中文液晶显示器使用说明书

ZX240128GB

十分感谢您关注和使用我们的图形点阵型配带触摸屏液晶显示模块，欢迎您提出您的要求、意见和建议，我们将竭诚为您服务！您可以使用下列方式获取具体的技术咨询与服务。

北京中显电子有限公司

WWW.ZXLCD.COM

目 录

一、概述:

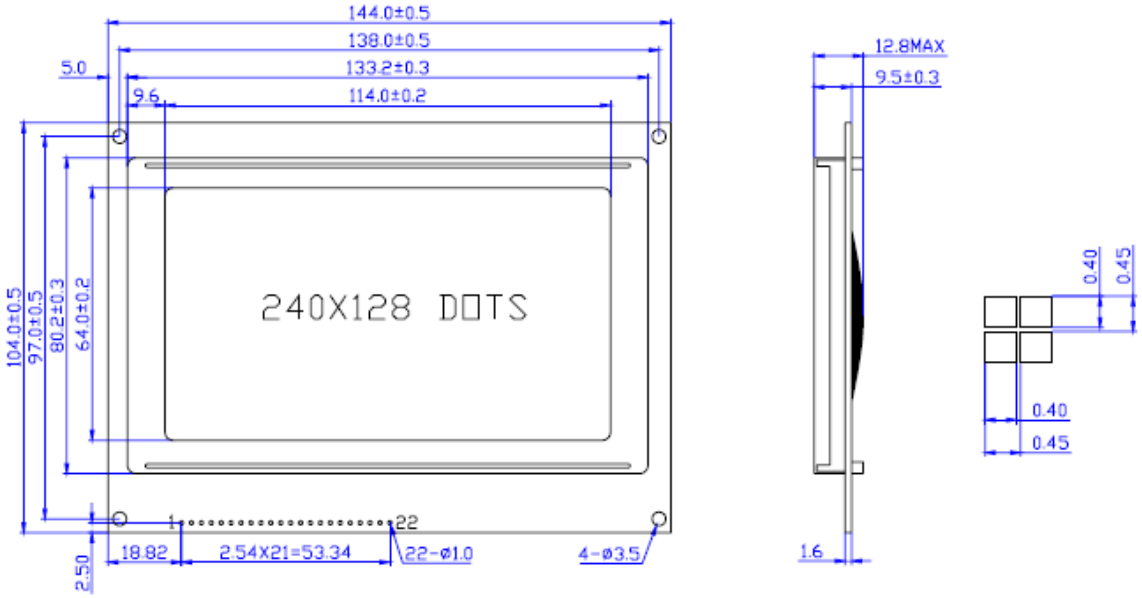
ZX240128GB中文字库液晶显示模块是一个中英文文字与绘图模式的点矩阵液晶显示模块，内建512KByte 的ROM 字形码，可以显示中文字型、数字符号、英日欧文等字母，并且内建双图层(Two Page)的显示内存。在文字模式中，可接收标准中文文字内码直接显示中文，而不需要进入绘图模式以绘图方式描绘中文，可以节省许多微处理器时间，提升液晶显示中文之处理效率。 此液晶显示模块除了支持8080/6800 系列之MCU 外，也提供4-Bit 或8-Bit 的数据总线接口。此液晶显示模块支持240×128点阵的LCD 面板，当字型为16×16 时，可秀出15(列)×8(行)个全型中文字，在字型方面有多种字号可供选择使用，如16×16、32×32、48×48、64×64 及不同比例的混合显示模式，同时内建的512Byte SRAM 提供了自行造字的功能。

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 Byte 的显示RAM (Display Data 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，其中标准繁体中文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(16×16)与半角(8×16)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32×32、48×48 或64×64，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。

本产品可根据客户要求配带触摸屏功能!

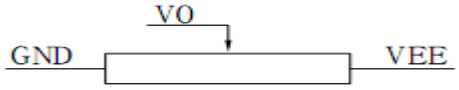
※注意：客户在使用ZX240128GB模块时，可选择中文字库控制器或图形点阵控制器，但需出厂前设定。此说明书只对中文字库控制器(简体中文)。同时本款型号可用于的控制器类型如RA8822、RA8803、RA8806三种，其中RA8806是一款抗干扰功能超强，性能稳定，建议后期客户开发时选用此芯片!

二、模块结构图:



液晶显示器在上电之后要先对整个模块进行一次复位，即在 RESET 脚加上一个低电平一段时间，(ZX240128GB 复位时间需要较长,建议 500 毫秒)，然后拉高到高电平，再开始对模块进行初始化操作。用户自己拉个 IO 口进行软件复位比较好

2.1、板载负压电路客户只需要接一个 10-20K 的电位器调到一个合适的值，其电位器的一端可接 GND 地，也可接 VDD，



三、主要参数:

项　　目	参　考　值
逻辑工作电压（Vdd）	+4.5～+5.5V
LCD 驱动电压（Vdd-Vo）	14.5V
工作温度（Ta）	－20～70℃（宽温）
储存温度(Tsto)	－30～80℃（宽温）
工作电流（背光除外）	30.0mA(max)

四、接口引脚说明: (以下接口定义为中文字库控制器)

引脚	名　称	方向	说　　明
1	FG	--	大地。
2	VSS	--	电源负端(0V)。
3	VDD	--	电源正端(+5V)。
4	V0	I	LCD 驱动电压(外接可调电阻，调节对比度)
5	/WR(R/W)	I	6800 系列：读/写脚(R/W)；H：读，L：写。 8080 系列：写入脚(/WR)，低有效。
6	/RD(EN)	I	6800 系列：使能脚(EN)，高有效。 8080 系列：读入脚(/RD)，低有效。
7	/CS	I	当/CS 为 Lo 时，模块处于致能，可接受指令，反之不可接收指令。
8	RS	I	H：存取 DDRAM；L：存取缓存器。
9	/RESET	I	复位信号，低有效。
10～17	DB0～DB7	I/O	负责在 LCM 及微处理器之间做资料料传送与接收。 当 MCU 为 4 位模式下，高位 DB[7..4]需浮接。
18	BUSY	0	用以回应模块内部的执行使用状况，可设成高或低电平触发。
19	VEE	--	LCD 驱动负压电源输出。
20	LEDA	--	背光电源正端(+5V)。
21	LEDK	--	背光电源负端(0V。)
22	INT	0	用以回应模块内部的中断状况，可设成高或低电平触发。

说明: 1)DB 选择点与 1 处(VDD)连接时，为 8Bit；与 2 处(VSS)连接时，为 4Bit。
2)MI 选择点与 1 处(VDD)连接时，为 6800 系列；与 2 处(VSS)连接时，为 8080 系列。

五、微控制器(MPU)说明:

1、8080 系列MPU说明:

ZX240128GB 与8080 兼容系列的MPU 接口示意图，此时LCM 将只接受与8080 系列兼容的MPU 所传送出来的控制信号。

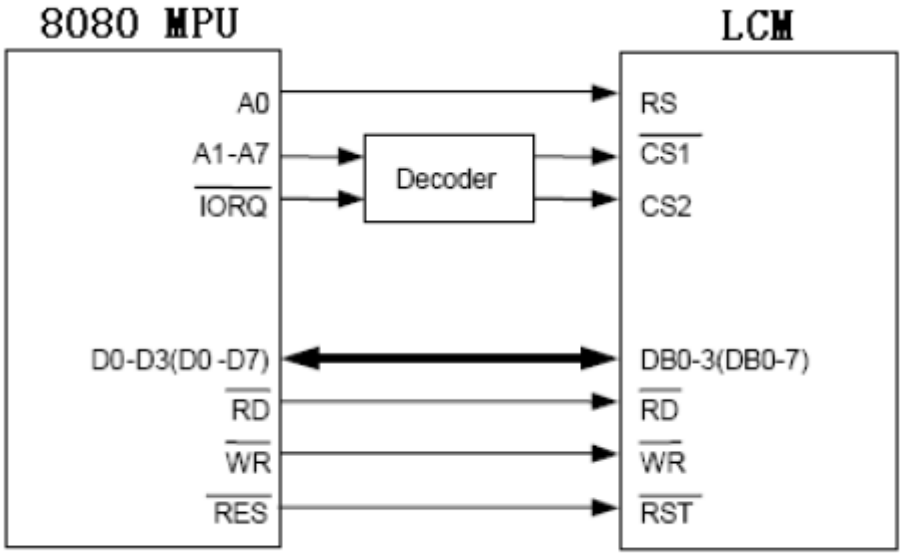


图5.1: 8080 (4/8-bit) MPU 与LCM的界面图

图5.2是8080系列MPU与LCM间的系统时序图，在LCM的定义中，RS 为“L”时，是表示对缓存器下命令，也就是对LCM的缓存器进行读写的动作(Register Access Cycle)，而RS 为“H”时是表示对Display RAM 进行Data 读写的动作(Data Access Cycle)。不论是8080 或6800，“RS”Pin 通常接到MPU 的Address Pin“A0”，8080 系列MPU 与6800 最大的不同是Read、Write 的控制信号是分开的，RD 为Low 时是进行读取动作，WR 为Low 时是进行写入动作，至于读写的目的地则由RS 决定。下图5.2表示如果是对缓存器进行读取动作，MPU 必须透过数据总线先送出缓存器的地址，然后才能在数据总线上读取缓存器的数据，如果是对缓存器进行写入动作，MPU 必须透过数据总线先送出缓存器的地址，然后再送出要写入的数据。当8088 MPU 对LCM Display RAM 进行数据的读取动作，MPU 能直接在数据总线上读取Display RAM 的数据，如果8088 MPU 对Display RAM 进行数据的写入动作，MPU 则直接在数据总线上送出要写入的数据。

