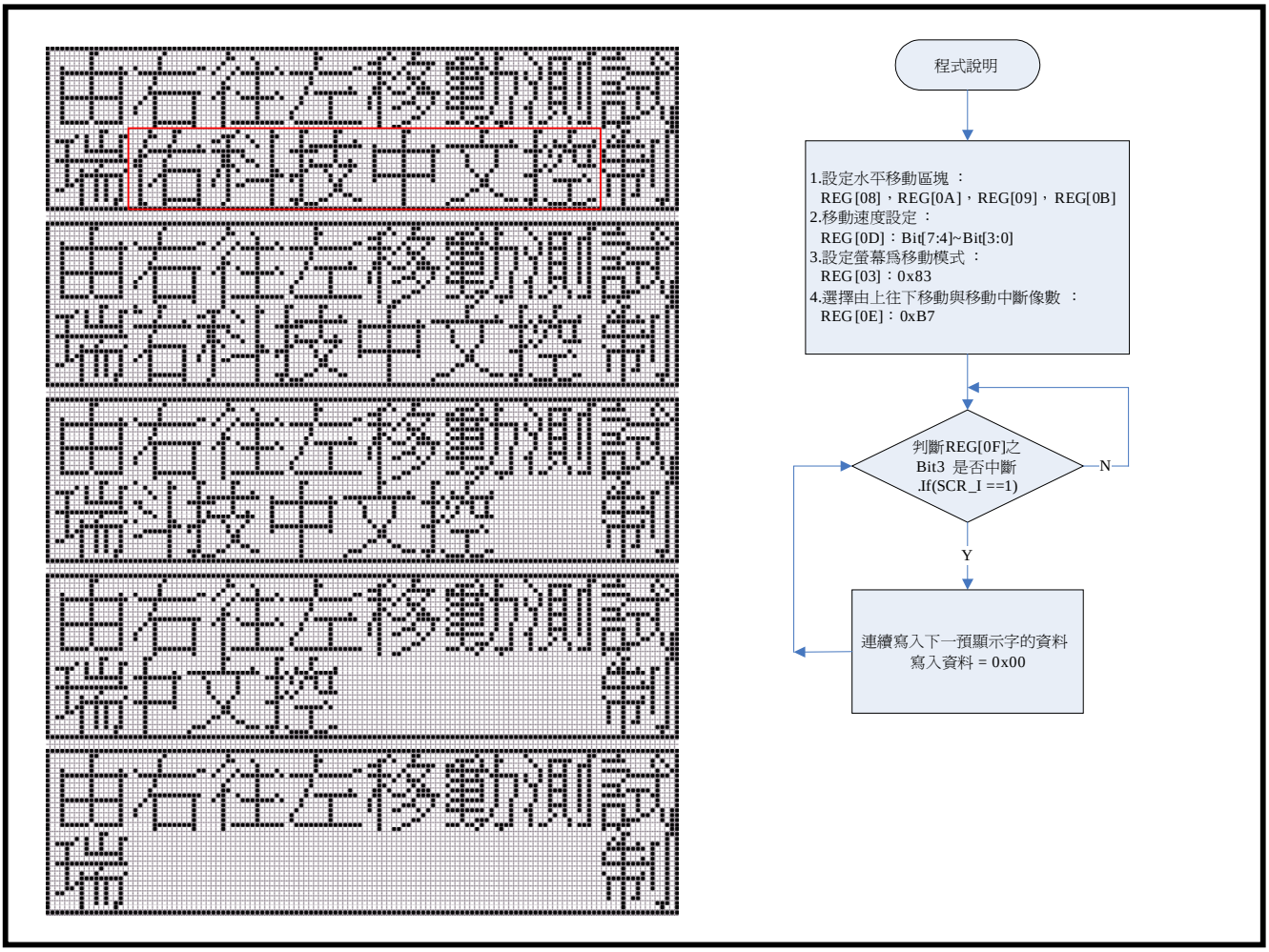


图 10-5： 水平移动(由右往左)

以下为 图 10-5 范例程序：

```
LCD_CmdWrite(0x0E,0x00);                    // 清除移动向量缓存器
LCD_CmdWrite(0x0C,0x00);                    // 清除移动功能缓存器
LCD_CmdWrite(0x03,0x00);                    // 文字模式
LCD_Clear_LCD                               // 清除所有显示

GotoXY(0,0)                                 // 设定光标地址
PrintStr ("由右往左移动测试瑞佑科技中文控制",1);  // 显示字符串
                                            // 设定移动区块
LCD_CmdWrite (0x08,0x02);                   // 设定移动范围水平起始地址
LCD_CmdWrite (0x09,0x10);                   // 设定移动范围垂直起始地址
```

```
LCD_CmdWrite (0x0a,0x0b);                    // 设定移动范围水平结束地址
LCD_CmdWrite (0x0b,0x1f);                    // 设定移动范围垂直结束地址

LCD_CmdWrite(0x0D,0x30);                     // 卷动速度设定
LCD_CmdWrite(0x03,0x83);                     // 设定文字移动功能模式
LCD_CmdWrite(0x0E,0xb7);                     // 启动由右往左移动功能，每移动 8 点数时
                                             // 就发出 SCR_I 中断
while(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)  // 侦测 SCR_I 是否为”1”(中断)
  {
    LCD_DataWrite(0x20);                     // 若是中断产生，就写入空格符
    LCD_CmdWrite(0x0F,LCD_CmdRead(0x0F)      // 清除 SCR_I 中断位为”0”
& 0xFB);
  }
}
```

RA8815 提供 LCD 显示画面可经由缓存器的设定指定一区块，使硬件自动做屏幕画面卷动的效果，其中缓存器的功能设定主要包刮卷动画面的方向与速度，及每次卷动画面的像数与卷动的范围，以下针对实际应用时程序撰写方式作一说明。

11. 绘图模式卷动

11-1 卷动方向

RA8815 可由缓存器 REG-[0Eh]之 BIT[3：2] 设定，选择四种不同的卷动方式。

缓存器设定：

	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
REG[0Eh]	SCR_IMD1	SCR_IMD0	AUTO_SCR	SBUF	**SCR_MD1**	**SCR_MD0**	SCR_INTEN	SCR_EN

Bit	Description	Default	Access
3-2	设定移动方向： 0 0 ：由左往右卷动(水平卷动) 0 1 ：由右往左卷动(水平卷动) 1 0 ：由上往下卷动(垂直卷动) 1 1 ：由下往上卷动(垂直卷动)	0h	R/W

11-2 卷动范围设定

设定 REG-[08h]、REG-[09h]、REG-[0Ah]、REG-[0Bh]四个缓存器，可指定一区块作为卷动的范围。

图 11-1： 卷动范围设定

(a)缓存器设定：卷动区块水平起始地址(X1)

	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
REG[08h]	--	--	--	SSX4	SSX3	SSX2	SSX1	SSX0

(b)缓存器设定：卷动区块垂直起始地址(Y1)

	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
REG[09h]								

	--	--	SSY5	SSY4	SSY3	SSY2	SSY1	SSY0

(c)缓存器设定：卷动区块水平卷动范围(X2)

REG[0Ah]	***DB7***	***DB6***	***DB5***	***DB4***	***DB3***	***DB2***	***DB1***	***DB0***
	--	--	--	SRX4	SRX3	SRX2	SRX1	SRX0

(d)缓存器设定：卷动区块垂直卷动范围(Y2)

REG[0Bh]	***DB7***	***DB6***	***DB5***	***DB4***	***DB3***	***DB2***	***DB1***	***DB0***
	--	--	SRY5	SRY4	SRY3	SRY2	SRY1	SRY0

11-3 卷动速度设定

(a)REG-[0Dh]之 Bit[7：4] 设定自动卷动的速度。

(b)REG-[0Dh]之 Bit[3：0] 设定每次位移的点数(每次卷动像素 1~16) 。

缓存器设定：

REG[0Dh]	***DB7***	***DB6***	***DB5***	***DB4***	***DB3***	***DB2***	***DB1***	***DB0***
	SPD3	SDP2	SPD1	SPD0	STP3	STP2	STP1	STP0

Bit	Description	Default	Access
7-4	设定自动卷动的速度： 0 0 0 0 ：最快(8 个 Frame) . . . 1 1 1 1 ：最慢(128 个 Frame)	0h	R/W
3-0	设定每次位移的像素： 0 0 0 0 ：最小 1 个位移的像素 . . . 1 1 1 1 ：最大 16 个位移的像素	0h	R/W

11-4 程序范例

(a)由上往下卷动(垂直卷动)

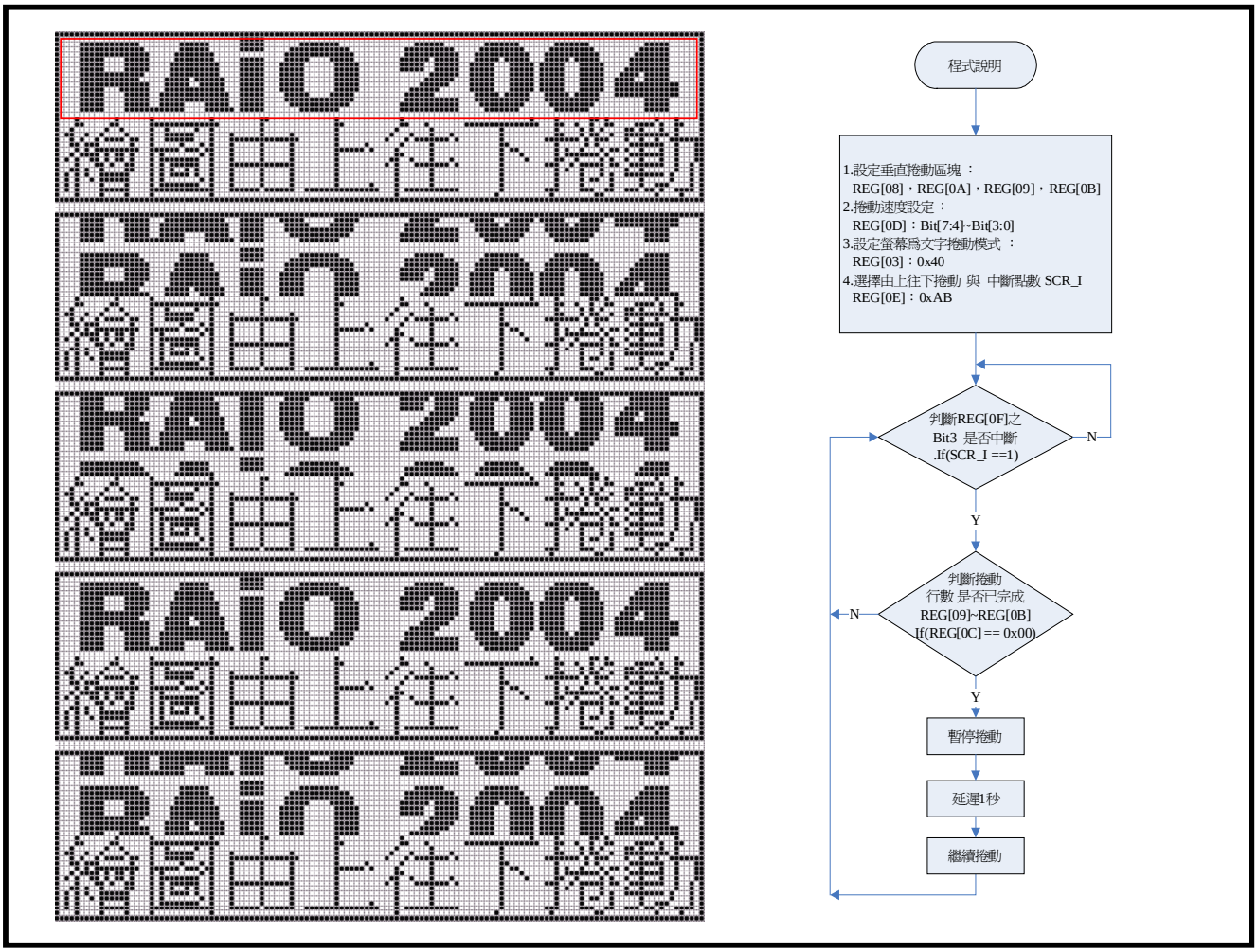


图 11-2： 垂直卷动(由上往下)