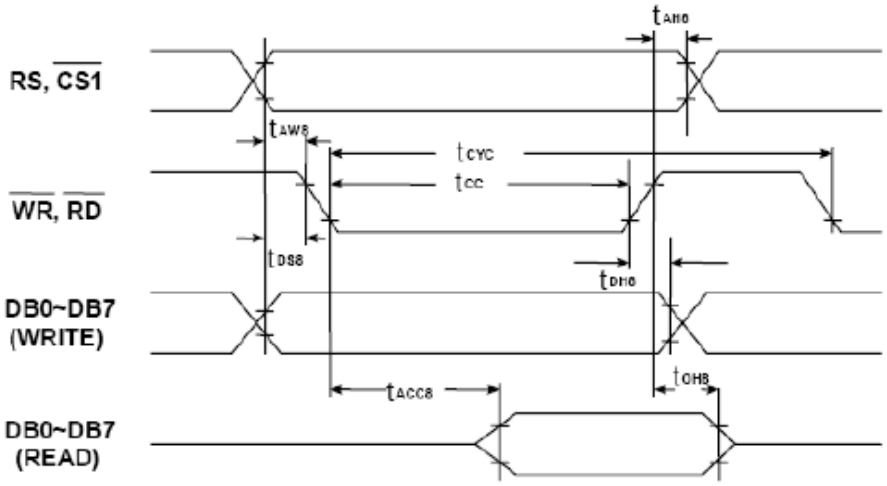


图 5.2: 8-Bit 8080 MCU 对 LCM 缓存器/DATA 进行读取/写入动作

Signal	Symbol	Parameter	Rating		Unit	Condition
			Min	Max		
A0, R/W#, CS1#	t_{AH6}	Address hold time	10	--	ns	System Clock: 8MHz Voltage: 3.3V
	t_{AW6}	Address setup time	63	--	ns	
	t_{CYC6}	System cycle time	800	--	ns	
DB0 to DB7	t_{DS6}	Data setup time	63	--	ns	
	t_{DH6}	Data hold time	10	--	ns	
	t_{ACC6}	Access time	--	330	ns	
	t_{OH6}	Output disable time	10	--	ns	
EN	t_{EW}	Enable pulse width	400	--	ns	

3、4Bit/8Bit 的MPU 接口:

此中文字库液晶显示模块除了支持8080 和6800 两大系列兼容的MPU 接口外，也可以设定MPU 上的数据总线接口是4-Bit 或是8-Bit，出厂时默认8-Bit 接口。因为控制IC 内部的缓存器大多是8-Bit 的架构，因此如果使用4-Bit 的数据总线接口，MCPU 将会花较多的周期(Cycle)去存取内部的缓存器。 当选择4-bit MCU 作传输模式时，中文字库液晶显示模块的MCU 接口只有用到数据总线的DB3～DB0，而没有用到的DB7～DB4 则不必理会(当成NC Pin)，同时每一个八位的指令或资料将被分为两个Nibble (4-Bit) 依序透过数据总线的DB3～DB0 进行传送，第一次先透过总线(DB3～DB0) 传送资料的较高位Bit[7..4]，第二次再透过总线(DB3～DB0) 传送资料的较低位Bit[3..0]。

六、中文字型 ROM:

此中文字库液晶显示模块内建有 512KByte 的 16×16 中文显示字型 ROM(Font ROM) 与 8×16 的 ASCII 半型字型。除了内建的 8×16 和 16×16 的字号外，还提供字型放大的功能，可利用 REG[F1h]的设定，将显示字号放大到 32×32、48×48 或 64×64。控制 IC 分带繁体字库 IC 和带简体字库 IC，其中标准繁体中文 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。

缓存器[F0h]是用来设定与字型 ROM 相关的功能，当使用带繁体字库 IC 时，必须将 Bit[5..4]设成“01”才能正确显示繁体字型，当使用带简体字库 IC 时，必须将 Bit[5..4]设成“10”才能正确显示简体字型。

REG [F0h] Font Control Register (FNCR)

Bit	Description	Text/Graph	Default	Access
7	字型 **ROM** 的转换电路控制 1：致能 0：Bypass (客户建立字型 ROM 时使用)	--	1h	R/W
6	字型 **ROM** 的地址空间选择 当 bit5~4 设定 "00"→ ROM Mode0，该位可以用来选择上或下的 256KB ROM 的地址空间。 1：选择下部 256KB 字型 ROM 0：选择上部 256KB 字型 ROM	--	0h	R/W
5-4	字型 **ROM** 的语系选择 0 0：选择简体（GB）字型 (256KB, Mode0) 0 1：选择繁体（BIG5）字型 (512KB, Mode1) 1 0：选择简体（GB）字型 (512KB, Mode2)	--	00h	R/W
2	强制为 **ASCII** 解码 **(注 1)** 1：所有输入的 Data，都以 ASCII 解码(00~FFh) 0：先检视输入 Data 的第一个字节介于： 00~9Fh，视为 ASCII (半角字) A0~FFh，视为 GB/BIG5 (全角字)	Text	0h	R/W

注意：中文内码不论是 GB 或 BIG5 码都是由两个 Byte 组成，但是英文及一些符号 ASCII 码只由一个 Byte 组成(00h～FFh)，通常 LCM 将送到 Display RAM 的 Data(00h～9Fh)视为 ASCII 码，也就半角文字(8×16)，大于等于“A0h”的视为全角码(如繁简中文)的高位，必须再送一次低位内码，才能显示全角字型。如果使用者有用到 A0h～FFh 的 ASCII 码，则 MPU 在送 Data(ASCII 码)到 Display RAM 之前必须将缓存器[F0h]的 Bit2 设成“1”。

七、功能应用介绍:

1、文字模式设定:

1) 文字显示:

ZX240128GB LCM的文字模式可以支持全角(中文或英文)及半角(英文)的显示，全角文字是以16×16 的点矩阵组成，半角文字是8×16 的点矩阵组成，如图7.1所示。而图7.2是全角(中文)及半角(英文)文字的混和显示:

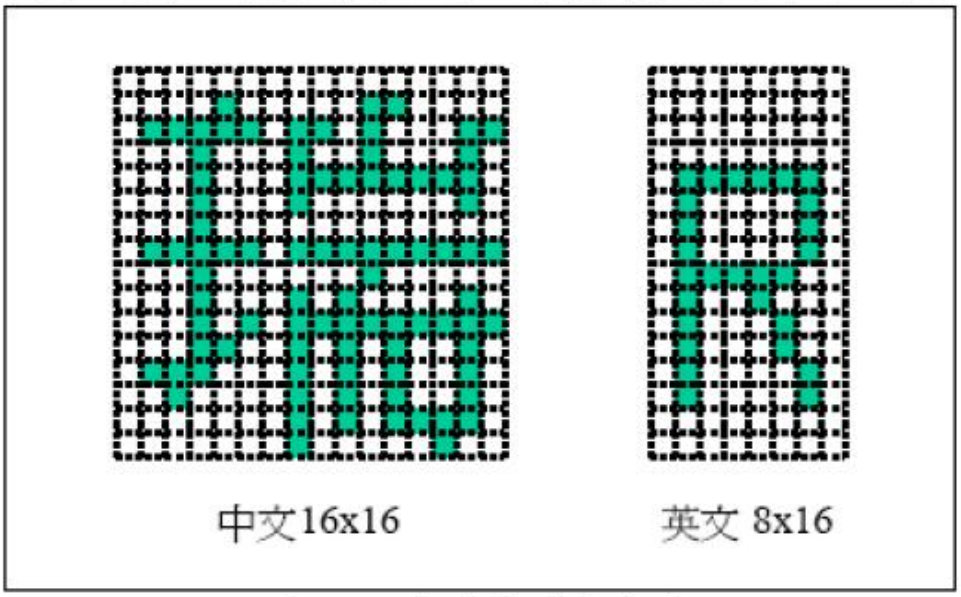


图7.1: 全角与半角文字

图7.2: 全角与半角文字的混和显示

ZX240128GB 的中文显示方式与传统的LCM控制器不同，传统的LCM控制器是在绘图模式下，以Bit-Map 的方式去绘出中文，ZX240128GB 的中文显示方式则是在文字模式，直接输入中文字码(GB 或BIG5 码)，就可以在光标所在位置显示中文。因为中文字码占两个Byte，所以如果MPU 接口是8-Bit，则MPU 必须分两次将中文字码(High Byte & Low Byte)写入LCM，而英文或数字码只占一个Byte，因此只要将内码一次写入LCM既可。