以下为 图 11-2 范例程序：

```
LCD_Clear_LCD                                   // 清除所有显示
LCD_CmdWrite (0x03,0x00);                       // 设定绘图显示模式
LCD_CmdWrite (0x04,0x74);                       // 关闭游标
LCD_CmdWrite (0x01,0x02);                       // 开启 LCD 显示
GotoXY (0,0);                                   // 设定光标地址
for(length=0; length< 256 ; length++)
    LCD_DataWrite (~(DataString2[length]));     // 写入 RAiO 2004 图形(128 * 16)
```

```
LCD_CmdWrite (0x03,0x03);                    // 设定文字显示模式
GotoXY (0,16);                               // 设定光标地址
PrintStr ("绘图由上往下卷动",1);              // 写入中文字符串
LCD_CmdWrite(0x08,0x00);                     // 设定卷动范围水平起始地址
LCD_CmdWrite(0x09,0x00);                     // 设定卷动范围垂直起始地址
LCD_CmdWrite(0x0A,0x0f);                     // 设定卷动范围水平结束地址
LCD_CmdWrite(0x0B,0x0f);                     // 设定卷动范围垂直结束地址

LCD_CmdWrite(0x0D,0x30);                     // 卷动速度设定
LCD_CmdWrite(0x03,0x40);                     // 设定绘图卷动功能模式
LCD_CmdWrite(0x0E,0xAB);                     // 由上往下卷动时，每位移 8 点，SCR_I 产生
                                             // 中断


While(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)  // 侦测 SCR_I 是否为”1”(中断)
  {
    if(LCD_CmdRead_SPI3(0x0C) == 0x00)       // 判断是否已经卷动 16 点完成
    {
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) & 0xFE);  // 暂停卷动
      delay(1000);                           // 延迟 1 秒
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) | 0x01);  // 继续卷动
    }
  }
}
```

(b)由下往上卷动(垂直卷动)：

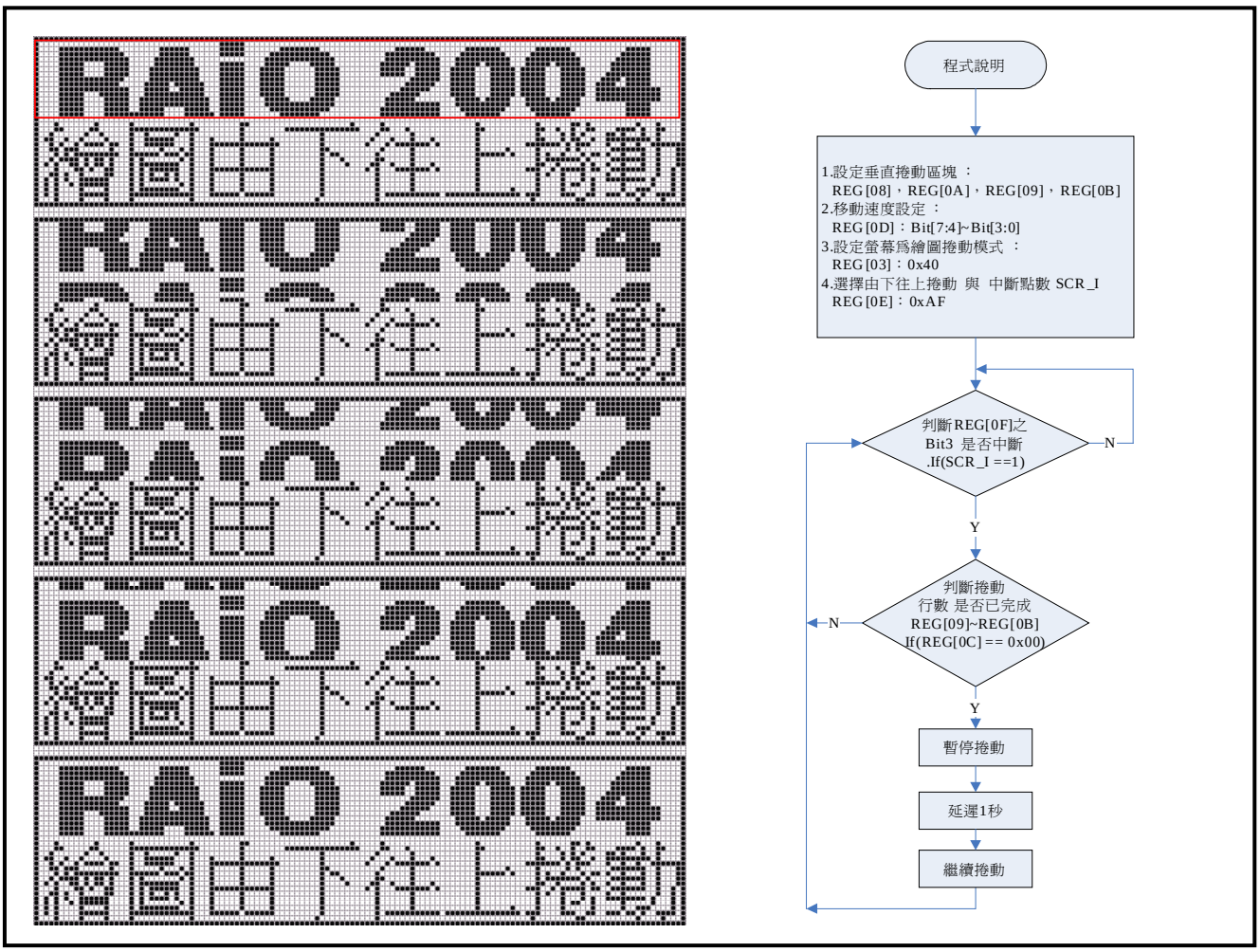


图 11-3： 垂直卷动(由下往上)

以下为 图 11-3 范例程序：

```
LCD_Clear_LCD                                   // 清除所有显示
LCD_CmdWrite (0x03,0x00);                       // 设定绘图显示模式
LCD_CmdWrite (0x04,0x74);                       // 关闭游标
LCD_CmdWrite (0x01,0x02);                       // 开启 LCD 显示
GotoXY (0,0);                                   // 设定光标地址
for(length=0; length< 256 ; length++)
    LCD_DataWrite (~(DataString2[length]));     // 写入 RAiO 2004  图形(128 * 16)

LCD_CmdWrite (0x03,0x03);                       // 设定文字显示模式
GotoXY (0,16);                                  // 设定光标地址
```

```
PrintStr ("绘图由下往上卷动",1);                    // 写入中文字符串
LCD_CmdWrite(0x08,0x00);                            // 设定卷动范围水平起始地址
LCD_CmdWrite(0x09,0x00);                            // 设定卷动范围垂直起始地址
LCD_CmdWrite(0x0A,0x0f);                            // 设定卷动范围水平结束地址
LCD_CmdWrite(0x0B,0x0f);                            // 设定卷动范围垂直结束地址

LCD_CmdWrite(0x0D,0x30);                            // 卷动速度设定
LCD_CmdWrite(0x03,0x40);                            // 设定绘图卷动功能模式
LCD_CmdWrite(0x0E,0xAF);                            // 由下往上卷动时，每位移 8 点，SCR_I 产生
                                                    // 中断

While(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)         // 侦测 SCR_I 是否为”1”(中断)
  {
    if(LCD_CmdRead_SPI3(0x0C) == 0x00)              // 判断是否已经卷动 16 点完成
    {
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) & 0xFE);  // 暂停卷动
      delay(1000);                                  // 延迟 1 秒
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) | 0x01);  // 继续卷动
    }
  }
}
```

(c)由左往右卷动(水平卷动)：

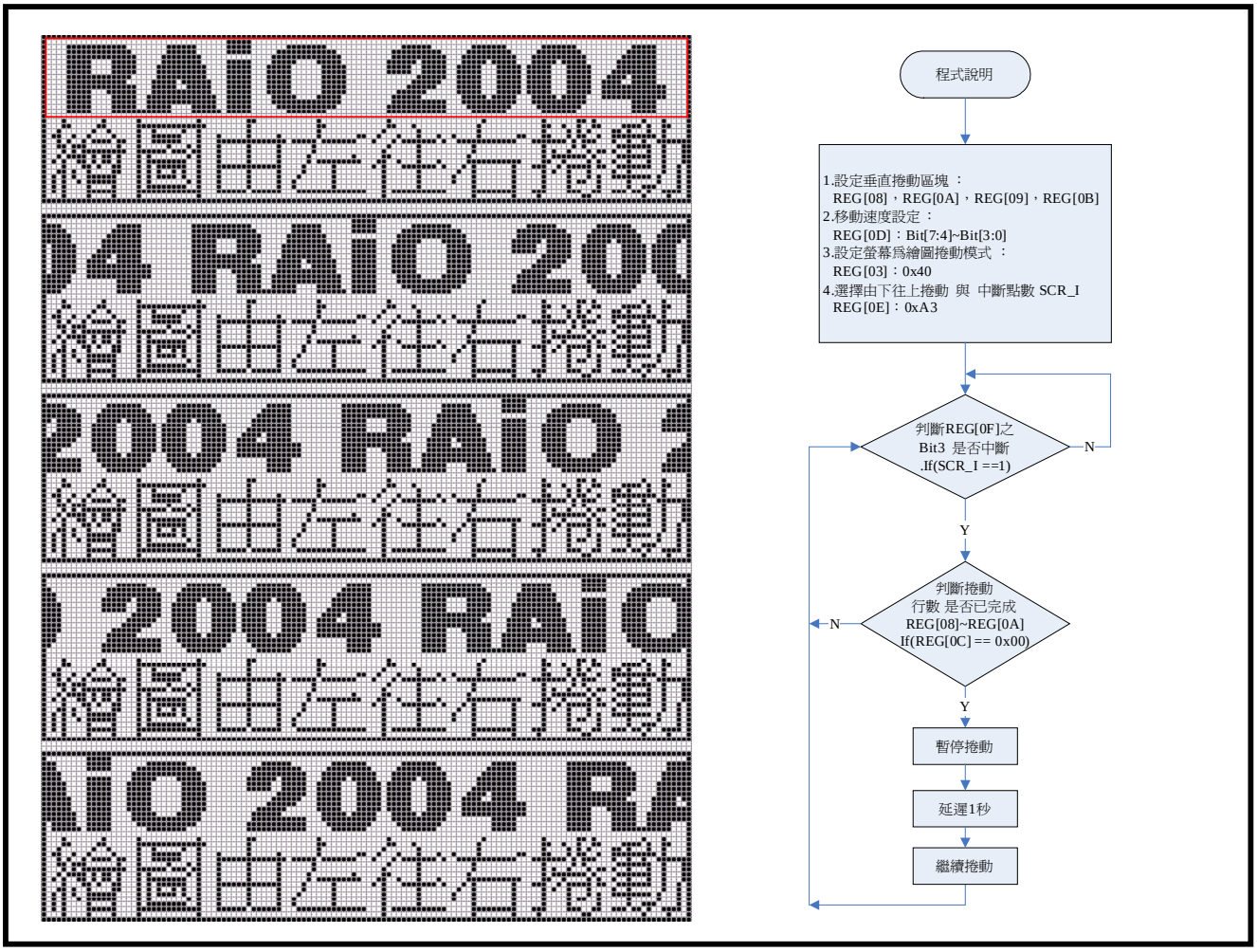


图 11-4： 水平卷动(由左往右)

以下为 图 11-4 范例程序：

```
LCD_Clear_LCD                                   // 清除所有显示
LCD_CmdWrite (0x03,0x00);                       // 设定绘图显示模式
LCD_CmdWrite (0x04,0x74);                       // 关闭游标
LCD_CmdWrite (0x01,0x02);                       // 开启 LCD 显示
GotoXY (0,0);                                   // 设定光标地址
for(length=0; length< 256 ; length++)
    LCD_DataWrite (~(DataString2[length]));     // 写入 RAiO 2004  图形(128 * 16)

LCD_CmdWrite (0x03,0x03);                       // 设定文字显示模式
GotoXY (0,16);                                  // 设定光标地址
```

```
PrintStr ("绘图由左往右卷动",1);                    // 写入中文字符串
LCD_CmdWrite(0x08,0x00);                            // 设定卷动范围水平起始地址
LCD_CmdWrite(0x09,0x00);                            // 设定卷动范围垂直起始地址
LCD_CmdWrite(0x0A,0x0f);                            // 设定卷动范围水平结束地址
LCD_CmdWrite(0x0B,0x0f);                            // 设定卷动范围垂直结束地址

LCD_CmdWrite(0x0D,0x30);                            // 卷动速度设定
LCD_CmdWrite(0x03,0x40);                            // 设定绘图卷动功能模式
LCD_CmdWrite(0x0E,0xA3);                            // 由左往右卷动时，每位移 8 点，SCR_I 产生
                                                    // 中断

While(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)         // 侦测 SCR_I 是否为”1”(中断)
  {
    if(LCD_CmdRead_SPI3(0x0C) == 0x00)              // 判断是否已经卷动 16 点完成
    {
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) & 0xFE);  // 暂停卷动
      delay(1000);                                  // 延迟 1 秒
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) | 0x01);  // 继续卷动
    }
  }
}
```

(d)由右往左卷动(水平卷动)：

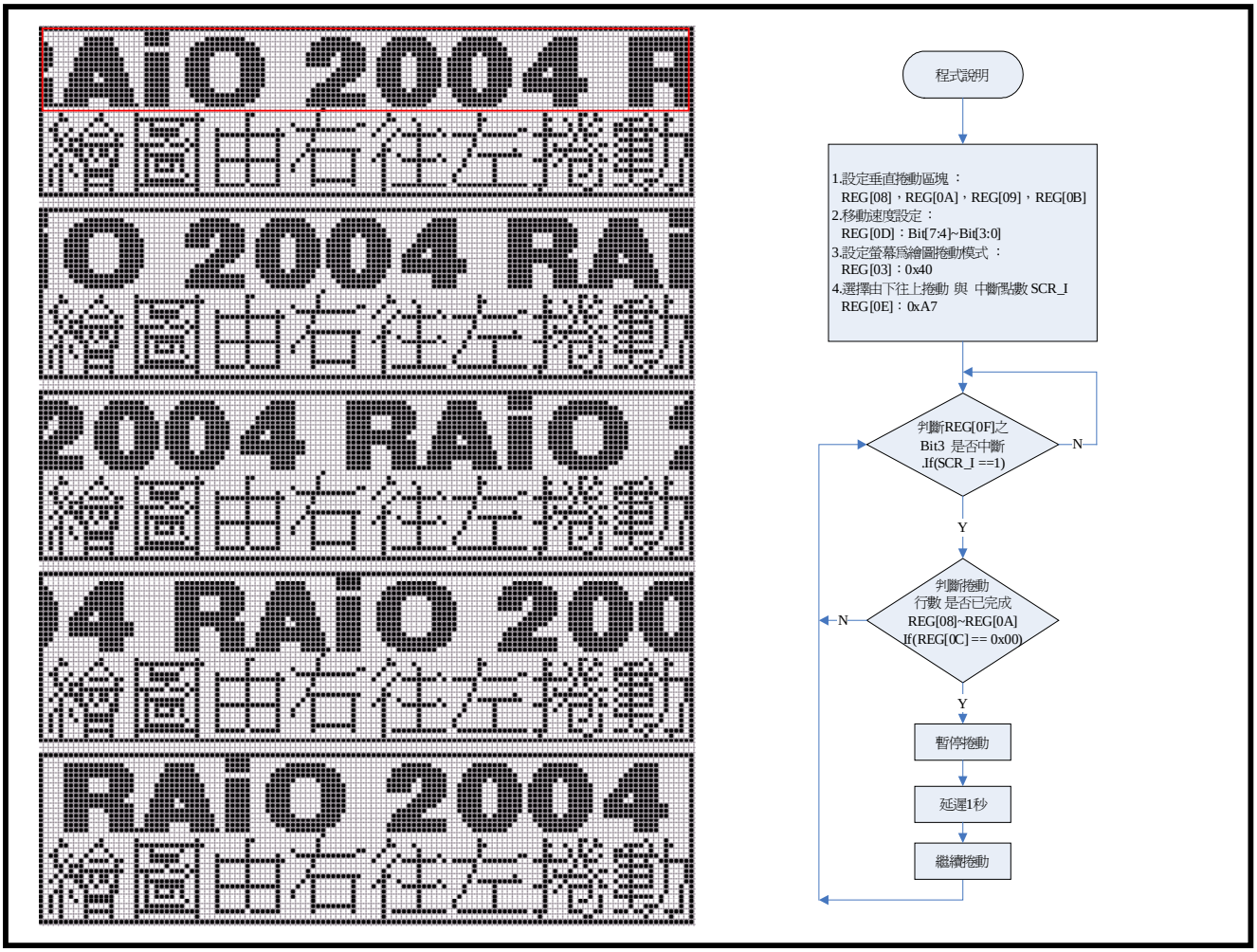


图 11-5： 水平卷动(由右往左)

以下为 图 11-5 范例程序：

```
LCD_Clear_LCD                                   // 清除所有显示
LCD_CmdWrite (0x03,0x00);                       // 设定绘图显示模式
LCD_CmdWrite (0x04,0x74);                       // 关闭游标
LCD_CmdWrite (0x01,0x02);                       // 开启 LCD 显示
GotoXY (0,0);                                   // 设定光标地址
for(length=0; length< 256 ; length++)
    LCD_DataWrite (~(DataString2[length]));     // 写入 RAiO 2004  图形(128 * 16)

LCD_CmdWrite (0x03,0x03);                       // 设定文字显示模式
GotoXY (0,16);                                  // 设定光标地址
```

```
PrintStr ("绘图由右往左卷动",1);                    // 写入中文字符串
LCD_CmdWrite(0x08,0x00);                            // 设定卷动范围水平起始地址
LCD_CmdWrite(0x09,0x00);                            // 设定卷动范围垂直起始地址
LCD_CmdWrite(0x0A,0x0f);                            // 设定卷动范围水平结束地址
LCD_CmdWrite(0x0B,0x0f);                            // 设定卷动范围垂直结束地址

LCD_CmdWrite(0x0D,0x30);                            // 卷动速度设定
LCD_CmdWrite(0x03,0x40);                            // 设定绘图卷动功能模式
LCD_CmdWrite(0x0E,0xA7);                            // 由右往左卷动时，每位移 8 点，SCR_I 产生
                                                    // 中断

While(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)         // 侦测 SCR_I 是否为”1”(中断)
  {
    if(LCD_CmdRead_SPI3(0x0C) == 0x00)              // 判断是否已经卷动 16 点完成
    {
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) & 0xFE);   // 暂停卷动
      delay(1000);                                  // 延迟 1 秒
      LCD_CmdWrite_SPI3(0x0E,LCD_CmdRead_SPI3(0x0E) | 0x01);   // 继续卷动
    }
  }
}
```

12. 绘图模式移动

RA8815 提供 LCD 显示画面可经由缓存器的设定指定一区块，使硬件自动做屏幕画面移动的效果，其中缓存器的功能设定主要包刮移动画面的方向与速度，及每次移动画面的像数与移动的范围，以下针对实际应用时程序撰写方式作一说明。

12-1 移动方向

可由缓存器 REG-[0Eh]之 BIT[3：2] 设定，选择四种不同的卷动方式。

缓存器设定：

	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
REG[0Eh]	SCR_IMD1	SCR_IMD0	AUTO_SCR	SBUF	**SCR_MD1**	**SCR_MD0**	SCR_INTEN	SCR_EN

Bit	Description	Default	Access
3-2	设定移动方向： 0 0 ：由左往右移动(水平移动) 0 1 ：由右往左移动(水平移动) 1 0 ：由上往下移动(垂直移动) 1 1 ：由下往上移动(垂直移动)	0h	R/W

12-2 移动范围设定

设定 REG-[08h]、REG-[09h]、REG-[0Ah]、REG-[0Bh]四个缓存器，可指定一区块作为移动的范围。

图 12-1： 设定移动范围

(a)缓存器设定：移动区块水平起始地址(X1)

REG[08h]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	--	--	--	SSX4	SSX3	SSX2	SSX1	SSX0

(b)缓存器设定：移动区块垂直起始地址(Y1)

REG[09h]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	--	--	SSY5	SSY4	SSY3	SSY2	SSY1	SSY0

(c)缓存器设定：移动区块水平移动范围(X2)

REG[0Ah]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	--	--	--	SRX4	SRX3	SRX2	SRX1	SRX0

(d)缓存器设定：移动区块垂直移动范围(Y2)

REG[0Bh]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	--	--	SRY5	SRY4	SRY3	SRY2	SRY1	SRY0

12-3 移动速度设定

(a)REG-[0Dh]之 Bit[7：4]设定自动移动的速度。

(b)REG-[0Dh]之 Bit[3：0]设定每次位移的点数(每次移动像素 1~16) 。

缓存器设定：

REG[0Dh]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	SPD3	SDP2	SPD1	SPD0	STP3	STP2	STP1	STP0

Bit	Description	Default	Access
7-4	设定自动移动的速度： 0 0 0 0 ：最快(8 个 Frame) . . . 1 1 1 1 ：最慢(128 个 Frame)	0h	R/W
3-0	设定每次位移的像素： 0 0 0 0 ：最小 1 个位移的像素 . .	0h	R/W