表7.1为图7.2所示之全角(中文)与半角文字的字型码，下面例题程序就是说明如何显示图7.2的画面。

表7.1：文字码的对照表（BIG5）

显示字型	字型码
中	A4A4
文	A4E5
文	A4E5
字	A672
/	2F
图	B9CF
形	2F
L	4C
C	43
D	44
控	B1B1
制	A8EE
器	BEB9

```
例 题:
MOV A,#A4H ; 写入“中”的字型码High Byte
CALL RegData_Write
MOV A,#A4H ; 写入“中”的字型码Low Byte
CALL RegData_Write ; 在光标所在位置会显示“中”
MOV A,#A4H ; 写入“文”的字型码High Byte
CALL RegData_Write
MOV A,#E5H ; 写入“文”的字型码Low Byte
CALL RegData_Write ; 在光标所在位置会显示“文”
:
:
```

2)、: 粗体字之显示功能:

ZX240128GB 模块的中英文显示都可以秀出粗体字显示效果，设定缓存器[10]的 bit4 为“1”就可以显示粗体文字。

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

Bit	Description	Text/Graph	Default	Access
4	设定粗体字型(仅文字模式适用) 0：正常字型 1：粗体字型	Text	1h	R/W

图7.3: 粗体字的显示

2、绘图模式设定： ZX240128GB 中文图形液晶显示模块是以字符映像（bit map）填入图形资料在Display RAM 上，图7.4 说明进入绘图模式时，缓存器要如何设定：

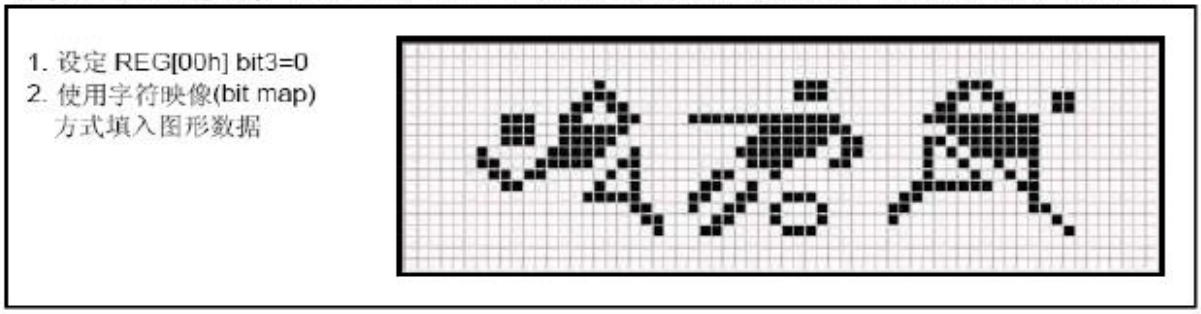


图7.4：绘图模式的显示

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

Bit	Description	Text/Graph	Default	Access
3	选择显示工作模式 1：文字模式，写入的数据会被视为是 GB/BIG/ASCII 等字码。 0：绘图模式，写入的数据会被视为是 Bit-Map 的模式。	--	1h	R/W

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

Bit	Description	Default	Access
7	图形模式时，光标自动移位的方向选择 1：先水平移动再垂直移动 0：先垂直移动再水平移动	1h	R/W

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

Bit	Description	Text/Graph	Default	Access
7	设定当数据读出 **DDRAM** 时，光标是否自动移位。 1：致能(自动移位) 0：禁能(不自动移位)	Text/Graph	0h	R/W
3	此位用来设定当数据写入 **DDRAM** 时，光标是否自动移位 1：致能(自动移位) 0：禁能(不自动移位)	Text/Graph	1h	R/W

当模块在显示图形的时候，是以字符映像(Bit Map)的方式写入DDRAM，若DDRAM的某个位置被填满为‘1’时，相对于LCD 面板的位置会被显示出亮点，由图7.5可看出，在DDRAM 上所储存之像素数据，会对应到显示屏幕(LCD)上，而构成文字、符号或图形之显示效果。

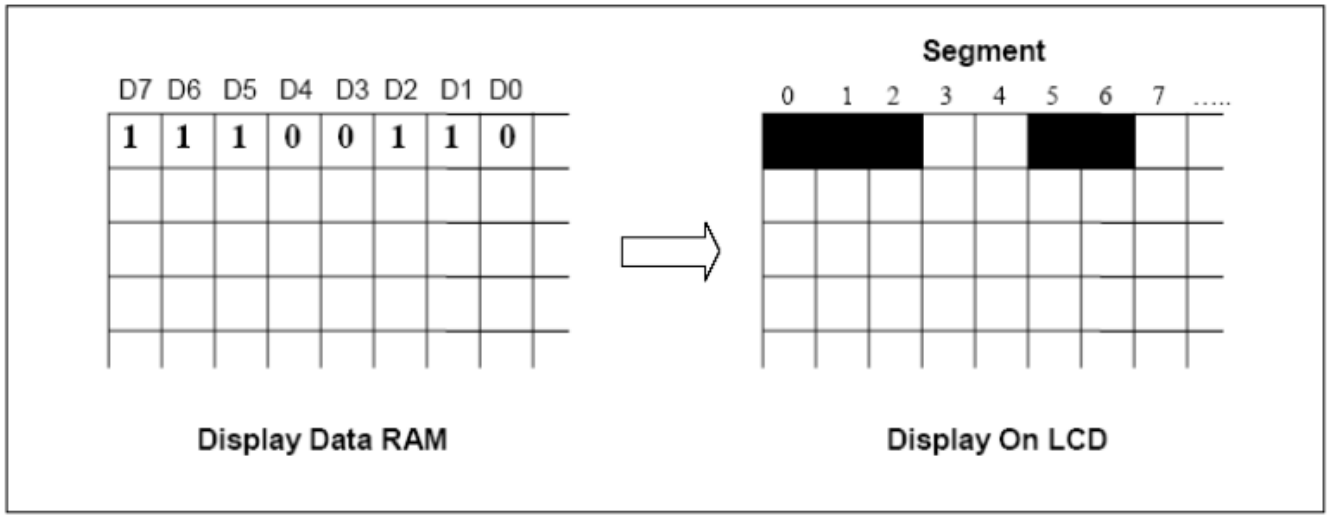


图7.5: Display Data 到LCD 显示的映像

以下程序是以图7.5做例子，用绘图模式在LCD Panel 的左上角秀出Pattern:

例 题: (8051-ASM)

```
MOV A, #60h ; 选择光标设定缓存器 (CPXR)
CALL RegAddr_WRITE
MOV A, #00h ; 设定坐标 X=0
CALL RegAddr_WRITE
MOV A, #70h ; 选择光标设定缓存器 (CPYR)

CALL RegAddr_WRITE
MOV A, #00h ; 设定坐标 Y=0
CALL RegAddr_WRITE ; 设定光标位置为(0,0)
MOV A,#E6H ;在LCD Panel 的左上角秀出“E6”的图形Pattern
CALL RegData_Write
```

例 题: (8051-C)

```
LCD_CmdWrite(0x60,0x00); // 设定坐标 X=0
LCD_CmdWrite(0x70,0x00); // 设定坐标 Y=0
LCD_DataWrite(0xE6); //在LCD Panel 的左上角秀出“E6”的图形Pattern
```