	. 1 1 1 1 ：最大 16 个位移的像素		

12-4 程序范例

(a)由上往下移动(垂直移动)：

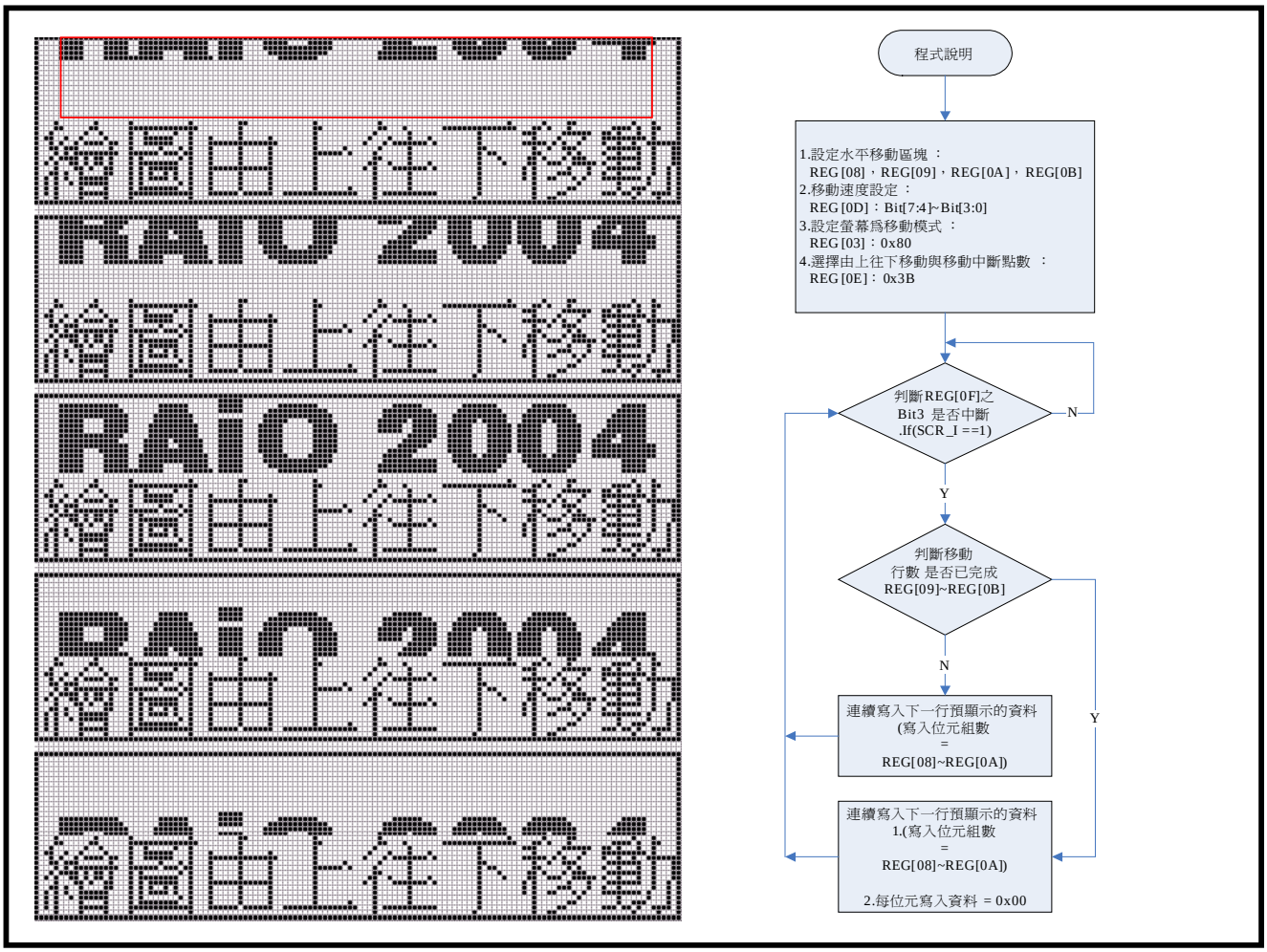


图 12-2： 垂直移动(由上往下)

以下为 图 12-2 范例程序：

```
LCD_Clear_LCD                          // 清除所有显示
LCD_CmdWrite (0x03,0x43);              // 中文模式
GotoXY_SPI3(0,16);                     // 设定光标地址
PrintStr_SPI3("绘图由上往下移动试",1    // 显示中文字符串
```

```
LCD_CmdWrite(0x0E,0x00);                // 取消所有卷动与移动功能
LCD_CmdWrite(0x03,0x40);                // 绘图模式

LCD_CmdWrite(0x08,0x00);                // 设定移动范围水平起始地址
LCD_CmdWrite(0x09,0x00);                // 设定移动范围垂直起始地址
LCD_CmdWrite(0x0A,0x0F);                // 设定移动范围水平结束地址
LCD_CmdWrite(0x0B,0x0F);                // 设定移动范围垂直结束地址

LCD_CmdWrite(0x0D,0x30);                // 卷动速度设定
LCD_CmdWrite(0x03,0x80);                // 移动功能设定
LCD_CmdWrite(0x0E,0x3B);                // 启动由上而下卷动功能，每卷动 1 行点数时
                                        // 就发出 SCR_I 中断
Shift_Count2 = 240;
Shift_Count  = 0;

While(1)
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)     // 侦测 SCR_I 是否为”1”(中断)
  {
    if(Shift_Count++ < 15)                      // 判断是否已经移动 15 行(15 点-LCD COM )
    {
      for(i=0 ; i<16 ; i++)                     // 每中断一次就连续写入 16Byte 的图文件资料
      {
        LCD_DataWrite(~(DataString2[Shift_Count2]));
        Shift_Count2++;
      }
      if((Shift_Count2 -= 32) == 0)
        Shift_Count2 = 240;
    }
    Else
    {
      for(i=0 ; i<16 ; i++)                     // 当已经写入并移动至 16 次之后，接着再发生其它的
                                                // 中断时就将 16byte 的资料全部写入”0x00”如此可达
                                                // 到移动的效果
        LCD_DataWrite(0x00);
```

```
    }
LCD_CmdWrite(0x0F,LCD_CmdRead(0x0F) & 0xFB);   // 清除 SCR_I 中断旗标为”0”
```

(b)由下往上移动(垂直移动)：

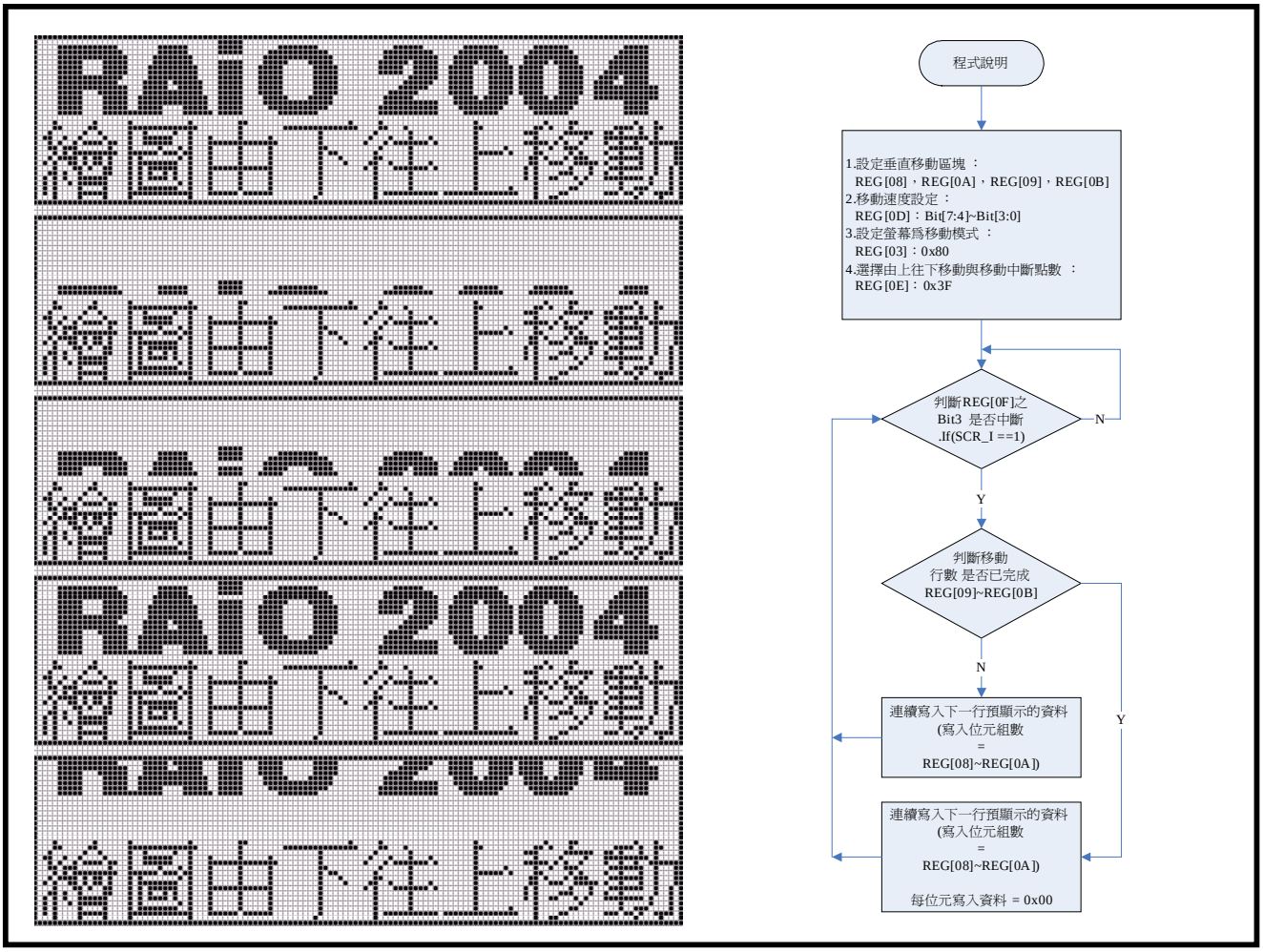


图 12-3： 垂直移动(由下往上)

以下为 图 12-3 范例程序：

```
LCD_CmdWrite(0x0E,0x00);                    // 清除移动缓存器
LCD_CmdWrite(0x03,0x03);                    // 设定正常文字显示模式
LCD_Clear();                                // 清除所有显示
GotoXY (0,16);                              // 设定光标地址
PrintStr("绘图由下往上移动",1);              // 写入中文字符串
```

```
LCD_CmdWrite(0x08,0x00);                // 设定移动范围水平起始地址
LCD_CmdWrite(0x09,0x00);                // 设定移动范围垂直起始地址
LCD_CmdWrite(0x0A,0x0F);                // 设定移动范围水平结束地址
LCD_CmdWrite(0x0B,0x0F);                // 设定移动范围垂直结束地址

LCD_CmdWrite(0x0d,0x60);                // 设定移动速度
LCD_CmdWrite(0x03,0x80);                // 设定绘图移动模式
LCD_CmdWrite(0x0E,0x3F);                // 开启由下往上移动功能，并且每移动一行就产生中
                                        断 SCR_I


Shift_Count2 = 0;                       // 清除使用者判断旗标
Shift_Count = 0;                        // 清除使用者判断旗标
while(1)                                //
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)    // 判断中断 SCR_I
  {
    if(Shift_Count++ < 16)              // 判断是否已经移动 16 行(16 点-LCD COM )
    {
      for(i=0 ; i<16 ; i++)             // 每中断一次就连续写入 16Byte 的图文件资料
        LCD_DataWrite(~(DataString2[Shift_Count2++]));
    }
    else
    {
      for(i=0 ; i<16 ; i++)             // 当已经写入并移动至 16 次之后，接着再发生其它
                                        // 的中断时就将 16byte 的资料全部写入”0x00”如此
                                        // 可达到移动的效果
        LCD_DataWrite(0x00);
    }
//-------------------------------------------------------------------------------------------------------------------//
//--- 重复执行绘图移动的功能 ----------------------------------------------------------------------------------------//
    if(Shift_Count2 >= 512)
      Shift_Count2 = 0;
    if(Shift_Count > 64)
    {
```

```
            Shift_Count2 = 0;
            Shift_Count = 0;
        }
        LCD_CmdWrite(0x0F,LCD_CmdRead(0x0F) & 0xFB);     // 清除 SCR_I 中断旗标为"0"
    }
}
```

(c)由左往右移动(水平移动)：

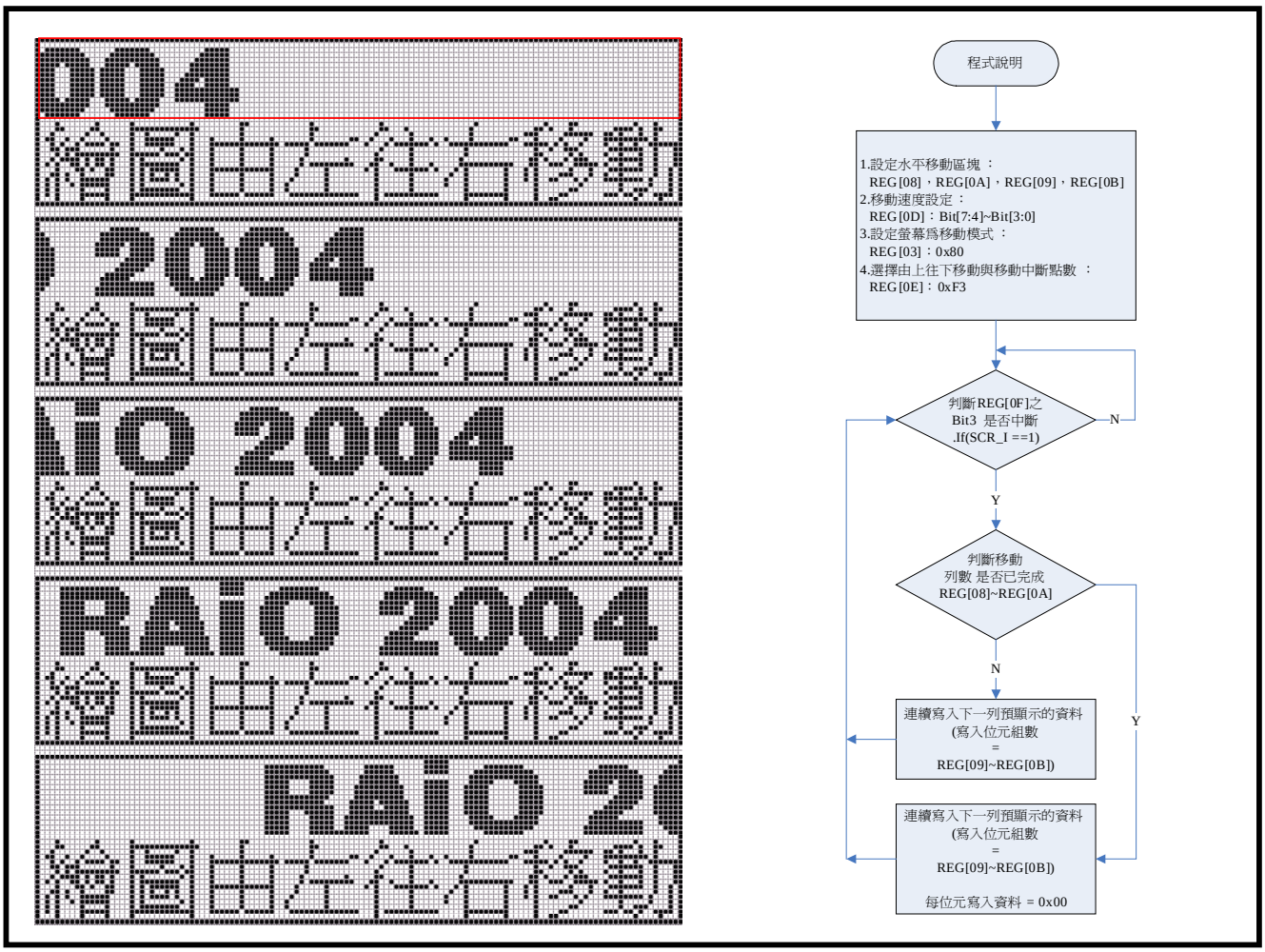


图 12-4： 水平移动(由左往右)

以下为 图 12-4 范例程序：

```
LCD_CmdWrite(0x0E,0x00);                    // 清除移动缓存器
LCD_CmdWrite(0x03,0x03);                    // 设定正常文字显示模式
LCD_Clear();                                // 清除所有显示
GotoXY (0,16);                              // 设定光标地址
PrintStr ("绘图由左往右移动",1);             // 写入中文字符串

LCD_CmdWrite(0x08,0x00);                    // 设定移动范围水平起始地址
LCD_CmdWrite(0x09,0x00);                    // 设定移动范围垂直起始地址
LCD_CmdWrite(0x0A,0x0F);                    // 设定移动范围水平结束地址
LCD_CmdWrite(0x0B,0x0F);                    // 设定移动范围垂直结束地址

LCD_CmdWrite(0x0d,0x20);                    // 设定移动速度
LCD_CmdWrite(0x03,0x80);                    // 设定绘图移动模式
LCD_CmdWrite(0x0E,0xF3);                    // 开启由左往右移动功能，并且每移动一列就产生中断
                                            SCR_I


Shift_Count2 = 14;                          // 清除使用者判断旗标
Shift_Count = 0;                            // 清除使用者判断旗标
R1 = 0;                                     // 清除使用者判断旗标

while(1)                                    //
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)    // 判断中断 SCR_I
  {
    if(Shift_Count++ < 16)                  // 判断是否已经移动 16 行(16 点-LCD COM )
    {
      for(i=0 ; i<16 ; i++)                 // 每中断一次就连续写入 32Byte 的图文件资料
      {
        for(j=0 ; j<2 ;j++)
          LCD_DataWrite(~(DataString2[Shift_Count2+j+(i*16)]));
      }
    Shift_Count2-=2;
```

```
        Shift_Count+=2;
        }
        else
        {
          for(i=0 ; i<32 ; i++)                    // 当已经写入并移动至 16 次之后，接着再发生其它的中
                                                   // 断时就将 32byte 的资料全部写入”0x00”如此可达到移
                                                   // 动的效果
            LCD_DataWrite(0x00);
        }
//-------------------------------------------------------------------------------------------------------------------//
//--- 重复执行绘图移动的功能 ------------------------------------------------------------------------------------------//
    if(R1++ > 16)
    {
        Shift_Count = 0;                           // 清除使用者判断旗标
        Shift_Count2 = 14;                         // 清除使用者判断旗标
        R1 = 0;                                    // 清除使用者判断旗标
    }
    LCD_CmdWrite(0x0F,LCD_CmdRead(0x0F) & 0xFB);     // 清除 SCR_I 中断旗标为”0”
  }
}
```

(d)由右往左移动(水平移动)：

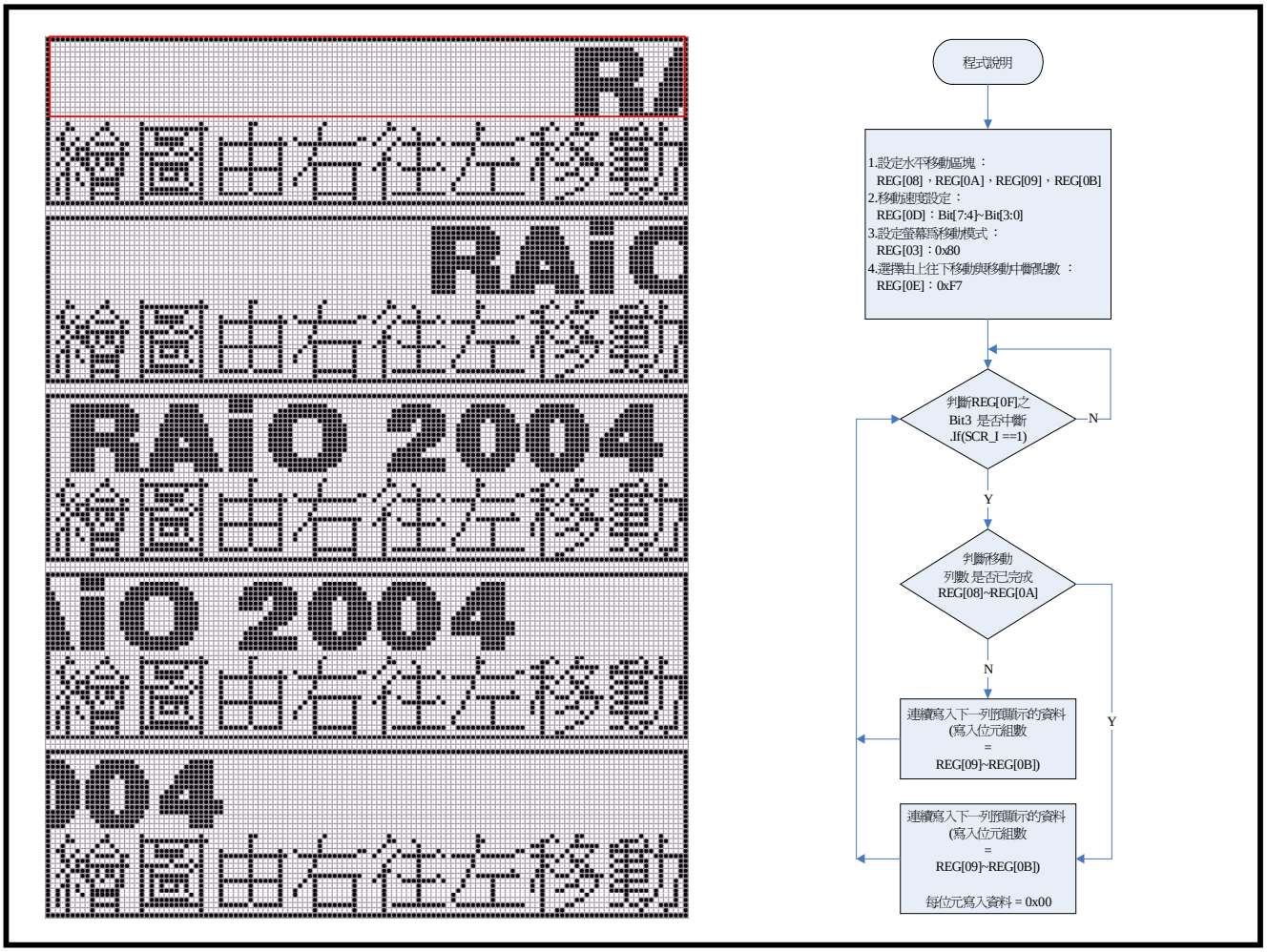


图 12-5： 水平移动(由右往左)