在绘图模式下，缓存器[12h]的Bit7 用来选择光标的移动是先水平移动再垂直移动或是先垂直移动再水平移动，如图7.6。

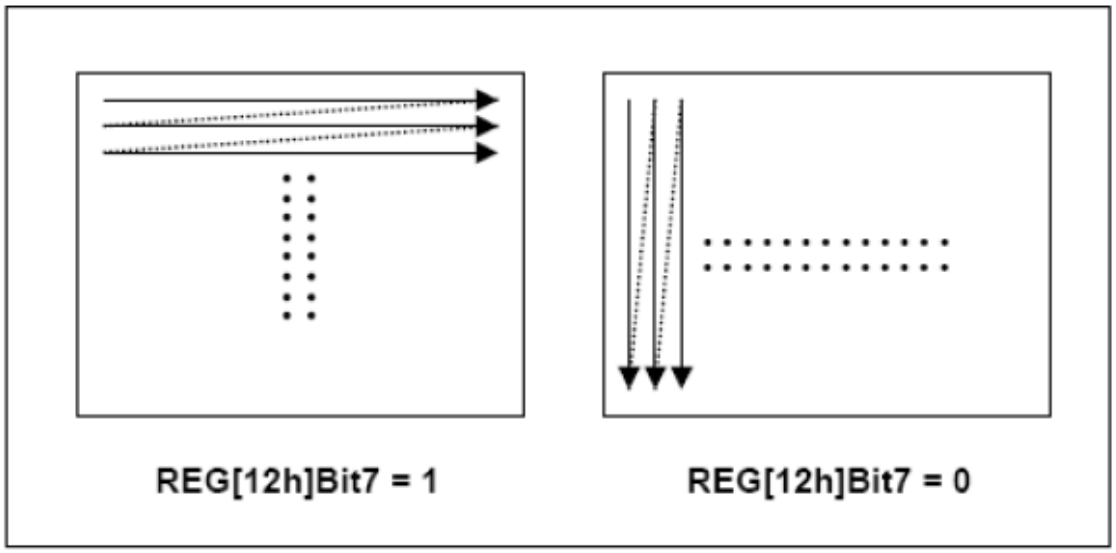


图7.6: 选择光标的移动

图7.7范例:

```
MOV A, #12h ; 选择缓存器 [12h] (MAMR)
CALL RegAddr_WRITE
MOV A, #91h ; Bit7=1, 先水平移动再垂直移动
CALL RegAddr_WRITE
MOV A,#11H ;在LCD Panel 的左上角秀出“11”的图形Pattern
CALL RegData_Write
MOV A,#22H ;在LCD Panel 的左上角秀出“22”的图形Pattern
CALL RegData_Write
MOV A,#33H ;在LCD Panel 的左上角秀出“33”的图形Pattern
CALL RegData_Write
MOV A,#44H ;在LCD Panel 的左上角秀出“44”的图形Pattern
CALL RegData_Write
```

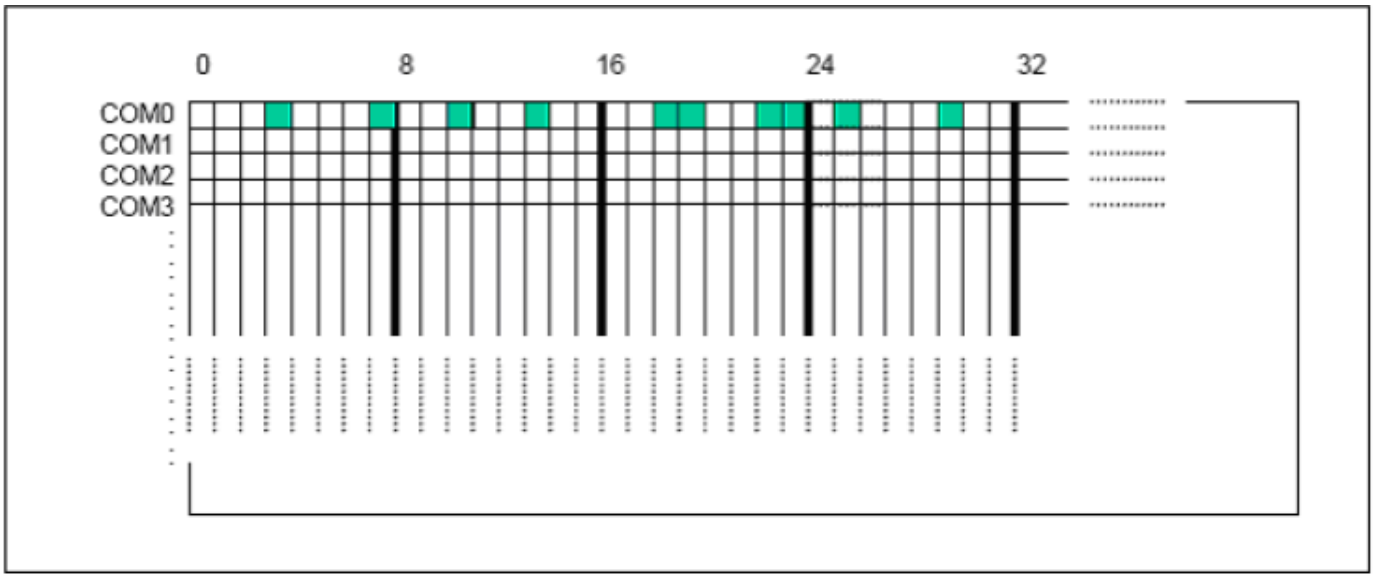


图7.7: 光标先水平移动再垂直移动

图7.8范例:

```
MOV A, #12h ; 选择缓存器 [12h] (MAMR)
CALL RegAddr_WRITE
MOV A, #11h ; Bit7=0, 先垂直移动再水平移动
CALL RegAddr_WRITE
MOV A,#11H ;在LCD Panel 的左上角秀出“11”的图形Pattern
CALL RegData_Write
MOV A,#22H ;在LCD Panel 的左上角秀出“22”的图形Pattern
CALL RegData_Write
MOV A,#33H ;在LCD Panel 的左上角秀出“33”的图形Pattern
CALL RegData_Write
MOV A,#44H ;在LCD Panel 的左上角秀出“44”的图形Pattern
CALL RegData_Write
```

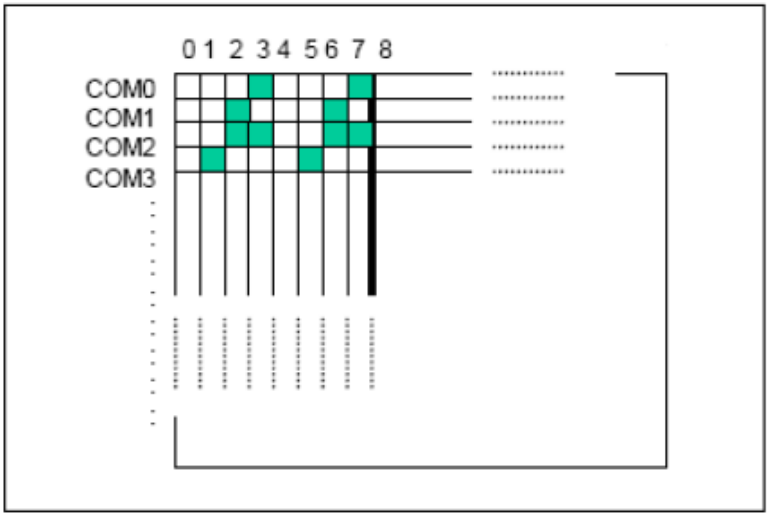


图7.8: 光标先垂直移动再水平移动

在绘图模式下，若要读取Display RAM 的数据时，也是由缓存器[12h]的Bit7用来选择光标的移动是先水平移动再垂直移动或是先垂直移动再水平移动，如图7.6。不论写入或读取Display RAM 的数据都必须注意光标的设定是否有自动加一的功能，也就是缓存器[10h]的Bit7 与Bit3。如图7.9是代表缓存器[12h]Bit7=1