以下为 图 12-5 范例程序：

```
LCD_CmdWrite(0x0E,0x00);                // 清除移动缓存器
LCD_CmdWrite(0x03,0x03);                // 设定正常文字显示模式
LCD_Clear();                            // 清除所有显示
GotoXY (0,16);                          // 设定光标地址
PrintStr ("绘图由左往右移动",1);          // 写入中文字符串

LCD_CmdWrite(0x08,0x00);                // 设定移动范围水平起始地址
LCD_CmdWrite(0x09,0x00);                // 设定移动范围垂直起始地址
LCD_CmdWrite(0x0A,0x0F);                // 设定移动范围水平结束地址
```

```
LCD_CmdWrite(0x0B,0x0F);                    // 设定移动范围垂直结束地址

LCD_CmdWrite(0x0d,0x20);                    // 设定移动速度
LCD_CmdWrite(0x03,0x80);                    // 设定绘图移动模式
LCD_CmdWrite(0x0E,0xF7);                    // 开启由右往左移动功能，并且每移动一列就产生
                                            中断 SCR_I


Shift_Count = 0;                            // 清除使用者判断旗标
R1 = 0;                                     // 清除使用者判断旗标

while(1)                                    //
{
  while((LCD_CmdRead(0x0f) & 0x04) == 0x04)    // 判断中断 SCR_I
  {
    if(Shift_Count++ < 16)                  // 判断是否已经移动 16 行(16 点-LCD COM )
    {
      for(i=0 ; i<16 ; i++)                 // 每中断一次就连续写入 32Byte 的图文件资料
      {
        for(j=0 ; j<2 ;j++)
          LCD_DataWrite(~(DataString2[Shift_Count+j+(i*16)]));
      }
    Shift_Count+=2;
    }
    else
    {
      for(i=0 ; i<32 ; i++)                 // 当已经写入并移动至 16 次之后，接着再发生其
                                            // 他的中断时就将 32byte 的资料全部写入”0x00”
                                            // 如此可达到移动的效果
        LCD_DataWrite(0x00);
    }
//-------------------------------------------------------------------------------------------------------------------//
//--- 重复执行绘图移动的功能 ----------------------------------------------------------------------------------------//
   if(R1++ > 16)
   {
```

```
            Shift_Count = 0;                          // 清除使用者判断旗标
            R1 = 0;                                   // 清除使用者判断旗标
        }
        LCD_CmdWrite(0x0F,LCD_CmdRead(0x0F) & 0xFB);     // 清除 SCR_I 中断旗标为”0”
      }
    }
```

13. EL(冷光)驱动信号

RA8815 提供使用者冷光驱动信号(EL_DCHG 与 EL_CHRG)，只要加入简易的外部电路，即可产生(EL)所需要的交流高压讯号，而直接驱动冷光板。图 13-1 为简单的驱动电路及充放电所需的输出波形，另外也可透过缓存器去设定每次产生冷光控制信号的输出时间。

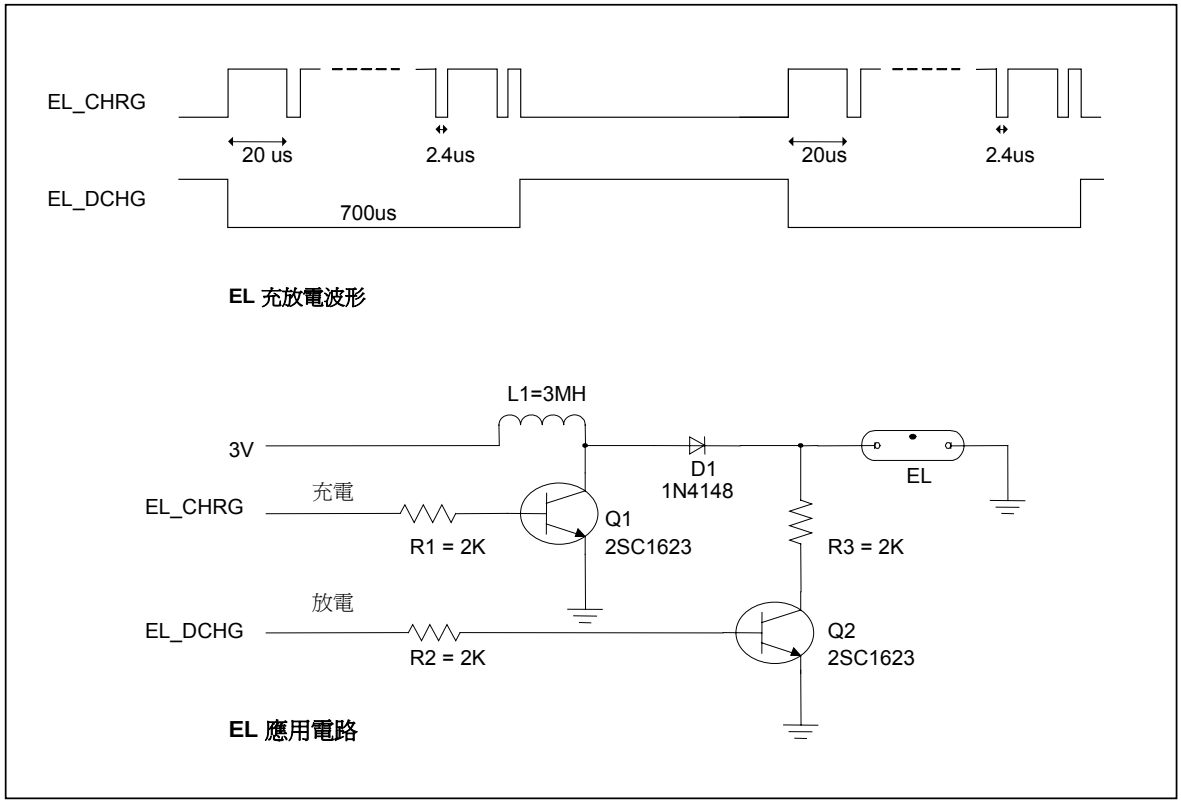


图 13-1：冷光驱动信号与电路

缓存器设定：

REG[16h]	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	EL_EN	--	--	--	ELT3	ELT2	ELT1	ELT0

Bit	Description	Default	Access
2-0	8 个 16x16 全型字内存的地址分配 0 0 0 0 ： 1 秒 0 0 0 1 ： 2 秒 0 0 1 0 ： 4 秒 0 0 1 1 ： 6 秒 0 1 0 0 ： 8 秒 0 1 0 1 ： 10 秒 0 1 1 0 ： 12 秒 0 1 1 1 ： 14 秒 1 0 0 0 ： 16 秒 1 0 0 1 ： 18 秒 1 0 1 0 ： 20 秒 1 0 1 1 ： 22 秒 1 1 0 0 ： 24 秒 1 1 0 1 ： 26 秒 1 1 1 0 ： 28 秒 1 1 1 1 ： 30 秒	0h	W

上表的输出时间是以 RC 振荡(系统频率)为 45KHz 时为标准，RC 振荡频率愈高则输出时间愈短。

范例程序：

```
LCD_CmdWrite(0x16,0x85);          // 打开冷光驱动讯号输出，约 10 秒之后自动关闭。
```

14. 使用者造字功能

RA8815 内建繁体或简体中文字型码，以及常用的 ASCII 字型，除此之外，另外也提供存放 8 个 16x16 的全型字的内存，可让使用者将自行描绘的全型字或图案存入这些地址，如此可提高使用者每次读取这些字型或图案显示时的效率。使用方式如下说明：

缓存器设定：(造字缓存器)

REG[17h]	***DB7***	***DB6***	***DB5***	***DB4***	***DB3***	***DB2***	***DB1***	***DB0***
	--	--	--	--	--	UMI2	UMI1	UMI0

Bit	Description	Default	Access
2-0	8 个 16x16 全型字内存的地址分配 0 0 0 ： FFF0h 0 0 1 ： FFF1h 0 1 0 ： FFF2h 0 1 1 ： FFF3h 1 0 0 ： FFF4h 1 0 1 ： FFF5h 1 1 0 ： FFF6h 1 1 1 ： FFF7h	0h	R

缓存器设定：(造字数据缓存器)

REG[17h]	***DB7***	***DB6***	***DB5***	***DB4***	***DB3***	***DB2***	***DB1***	***DB0***
	CGMD7	CGMD6	CGMD5	CGMD4	CGMD3	CGMD2	CGMD1	CGMD0

范例说明：

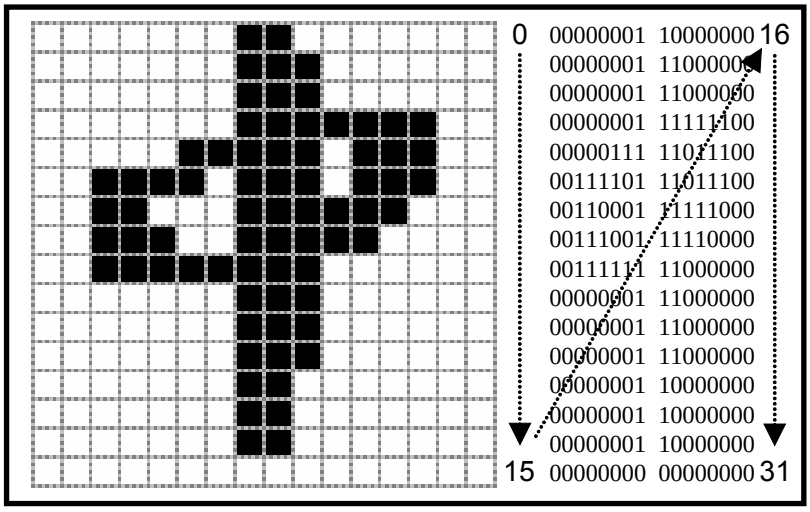


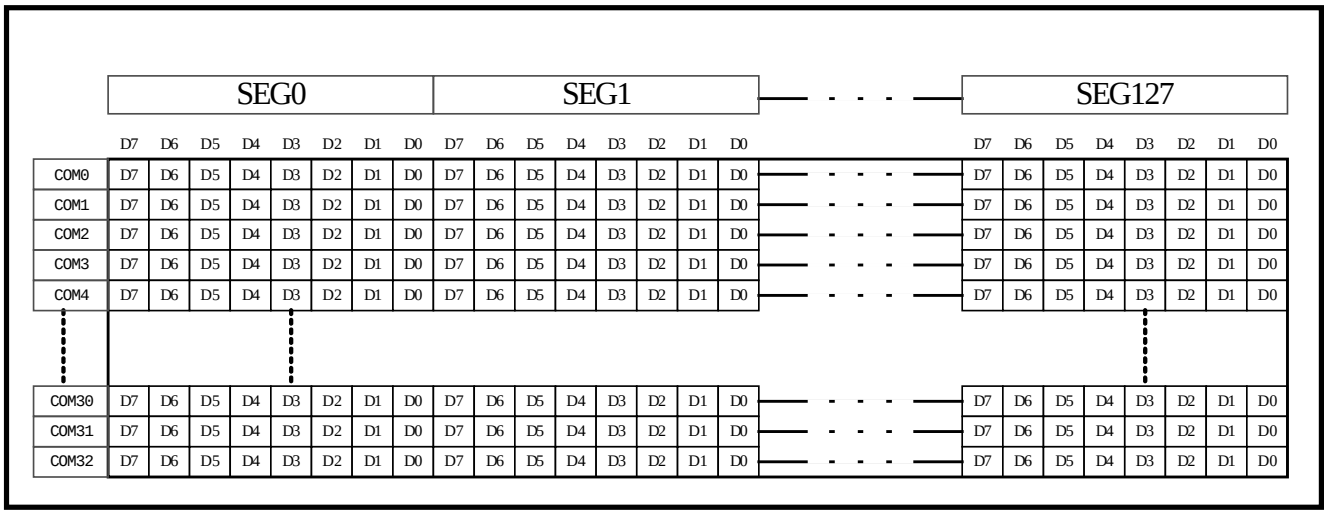
图 14-1 (BMP = FFF2h)

以下为 图 14-1 范例程序：

```
//*****************************************************************************************//
//*** 造字范例 ： 写入内存地址 = FFF2h ***************************************************//
//*****************************************************************************************//
LCD_CmdWrite(0x03,0x03);                    // 设定正常文字显示模式
LCD_CmdWrite_SPI3(0x17,0x02);               // 设定造字存放地址为 FFF2h
length = 0;
for(i=0 ;I < 32 ; i++)
   LCD_CmdWrite_SPI3(0x18,BMP[length]++)    // 连续写入 32 字节的造字数据
;
.                                           // (写入顺序如图 14-1 所示)
.
//*****************************************************************************************//
//*** 造字范例 ： 读取内存地址 = FFF2h ***************************************************//
//*****************************************************************************************//

LCD_DataWrite_SPI3(0xff);                   // LCD 显示(图 14-1)内存地址 FFF2h 的造字字型
LCD_DataWrite_SPI3(0xf2);
.
.
```

附录 A. 显示内存扫描映像图



附录 B. 子程序范例

```
//=============================================================================
//====== COMMAND.h    ********* RA8815 共享子程序 ******=======================//
//======  MPU  Interface  :  8080  mode  /  8bit  Databus
//=============================================================================
extern void LCD_Reset(void);
extern void LCD_CmdWrite(unsigned char,unsigned char);
extern unsigned char LCD_CmdRead(unsigned char);
extern void LCD_DataWrite(unsigned char);
extern unsigned char LCD_DataRead();
extern void LCD_ChkBusy(void);
extern void LCD_Initial(void);
extern void LCD_Clear(void);
extern void PrintStr(char *ptr,int delay_time);
extern void putHEX(unsigned int var);
extern void LCD_ChkBusy(void);
extern void GotoXY(unsigned char x1,unsigned char y1);

//*****************************************************************************//
// ***  重置子程序 ***************************************************************//
//*****************************************************************************//
void LCD_Reset(void)
{
  LCD_RST = 1;
  LCD_WR = 1;

  LCD_RST = 0;                          // MPU 控制 RA8815 为低准位
  delay(100);                           // 低准位延迟至少 5 毫秒
  LCD_RST = 1;                          // MPU 控制 RA8815 为高准位
  delay(100);                           // 延迟至少 350 毫秒
}

//*****************************************************************************//
// *** 缓存器写入数据子程序*****************************************************//
//*****************************************************************************//
void LCD_CmdWrite(unsigned char cmdReg,unsigned char cmdData)
{
  LCD_cmdReg = cmdReg;                  // 缓存器地址
  LCD_CS =0;                            // 致能读写 RA8815
  LCD_RD = 1;
  LCD_RS = 0;                           // 设定为读写缓存器模式

  LCD_WR = 0;                           // 开始写入缓存器地址
  LCD_WR = 1;
```

```
    LCD_RS = 1;
    LCD_CS =1;                          // 禁能读写 RA8815

    LCD_cmdReg = cmdData;               // 缓存器数据
    LCD_CS =0;                          // 致能读写 RA8815
    LCD_RD = 1;
    LCD_RS = 0;                         // 设定为读写缓存器模式

    LCD_WR = 0;                         // 开始写入缓存器数据
    LCD_WR = 1;

    LCD_RS = 1;
    LCD_CS =1;                          // 禁能读写 RA8815
}

//*********************************************************************************//
// *** 显示内存写入数据子程序 ***********************************************//
//*********************************************************************************//
void LCD_DataWrite(unsigned char WrData)
{
    LCD_ChkBusy();
    LCD_DATA = WrData;                  // 准备显示内存数据
    LCD_CS =0;                          // 致能读写 RA8815
    LCD_RD = 1;
    LCD_RS = 1;                         // 设定为读写显示内存模式

    LCD_WR = 0;                         // 开始写入显示内存的数据
    LCD_WR = 1;

    LCD_CS =1;                          // 禁能读写 RA8815
}

//*********************************************************************************//
// *** 读取缓存器数据子程序*************************************************//
//*********************************************************************************//
unsigned char LCD_CmdRead(unsigned char cmdReg)
{
    unsigned char REG_Read;
    LCD_cmdReg = cmdReg;
    LCD_CS =0;                          // 致能读写 RA8815
    LCD_RD = 1;
    LCD_RS = 0;                         // 设定为读写缓存器模式

    LCD_WR = 0;                         // 开始写入缓存器地址
```

```
    LCD_WR = 1;

    LCD_RS = 1;
    LCD_CS =1;                          // 禁能读写 RA8815

    LCD_DATA = 0xff;
    LCD_CS =0;                          // 致能读写 RA8815
    LCD_WR = 1;
    LCD_RS = 0;

    LCD_RD = 0;                         // 开始读取缓存器数据
    REG_Read = LCD_DATA;
    LCD_RD = 1;

    LCD_RS = 1;
    LCD_CS =1;                          // 禁能读写 RA8815
    return REG_Read;                    // 回传读取缓存器内部的数据
}

//*********************************************************************************************//
// *** 读取显示内存数据子程序*****************************************************************//
//*********************************************************************************************//
unsigned char LCD_DataRead()
{
    unsigned char REG_Read;
    LCD_ChkBusy();
    LCD_DATA = 0xff;
    LCD_WR = 1;
    LCD_RS = 1;                         // 设定为读写显示内存模式
    LCD_CS =0;                          // 致能读写 RA8815
    LCD_RD = 0;

    REG_Read = LCD_DATA;
    LCD_RD = 1;
    LCD_CS =1;                          // 禁能读写 RA8815
    return REG_Read;
}

//*********************************************************************************************//
// *** 清除显示内存数据子程序*****************************************************************//
//*********************************************************************************************//
void LCD_Clear(void)
{
    unsigned char READ_REG;
    READ_REG = LCD_CmdRead(0x01);
```

```
    READ_REG &= 0xBf;
    READ_REG |= 0x42;                    // 设定 Bit6 为"1"，硬件自动将显示内存写入"0"清
    LCD_CmdWrite(0x01,READ_REG);
    LCD_ChkBusy();
    delay(10000);                        // 延迟 10 毫秒
}

//*********************************************************************************//
// *** 初始 RA8815 所有缓存器子程序*************************************************//
//*********************************************************************************//
void LCD_Initial(void)
{
    LCD_CmdWrite(0x00,0x00);             // 驱动波形设定缓存器
    LCD_CmdWrite(0x01,0x00);             // 电源控制缓存器
    LCD_CmdWrite(0x02,0x79);             // 系统设定缓存器
    LCD_CmdWrite(0x03,0x00);             // 内存输入模式缓存器

    LCD_CmdWrite(0x04,0x75);             // 光标控制缓存器
    LCD_CmdWrite(0x05,0x00);             // 光标 X 位置缓存器
    LCD_CmdWrite(0x06,0x00);             // 光标 Y 位置缓存器
    LCD_CmdWrite(0x07,0x00);             // 键盘扫描控制缓存器

    LCD_CmdWrite(0x08,0x00);             // X 轴卷动起始点缓存器
    LCD_CmdWrite(0x09,0x00);             // Y 轴卷动起始点缓存器

    LCD_CmdWrite(0x0A,0x00);             // X 轴卷动范围缓存器
    LCD_CmdWrite(0x0B,0x00);             // Y 轴卷动范围缓存器


    LCD_CmdWrite(0x0C,0x00);             // 卷动位移量缓存器
    LCD_CmdWrite(0x0D,0x00);             // 自动卷动控制缓存器
    LCD_CmdWrite(0x0E,0x00);             // 卷动控制缓存器
    LCD_CmdWrite(0x0F,0x00);             // 中断状态缓存器

    LCD_CmdWrite(0x10,0x00);             // 对比调整缓存器

    LCD_CmdWrite(0x11,0x00);             // 驱动控制缓存器 A
    LCD_CmdWrite(0x12,0x00);             // 驱动控制缓存器 B

    LCD_CmdWrite(0x13,0x08);             // 闪烁设定缓存器
    LCD_CmdWrite(0x14,0x00);             // IO 端口方向设定缓存器
    LCD_CmdWrite(0x15,0x00);             // IO 端口数据缓存器
    LCD_CmdWrite(0x16,0x00);             // 冷光控制缓存器
    LCD_CmdWrite(0x17,0x00);             // 造字选择缓存器
    LCD_CmdWrite(0x18,0x00);             // 造字数据缓存器
```

```
}

//*********************************************************************************//
// *** LCD 显示字符串子程序*****************************************************//
//*********************************************************************************//
void PrintStr(char *ptr,int delay_time)          // delay_time ： 每个字显示的间隔
{
  while(*ptr != '\0')                            // 判断是否为字符串的最后(空字符串)
  {
    LCD_DataWrite(*ptr);                         // 例如： PrintStr("ABC12345",1)
    ++ptr;                                       // 则 LCD 显示 ABC12345
  }
}

//*********************************************************************************//
// *** 设定光标位置子程序*******************************************************//
//*********************************************************************************//
extern void GotoXY(unsigned char x1,unsigned char y1)
{
  LCD_CmdWrite(0x05,x1);                         // 设定 X 坐标值
  LCD_CmdWrite(0x06,y1);                         // 设定 Y 坐标值
}

//*********************************************************************************//
// *** 读取 RA8815 忙碌旗标子程序***********************************************//
//*********************************************************************************//
void LCD_ChkBusy(void)
{
  do
  {
  }while((LCD_CmdRead(0x0F) & 0x80) ==           //判断忙碌旗标是否为"1"，若是，表示忙碌中。
}

//*********************************************************************************//
// *** LCD 显示两位数 16 进制子程序*********************************************//
//*********************************************************************************//
void PutHEX(unsigned int var)
{
  unsigned char div_val,base;
  base = 16;
  div_val = 0x10;
  do
  {
    LCD_DataWrite(ASCIITable[var / div_val]);    // 例如：0x35，则 LCD 显示 35
    var %= div_val;
```

```
        div_val /= base;
    }while (div_val);
}
```

附录 C. 简单程序范例

```
void main (void)
{
  LCD_Reset();
  LCD_Clear();

  LCD_CmdWrite(0x02,0x79);          // 设定系统显示内存为 128x32 模式
                                    // 中文版本为简体字型(GB 码)
  LCD_CmdWrite(0x11,0xF0);          // 开启内部升压电路
  LCD_CmdWrite(0x12,0x17);          // 调整电压调整器的倍数
  LCD_CmdWrite(0x10,0x57);          // 设定 Bias 与显示对比度

  LCD_CmdWrite(0x01,0x02);          // Bi2=”1”，开启显示屏

  LCD_CmdWrite(0x03,0x02);          // 设定显示为 ASCII 模式
  GotoXY(0,0);                      // 设定光标位置(0,0)
  PrintStr("ASCII mode：8x16",1);   // 印出字符串” ASCII mode：8x16”

  LCD_CmdWrite(0x03,0x03);          // 设定显示为全型文字模式
  GotoXY(0,16);                     // 设定光标位置(0,16)
  PrintStr("中文模式控制",1);       // 印出字符串”中文模式控制”
}
```

附录 D. 字型与字码表(GB)

A1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A			、	。	·	ˉ	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		i	ii	iii	iv	v	vi	vii	viii	ix	x					
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	{	|	}	¯	