(先水平移动再垂直移动)时 Display RAM 数据的读取方向（以 ZX240128GB 显示模块为例）

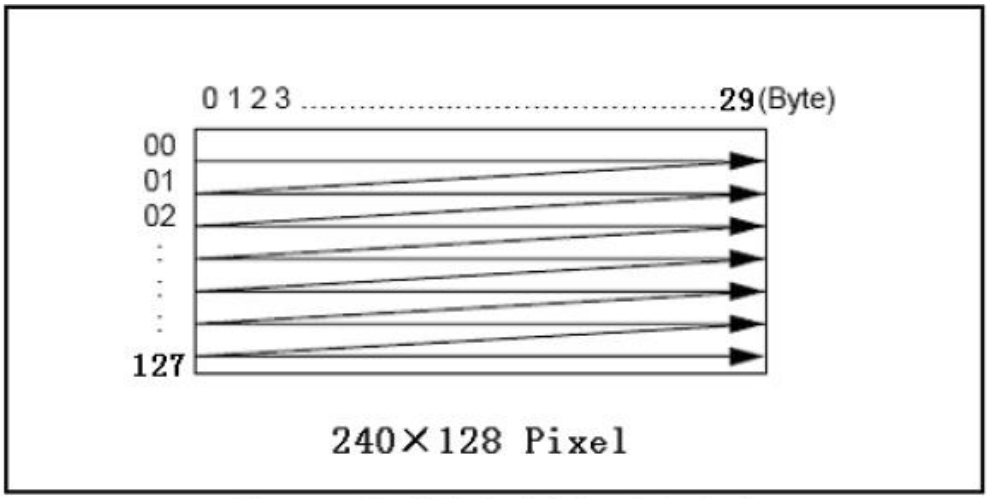


图7.9：图形模式时数据读取方向

3、闪烁与反白显示：

1）闪烁显示：

下图7.10说明要闪烁显示时，缓存器要如何设定：

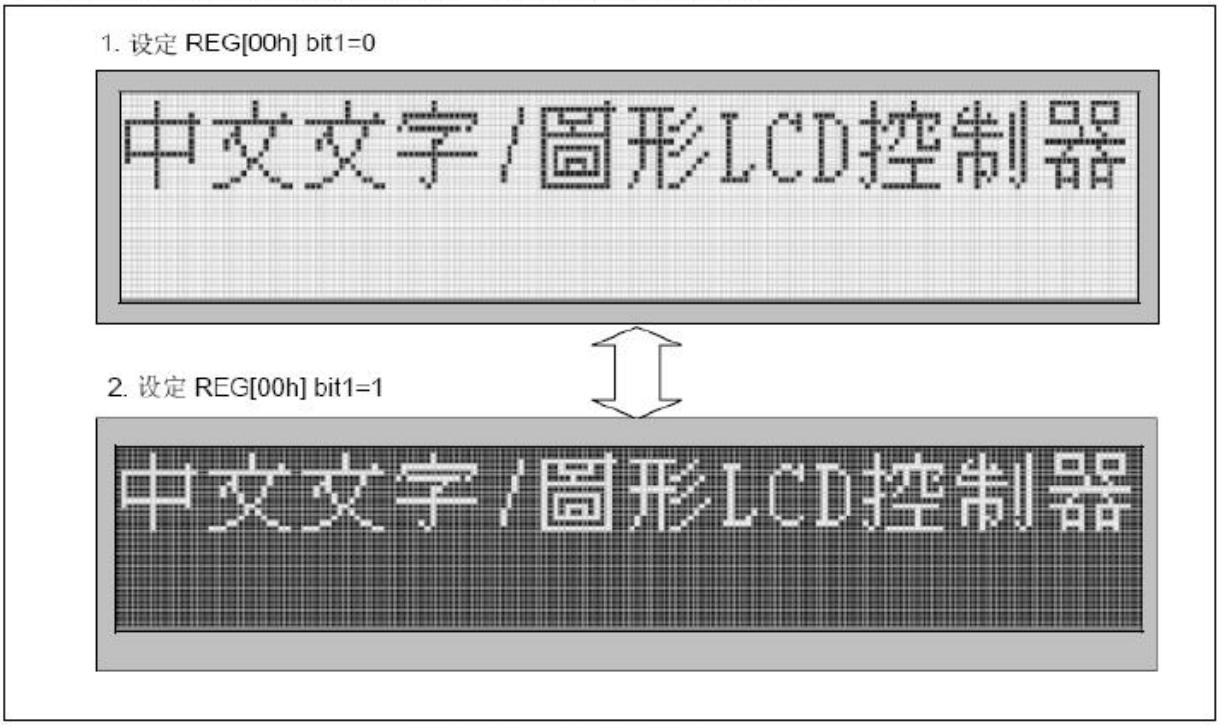


图7.10：屏幕闪烁

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

Bit	Description	Text/Graph	Default	Access
1	闪烁模式选择 0：正常显示 1：整个屏幕闪烁，闪烁时间由缓存器[80h]BTR 来设定	Text/Graph	0h	R/W

2）屏幕反白：

如果要将LCD 画面全部反白只要设定缓存器[00]的Bit0 既可，图7.11说明要反白显示时，缓存器要如何设定：

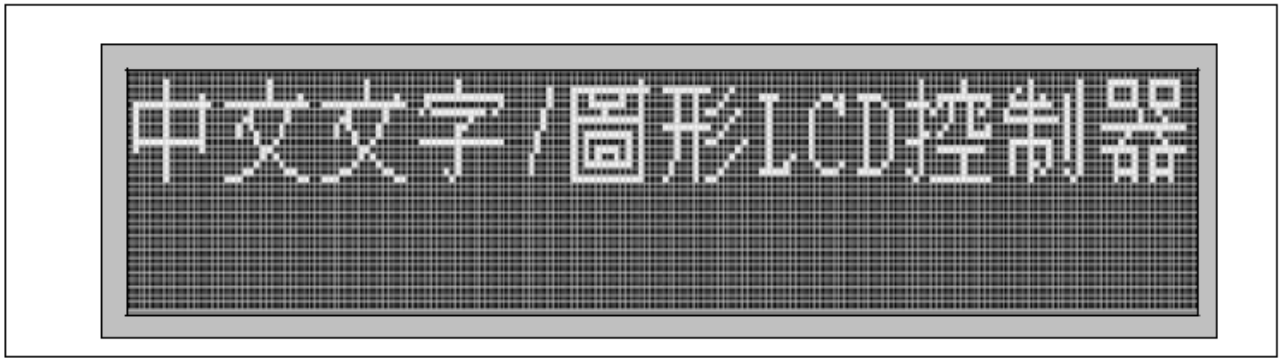


图7.11: 屏幕反白

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

Bit	Description	Text/Graph	Default	Access
0	屏幕反白模式选择 1：正常显示 0：全屏幕反白显示，DDRAM 内的数据会被全部反相。	Text/Graph	1h	R/W

3) 文字反白:

如果要将LCD 画面秀出反白的字体只要设定缓存器[10]的Bit5 既可，图7.12说明要反白显示时，缓存器要如何设

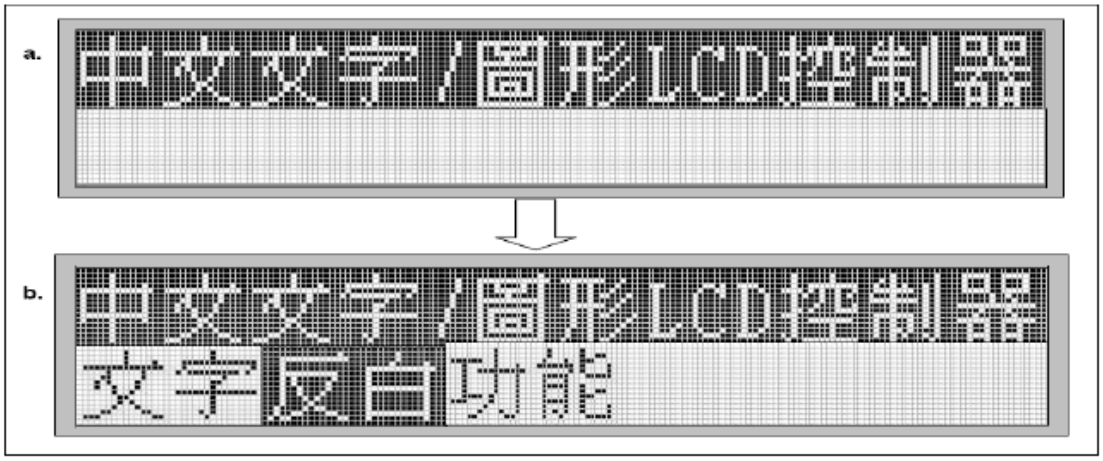


图7.12反白显示

(a)

1. 设定缓存器[10h] bit5=0
2. 写入"中文文字/图形LCD 控制器"的BIG5 码，然后可显示出"中文文字/图形LCD 控制器"

(b)

3. Hold on (a)
4. 设定缓存器[10h] bit5=1
5. 写入"文字"的BIG5 码，LCD 就可显示出"文字"

6. Hold on
7. 设定缓存器[10h] bit5=0
8. 写入"反白"的BIG5 码，LCD 就可显示出"反白"字样
9. Hold on
10. 设定缓存器[10h] bit5=1
11. 写入"功能"的BIG5 码，LCD 就可显示出 "功能"

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

Bit	Description	Text/Graph	Default	Access
5	储存 MPU 进来数据(正相/反相)于 DDRAM 1：直接储存数据于 DDRAM 中 0：存入相反的数据于 DDRAM 中	Text	1h	R/W

4、中/英文文字对齐:

由于英文字体与中文字体所占的宽度不一样，因此在显示中文英文都有的画面时必须考虑整体显示效果，RT240128GB可以设定中文英文显示时不同行的显示效果以决定文字是否对齐，图7.13说明要表现出中英文文字“对齐”之情形时，缓存器要如何设定:

1. 设定 缓存器 WCCR,ALG=1
2. 写入“中文文字/图形 LCD 控制器” 两次，则屏幕会秀出”中文文字/图形 LCD 控制器” ←上下两行文字对齐

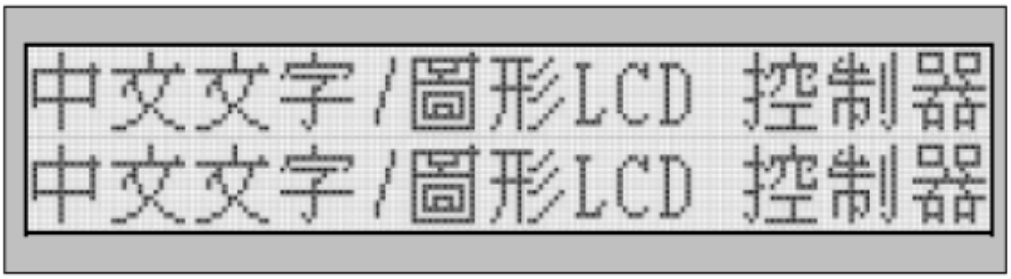


图7.13: 文字对齐的显示范例

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

Bit	Description	Text/Graph	Default	Access
6	中/英文字对齐 1：致能 0：禁能 此功能仅在文字模式时有效，可以将全角与半角混合显示时作对齐调整。	Text	1h	R/W

图7.14：说明要表现出中英文文字“不对齐”之情形时，缓存器要如何设定：

1. 设定 缓存器 WCCR,ALG=1
2. 写入“中文文字/图形 LCD 控制器”
3. 设定 缓存器 WCCR,ALG=0
4. 写入“中文文字/图形 LCD 控制器” ←上下两行文字没对齐

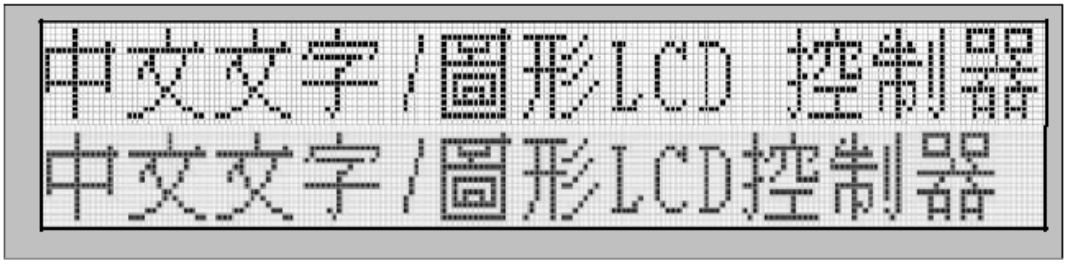


图7.14：文字不对齐的显示范例

5、屏幕显示on/off 设定：

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

Bit	Description	Text/Graph	Default	Access
2	设定屏幕显示为开启或关闭，此位用来控制连接到 LCD 驱动 IC 接口的 “DISPOFF#” 信号 1：“DISPOFF” 讯号输出 High(屏幕开启) 0：“DISPOFF” 讯号输出 Low(屏幕关闭)	Text/Graph	0h	R/W

6、光标On/Off 设定：

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

Bit	Description	Text/Graph	Default	Access
2	光标显示 **On/Off** 设定 1：设定光标显示 On 0：设定光标显示 Off	Text/Graph	0h	R/W

7、光标位置与移位设定：

1) 光标位置：

缓存器[60h]CPXR 的Bit[5..0]用来设定光标的Segment 地址，RT240128GB支持240x128 的Panel Size，但是光标的Segment 地址是以每8-Bit 为单位，例如，想在Panel 的左上角秀出“控”，则必须设定光标缓存器CPXR = 00h，CPYR = 00h；又例如想在Panel 的左上角第三个全角位置秀出“制”，则必须设定光标缓存器CPXR = 04h，CPYR = 00h，同理，想在Panel 的左上角第二行第一个全角位置秀出“器”，则必须设定光标缓存器CPXR = 00h，CPYR = 10h，请参考图7.15。

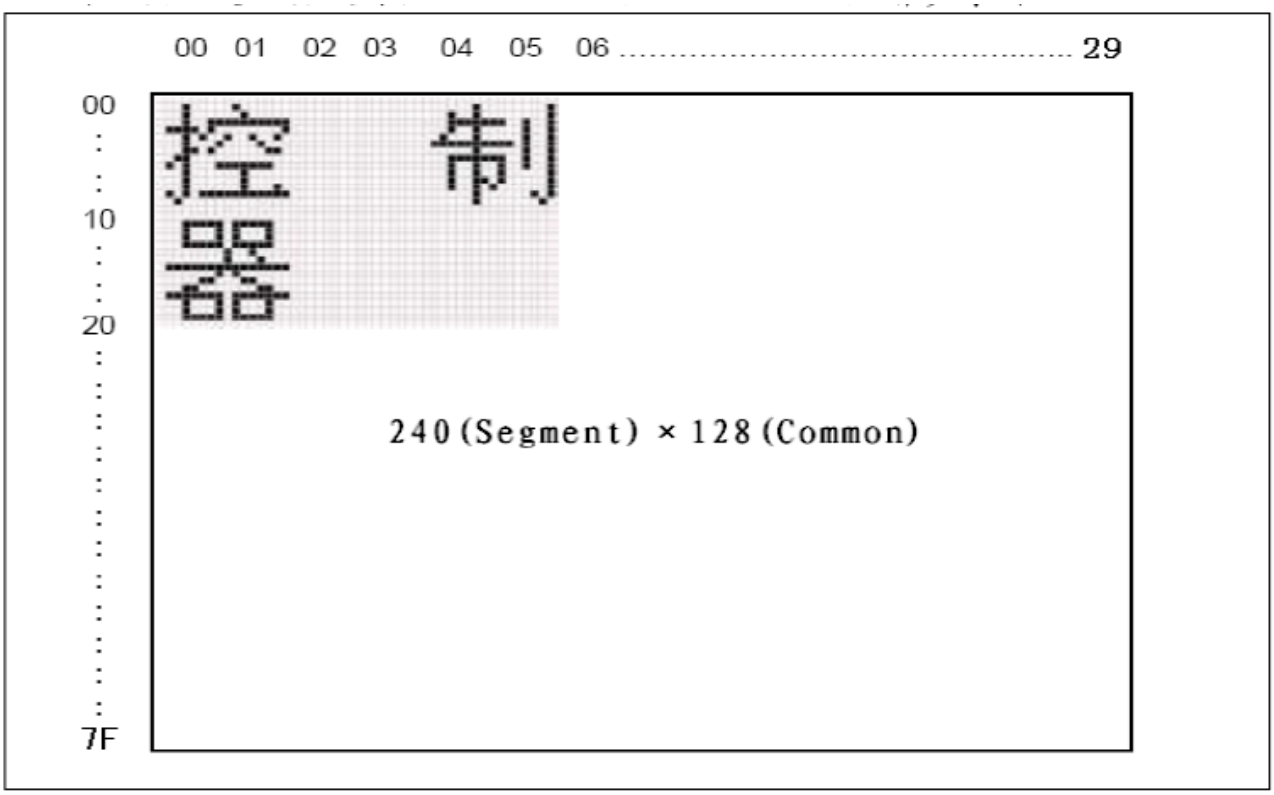


图7.15: 光标位置设定的显示范例

REG [60h] Cursor Position X Register (CPXR)

Bit	Description	Default	Access
5-0	设定光标 **Segment** 地址	0h	R/W

REG [70h] Cursor Position Y Register (CPYR)

Bit	Description	Default	Access
7-0	设定光标 **Common** 地址	0h	R/W

不论文字或是绘图模式,都是使用缓存器[60h]CPXR 与[70h]CPYR 来设定光标的地址。缓存器[60h]CPXR 与[70h]CPYR 的光标地址是属于绝对地址,不会因工作窗口大小而改变,也就是(0 ,0)始终是在屏幕的左上角。

2) 游标移位:

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

Bit	Description	Text/Graph	Default	Access
7	设定当数据读出 **DDRAM** 时，光标是否自动移位。 1：致能(自动移位) 0：禁能(不自动移位)	Text/Graph	0h	R/W
3	此位用来设定当数据写入 **DDRAM** 时，光标是否自动移位 1：致能(自动移位) 0：禁能(不自动移位)	Text/Graph	1h	R/W

8、光标闪烁设定:

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

Bit	Description	Text/Graph	Default	Access
1	光标闪烁控制 1：光标闪烁，闪烁时间由缓存器[80h] BTR 决定。 0：游标不闪烁	Text/Graph	0h	R/W

REG [80h] Blink Time Register (BTR)

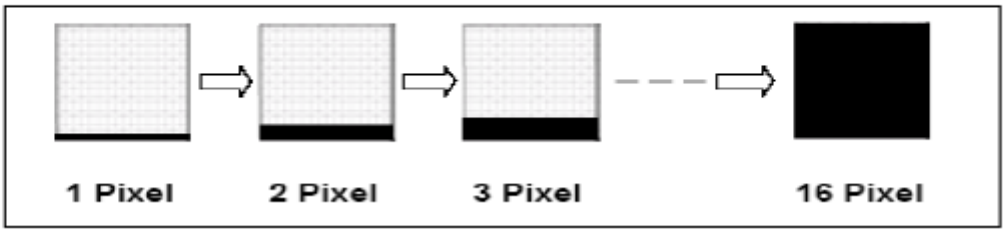
Bit	Description	Text/Graph	Default	Access
7-0	光标/屏幕闪烁时间设定 闪烁时间 = Bit[7..0] x (1/Frame_Rate)	Text/Graph	23h	R/W

如果Frame Rate = 60Hz，则1/Frame_Rate = 1/60Hz = 1.67ms，光标闪烁时间 = REG[80h] ×1.67ms，例如设定REG[80h] = 35h = 53(十进制)，因此光标闪烁时间 = 53×16.7ms = 885ms。

9、光标高度与宽度设定:

1) 光标高度:

ZX240128GB 在做文字显示时，有提供光标高度的设定，在显示文字时，光标的高度为一个Pixel 的高度，但依不同使用者的需要，提供了Pixel 的高度的设定，Pixel 的高度设定范围为(1～16)Pixel，使用者可依需求来决定光标的高度大小。