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	わ	ゎ
F	ゐ	ゑ	を	ん												

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

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		ā	á	ǎ	à	ē	é	ě	è	ī	ī	ǐ	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	叼	雕	凋	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	毛	夭	爻	卮	氐	囟	胤	馗	毓	睾	鼗	丶	亟	鼐	乜
C	乩	亓	毕	华	昙	瞎	仄	厍	厝	厣	厥	厮	厩	厝	匚	匝
D	瓠	匮	匾	赜	卦	卣	刂	刈	刎	刭	刖	剖	刺	剐	剡	
E	剜	蒯	剽	劂	劁	劐	剔	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	棰	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	鹺	躉	跫	踅	蹙	蹩
C	趵	趿	趼	趺	蹌	跖	跗	跚	躒	跎	跏	跛	跆	跬	蹺	蹕
D	跣	躚	躋	跤	踉	跽	踔	踝	踟	躓	踮	踣	躑	踺	蹀	踹
E	踵	踽	踱	蹉	蹁	蹂	躡	蹣	蹊	躕	蹶	蹼	蹯	蹴	躅	躪
F	躔	躐	躦	躞	豸	貂	貊	貅	貘	貔	斛	觖	觴	觚	觜	

F6	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		觥	觫	觶	訾	謦	靚	雩	靂	雯	霆	霽	霈	霏	霎	霪
B	靄	霰	霾	齔	齟	齙	齠	齜	齦	齬	齪	齷	黽	黿	鼉	隹
C	隼	雋	雎	雒	瞿	讎	銎	鑾	鋈	鏨	鍪	鏊	鎏	鐾	鑫	魷
D	魴	鮁	鮃	鮎	鱸	穌	鮒	鱟	鮐	鮭	鮚	鮪	鮞	鱭	鮫	鯗
E	鱘	鯁	鱺	鰱	鰹	鰣	鰷	鯀	鯊	鯇	鯽	鯖	鯪	鯫	鯡	鯤
F	鯧	鯝	鯢	鯰	鯛	鯴	鯔	鱝	鰈	鰐	鰍	鰒	鰉	鯿	鰠	

F7	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		鰲	鰭	鰨	鰥	鰩	鰳	鰾	鱈	鰻	鰵	鱅	鱖	鱔	鱒	鱧
B	靼	鞅	韃	鞽	鞔	韉	鞫	鞣	鞲	鞴	骱	骰	骷	鶻	骶	骺
C	骼	髁	髀	髏	髂	髖	髕	髑	魅	魃	魘	魎	魈	魍	魑	饗
D	饜	餮	饕	饔	髟	髡	髦	髯	髫	髻	髭	髹	鬈	鬏	鬢	鬟
E	鬣	麽	麾	縻	麂	麇	麈	麋	麒	鏖	麝	麟	黛	黜	黝	黠
F	黟	黢	黷	黧	黥	黲	黯	鼢	鼬	鼯	鼴	鼷	鼽	鼾	齇	

附录 E. 字型使用说明

RA8815 内建 256KByte 的字型 ROM，储存着中文字型、数字符号、英、日、欧、拉丁文等字型，因为中文字型的不同，RA8815 提供两种型号:

1. RA8815G-T：代表内建的中文字型为繁体字
2. RA8815G-S：代表内建的中文字型为简体字

因此在开发及采购时必须先确定使用的 RA8815 型号。而 RA8815 内建的字号共有三种:

1. **中文字型:** 所占的显示区域大小为 16x16，实际字体大小为 15x15，如图 E-1 的(1)，所支持的字收录在 RA8815 应用手册“附录 D. 字型与字码表”，其实由字码表中可以看出所包含的字不只中文，还有包括一些符号、数字等，而这些字型在 RA8815G-T 与 RA8815G-S 是不一样的，RA8815G-T 字型的对映码是 BIG5 码，RA8815G-S 字型的对映码是 BG 码，如使用 RA8815G-T，当 MCU 在设定好坐标位置后送出 Data “B7h”及”E7h”，就会显示” 瑞”的繁体中文字型，使用 RA8815G-S，当 MCU 在设定好坐标位置后送出 Data “C6h” 及 ”CCh”，就会显示” 铺” 的简体中文字型。

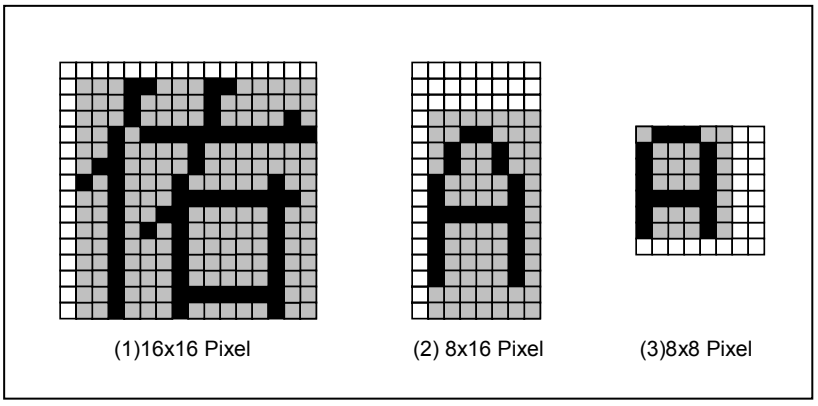


图 E-1：一般显示字型

2. **大 ASCII 字型:** 所占的显示区域大小为 8x16，实际字体大小区域一般显示字型为 6x13，如图 E-1 的(2)，粗体字显示字型为 7x13，如图 E-2 的(2)。大 ASCII 字型共有 1024 个字(请参考 Table-1 与 Table-2)，其中包括一些符号、数字、英、日、欧文及拉丁文，大 ASCII 字型在 RA8815G-T 与 RA8815G-S 也是一样的。只要 RA8815 设定成大 ASCII 字型模式(REG[03h]的 MD[1:0]=10)，由 MCU 送出 Table-1 或 Table-2 的字型码就会显示出所对映的 8x16 字型，如 MCU 在设定好坐标位置后送出 Data “41h”，就会显示”A” 的大 ASCII 字型。选择大 ASCII 表由缓存器[03h]MWMR 的 Bi4 ASCS 决定，ASCS 为 0 选择 ASCII 表 1(Table1)，为 1 选择 ASCII 表 2(Table2) 。

3. **小 ASCII 字型:** 所占的显示区域大小为 8x8，实际字体大小区域一般显示字型为 5x7，如图 E-1 的 (3)，粗体字显示字型为 6x7，如图 E-2 的 (3)。小 ASCII 字型共有 512 个字(请参考 Table-0)，其中包括一些符号、数字、英、日、欧文等，小 ASCII 字型在 RA8815G-T 与 RA8815G-S 是一样的。只要 RA8815 设定成小 ASCII 字型模式(REG[03h]的 MD[1:0]=01)，由 MCU 送出 Table-0 的字型码就会显示出所对映的 8x8 字型，如 MCU 在设定好坐标位置后送出 Data “50h”，就会显示 ”P” 的小 ASCII 字型。

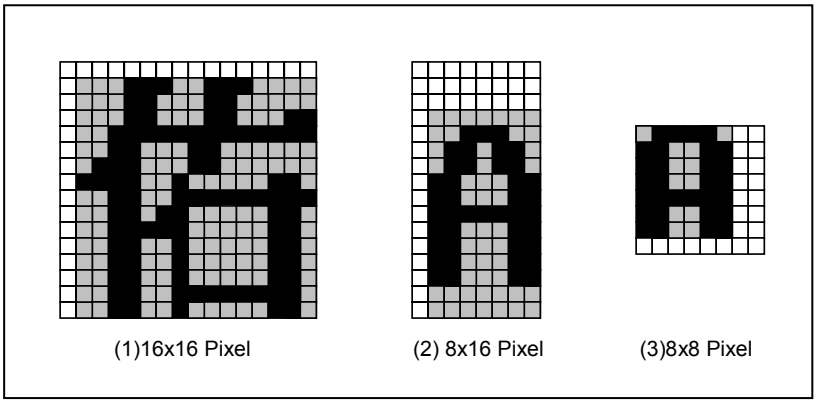


图 E-2： 粗体字显示字型

值得注意的是 Table E-1~3 这些字型所对映的码是由本公司自行规划的，虽然包括一些符号、数字、英、日、欧文及拉丁文，但所对映的字型码与英、日、欧文及拉丁文等操作系统所定义的标准码无关，因此 RA8815 不论透过 MCU 送出的是哪一种文字码或是哪一国的标准码，在大、小 ASCII 模式下，LCD 将只显示出由 RA8815 接收且对映到 Table E-1~3 的字型。而如果不同的操作系统应用上您想显示的字是在 Table E-1~3 内，但是操作系统产生的字型码与 Table E-1~3 上的不同，那么就必须在程序上自行建立一转码表，例如您想显示大 ASCII 字型 ”Ü”，在操作系统中产生的字型码却是”A5h”，那么在程序上的转码表就是要由”A5h” 对映到 ”C8h”，唯有由 MCU 送出 ”68” 才可以让 RA8815 显示出 ”Ü”，又如您想显示大 ASCII 字型 ”£”，在操作系统中产生的字型码却是 ”9Ch”，那么在程序上的转码表就是要由 ”98h” 对映到”9Ch”，因为唯有由 MCU 送出 ”9C” 才可以让 RA8815 显示出 ” £”。

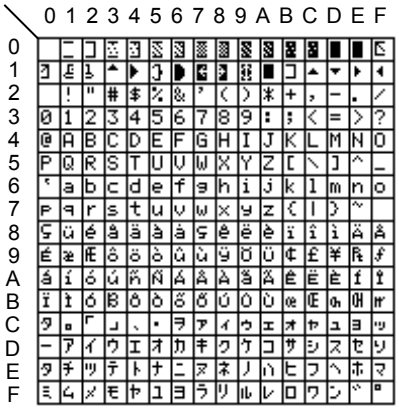


Table E-1: 小 ASCII 表

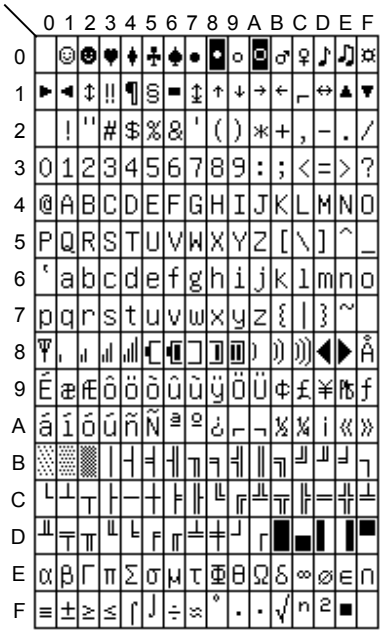


Table E-2: 大 ASCII 表

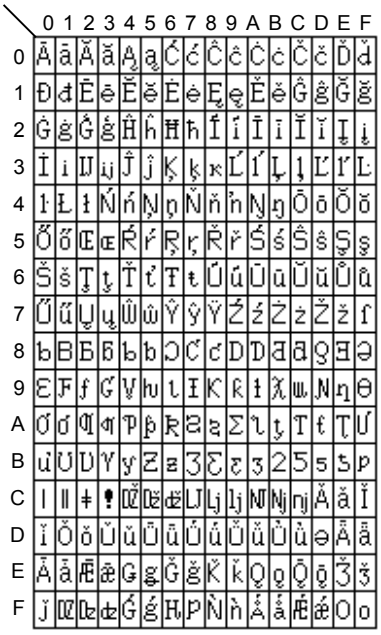


Table E-3: 大 ASCII 表