REG [11h] Distance of Words or Lines Register (DWLR)

Bit	Description	Text/Graph	Default	Access
7-4	设定光标高度	Text	0010h	R/W

2) 光标宽度:

此LCM在做文字显示时，有提供两种光标宽度的设定。第一种为REG[10h] bit0=0 时，光标的宽度将会固定为1 个Byte 的宽度(也就是8 个Pixel)。第二种为REG[10h] bit0=1 时，光标的宽度会随着所输入文字来做变化，例如当输入一个全角字时，文字后面的光标宽度会自动变为2 个Byte(也就是16 个Pixel)。当输入一个半角字时，文字后面的光标宽度会自动变为1 个Byte。

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

Bit	Description	Text/Graph	Default	Access
0	设定光标宽度 1：会随着输入的数据而变动光标宽度，当数据为半型时，光标为一个字节宽度(8 个 Pixel)，当数据为全型时，光标为二个字节宽度(16 个 Pixel)。 0：光标固定为一个字节的宽度(8 个 Pixel)	Text	0h	R/W

10、工作及显示窗口大小设定:

LCM在面板显示上，供使用者有两种窗口选择。一个是显示窗口(Display Window)，一个是工作窗口(Active Window)。显示窗口(Display Window)是实际LCD面板的大小，而工作窗口(Active Window)是在实际的显示窗口(Display Window)内设定比显示窗口小的子窗口。

例如面板大小为240×128，而它的显示窗口就为240×128。在显示窗口(240×128)内可依使用者需要，来设定工作窗口的大小，也就是子窗口的大小，子窗口也可在显示窗口内任意调整所要放置的地方。以下是相关的缓存器说明:

REG [21h] Display Window Right Register (DWRR)

Bit	Description	Default	Access
5-0	设定显示窗口(**Display Window**)右边位置 → **Segment-Right** (注 1) **Segment_ Right = (Segment Number / 8) – 1** RT240128GB:LCD Panel为240×128，此缓存器的值为： (240/8)-1=29=1Dh	xxh	R/W

REG [31] Display Window Bottom Register (DWBR)

Bit	Description	Default	Access
7-0	设定显示窗口(**Display Window**) 底边位置 → **Common_Bottom** **Common_Bottom = LCD Common Number –1** RT240128GB：LCD Panel为240×128，则此缓存器的值为： 128-1=127=7F	xxh	R/W

REG [41] Display Window Left Register (DWLR)

Bit	Description	Default	Access
7-0	设定显示窗口(**Display Window**) 左边位置 → **Segment-Left** (注 1) 通常将此缓存器的值设定为“00h”。	xxh	R/W

REG [51] Display Window Top Register (DWTR)

Bit	Description	Default	Access
7-0	设定显示窗口**(Display Window)** 顶边位置 → **Common-Top** (注 1) 通常将此缓存器的值设定为 "00h"。	xxh	R/W

注1: 光标地址应设定在显示窗口的范围内，因此缓存器[60h, 70h]、[B0h, B1h]与[21h, 31h, 41h, 51h]的设定必须遵照以下的规范:

1. DWRR ≥ AWRR ≥ CPXR ≥ AWLR ≥ DWLR
2. DWBR ≥ AWBR ≥ CPYR ≥ AWTR ≥ DWTR

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

Bit	Description	Default	Access
5-0	设定工作窗口**(Active window)**右边位置 → **Segment-Right** (注 2)	xxh	R/W

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

Bit	Description	Default	Access
7-0	设定工作窗口**(Active window)** 底边位置→ **Common-Bottom** (注 2)	xxh	R/W

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

Bit	Description	Default	Access
5-0	设定工作窗口**(Active window)**左边位置 → **Segment-Left** (注 2)	xxh	R/W

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

Bit	Description	Default	Access
7-0	设定工作窗口**(Active window)** 顶边位置 → **Common-Top** (注 2)	xxh	R/W

注2: REG [20h, 30h, 40h, 50h] 可作为换行/换页的功能，可让使用者利用这4 个Register 自行设定一个区块为工作窗口(Active Window)。当数据超过窗口的右边界REG [20h, 30h, 40h, 50h]所设定的值，光标会自动换行(也就是光标移到工作窗口的左边界REG[40h]所设定的值)，继续将数据写入。当数据写入到工作窗口的右下角时（REG[20h]与[30h]所设定的值），会自动把光标移到工作窗口的的左上角(REG[40h, 50h]所设定的值)，继续的将数据填入窗口。

设定完工作窗口后，光标地址不会自动移到工作窗口的范围内，因为缓存器[60h]CPXR 与[70h]CPYR 的光标地址是属于绝对地址，不会因工作窗口大小而改变，也就是(0 ,0)始终是在屏幕的左上角，因此设定完工作窗口后想要进行秀字，必须先将光标地址设定在工作窗口的范围内，之后光标地址就只会在工作窗口的范围内移动。

11、行距设定:

在做文字显示时，提供了行距设定的功能，尤其是做中文显示时，每一行如果有适当的间隔，LCD 的显示画面看起来会比较舒适。行与行相隔的间距设定范围为 1～16 Pixel 的高度，使用者可依需求来决定行与行间距的大小，一旦设定后，当每填完一行的中文字，跳到下一行时，其行距会依照先前所设定的间距来显示。

REG [11h] Distance of Words or Lines (DWLR)

Bit	Description	Text/Graph	Default	Access
3-0	行距设定	Text	0010h	R/W

12、自动填入数据到 DDRAM:

REG [E0h] Pattern Data Register (PNTR)

Bit	Description	Text/Graph	Default	Access
7-0	**设定写入到 DDRAM 的数据** 当缓存器[F0h]的 bit3 为 '1'，内部将自动读取本缓存器[E0h] 的 Data，然后全部填写到 DDRAM 内，之后缓存器[F0h]的 bit3 被清除为 '0'。	Graph	0h	R/W

REG [F0h] Font Control Register (FNCR)

Bit	Description	Text/Graph	Default	Access
3	**重复写入 PNTR -- REG [E0h]的数据到 DDRAM** 1：开始写入 0：未动作 当 FDA 为 '1'， 液晶显示模块内部将自动读取 PNTR 的 Data，填写到 DDRAM 内(Range:[AISR, AICR] ~ [AXSR, AXCR])，之后此位会被自动清除为 '0'。	Graph	0h	R/W

13、屏幕更新频率设定:

REG [90h] Shift Clock Control Register (SCCR)

Bit	Description	Default	Access
7-0	设定 **XCK** 讯号周期 **SCCR = (SCLK x DBW) / (Column x Row x FRS)** SCLK：系统频率(System Clock) (单位：Hz) DBW：LCD 驱动器 的 Data Bus 宽度(单位：Bit) Column：LCD 面板的 Segment 大小(单位：Pixel) Row：LCD 面板的 Common 大小 (单位：Pixel) FRS：LCD 面板的 Frame Rate(单位：Hz) 限制条件 SYS_DW=0，LCD 的 Data Bus 为 4it，SCCR ≥ 4 SYS_DW=1，LCD 的 Data Bus 为 8it，SCCR ≥ 2	--	R/W

例题:

① 如果使用X'tal + PLL 的方式，系统频率(SCLK) = 8MHz

② LCD 驱动器 的Data Bus 宽度(DBW) = 8Bit

③ 使用240×128 Pixel 的LCD 面板，Column = 240，Row = 128

④ LCD 面板的Frame Rate 为70Hz

则SCCR = (8MHz×8) / (240×128×70) = 29.8

所以建议设定SCCR = 30 = 01Eh

如果设定数值太大，会造成LCD 屏幕闪动，有时会伴随水波纹产生，除降低数值外也可以提高系统频率来解决→调整REG[01h] 设定数值。为达到良好LCD 显示质量，使用者必须根据Panel、Driver 之特性与VLCD 电压、系统频率等等进行调整。

14、中断(Interrupt)与忙碌(Busy)设定:

ZX240128GB提供一中断信号线(INT)用来表示有三种中断讯息可能发生:

① 假如光标Segment 地址缓存器(CPXR)与Segment 中断地址缓存器(INTX)值相同，发生中断。

② 假如光标Common 地址缓存器(CPYR)与Common 中断地址缓存器(INTY)值相同，发生中断。

③ 触控屏幕侦测到被Touch，发生中断。

这三种中断都可以单独被致能或禁能，而中断的设定与中断讯息可有由缓存器[A0h] INTR 来控制与读取。此外LCM还提供一忙碌(Busy)信号线，用来表示内部DDRAM 与ROM 的存取状态是否因Busy 而暂时无法接收MPU 来的Command。以下是相关的缓存器说明:

REG [01h] Misc. Register (MISC)

Bit	Description	Default	Access
4	设定输出脚 -- 中断讯号(INT)与忙碌讯号 BUSY 的触发准位 1：设定高电位触发动作 0：设定低电位触发动作	1h	R/W

REG [A0h] Interrupt Setup & Status Register (INTR)

Bit	Description	Default	Access
7	**Key Scan** 中断旗标 1：Key Scan 有侦测到按键输入 0：Key Scan 无侦测到按键输入	0h	R
6	触控屏幕侦测 1：触控屏幕有侦测到触摸(Touch) 0：触控屏幕未侦测到触摸	0h	R
5	光标 **Column** 状态 1：光标的 Column 等于缓存器[B0h]INTX 0：光标的 Column 不等于缓存器[B0h]INTX	0h	R
4	光标 **Row** 状态 1：光标的 Row 等于缓存器[B1h]INTY 0：光标的 Row 不等于缓存器[B1h]INTY	0h	R
3	**Key Scan** 中断屏蔽控制 1：致能 Key Scan 中断 0：禁能 Key Scan 中断	0h	R/W
2	触控屏幕中断屏蔽 1：如果触控屏幕被侦测到，则产生中断输出 0：如果触控屏幕被侦测到，则不产生中断输出	0h	R/W
1	设定缓存器**[B0h]INTX** 是否发生中断 1：致能 INTX 中断 0：禁能 INTX 中断	0h	R/W
0	设定缓存器**[B1h]INTY** 是否发生中断 1：致能 INTY 中断 0：禁能 INTY 中断	0h	R/W

REG [B0h] Interrupt Column Setup Register (INTX)

Bit	Description	Default	Access
5-0	设定行**(Column)**中断地址 假如光标位置 X 缓存器(CPXR)=INTX，发生中断。	27h	R/W

REG [B1h] Interrupt Row Setup Register (INTY)

Bit	Description	Default	Access
7-0	设定列**(Row)**中断地址 假如光标位置 Y 缓存器(CPYR)=INTY，发生中断。	EFh	R/W

15、省电模式:

电源工作模式分两种: 正常模式(Normal Mode), 关闭模式(Off Mode)。

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

Bit	Description	Text/Graph	Default	Access
7-6	**电源模式(Power Mode)** 1 1: 正常模式(Normal Mode) 所有功能都可以使用(Available)。 0 0: 关闭模式(Off Mode) 除了唤醒(Wake-Up)电路工作外, 其它功能都被禁止。 当 Wake-Up 电路被触发, 将进入正常 模式。	--	3h	R/W

16、如何读取Font ROM 字型:

MPU 读取Font ROM 的Data, 只要将缓存器[02h]的Bit3 设为1, 然后写入两个Byte 的中文码, 之后连续读取的32Byte Data 就是该中文码相对映的Font Data, 如下图7.16的流程图。

REG [02h] Advance Power Setup Register (APSR)

Bit	Description	Default	Access
3	字型 **ROM** 的直接读取 1: 致能 0: 禁能	0h	R/W

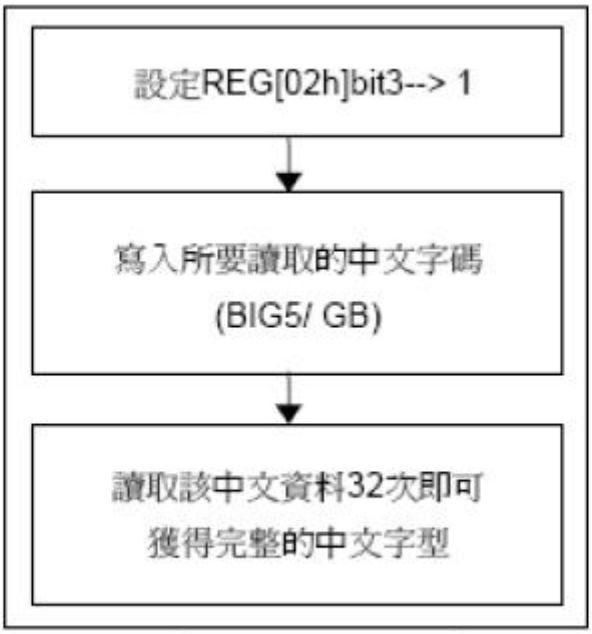


图7.16: 读取Font ROM 字型流程

ZX240128GB 中文液晶显示模块的全型字型为16×16 的Bitmap 所组成, 每个全型字型占用Font ROM 32Byte, 在MCU 读取Font ROM 的Data 时其顺序如下图7.17 所示。

Byte1	Byte17
2	18
3	19
4	20
5	21
6	22
7	23
8	24
9	25
10	26
11	27
12	28
13	29
14	30
15	31
16	32

图7.17: 读取Font ROM 字型Data 的顺序

17、字号放大设定:

此LCM模块内建有512KByte 的中文显示字型ROM(Font ROM)，全角16x16 中文与8x16 的ASCII半型字型。除了内建的8x16 和16×16 的字号外，还提供字型放大的功能，可利用REG[F1h]bit7～4 的设定，将显示字号放大到32×32 或48×48，64×64。下图7.18是表示16×16 的字型放大到32×32。

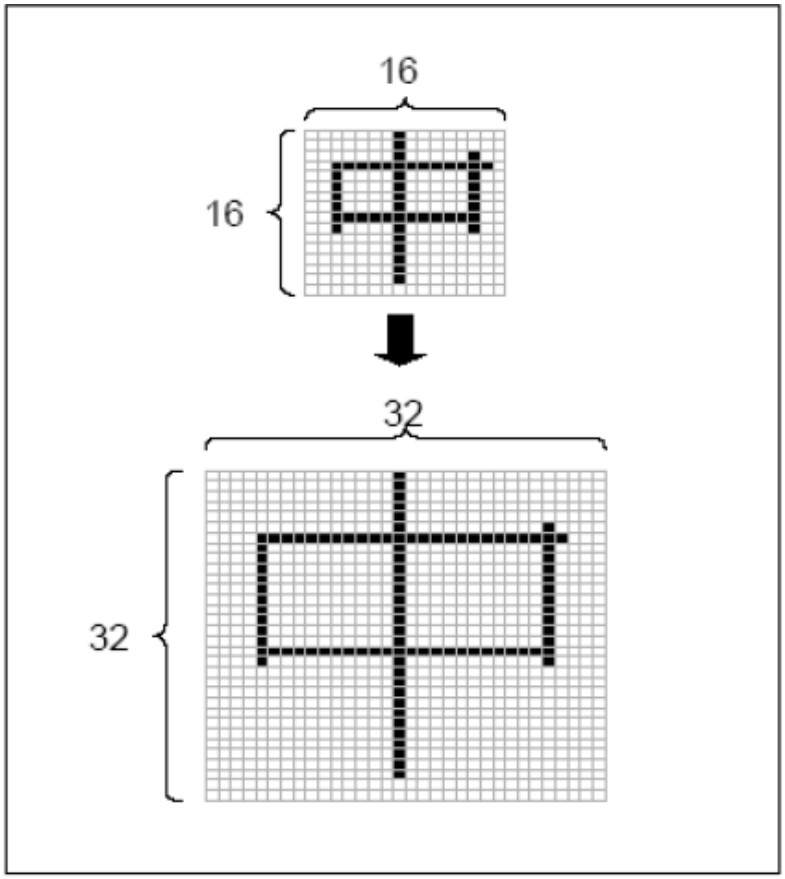


图 7.18: 字体放大

REG [F1h] Font Size Control Register (FVHT)

Bit	Description	Default	Access
7-6	设定字型水平的大小 0 0：一倍 0 1：二倍 1 0：三倍 1 1：四倍	0h	R/W
5-4	设定字型垂直的大小 0 0：一倍 0 1：二倍 1 0：三倍 1 1：四倍	0h	R/W

18、图层显示功能设定:

此LCM提供了双图层的功能，可经由缓存器REG[12h]来做设定，并提供4 种(OR, NOR, XOR和AND)图层显示模式，供使用者设定选用。实际的显示效果，请参考图7.19。

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

Bit	Description	Text/Graph	Default	Access
6-4	设定选择Display data RAM 的图层显示模式 001：只有显示Page1 的图层(单一上层显示模式) 010：只有显示Page2 的图层(单一下层显示模式) 011：同时显示Page1 和Page2 的图层(双层模式) 000：灰阶显示(Gray Mode)，此模式下每一个点的灰度决定于DDRAM Page1 与Page2 相对映的值。 Page1 Page2 灰度 --- 0 0 Level1 1 0 Level2 0 1 Level3 1 1 Level4 110：扩展模式(1)，此功能已屏蔽 111：扩展模式(2)，此功能已屏蔽		1h	R/W
3-2	在双层模式下图层逻辑关系 00：Page1 RAM “OR” Page2 RAM 01：Page1 RAM “XOR” Page2 RAM 10：Page1 RAM “NOR” Page2 RAM 11：Page1 RAM “AND” Page2 RAM		0h	R/W
1-0	设定Read/ Write 要在哪一个图层运行 00：存取Page0 (512B SRAM)的Display data RAM 01：存取Page1 (4.8KB SRAM)的Display Data RAM 10：存取Page2 (4.8KB SRAM)的Display Data RAM 11：同时存取Page1 和Page2 的Display Data RAM		1h	R/W

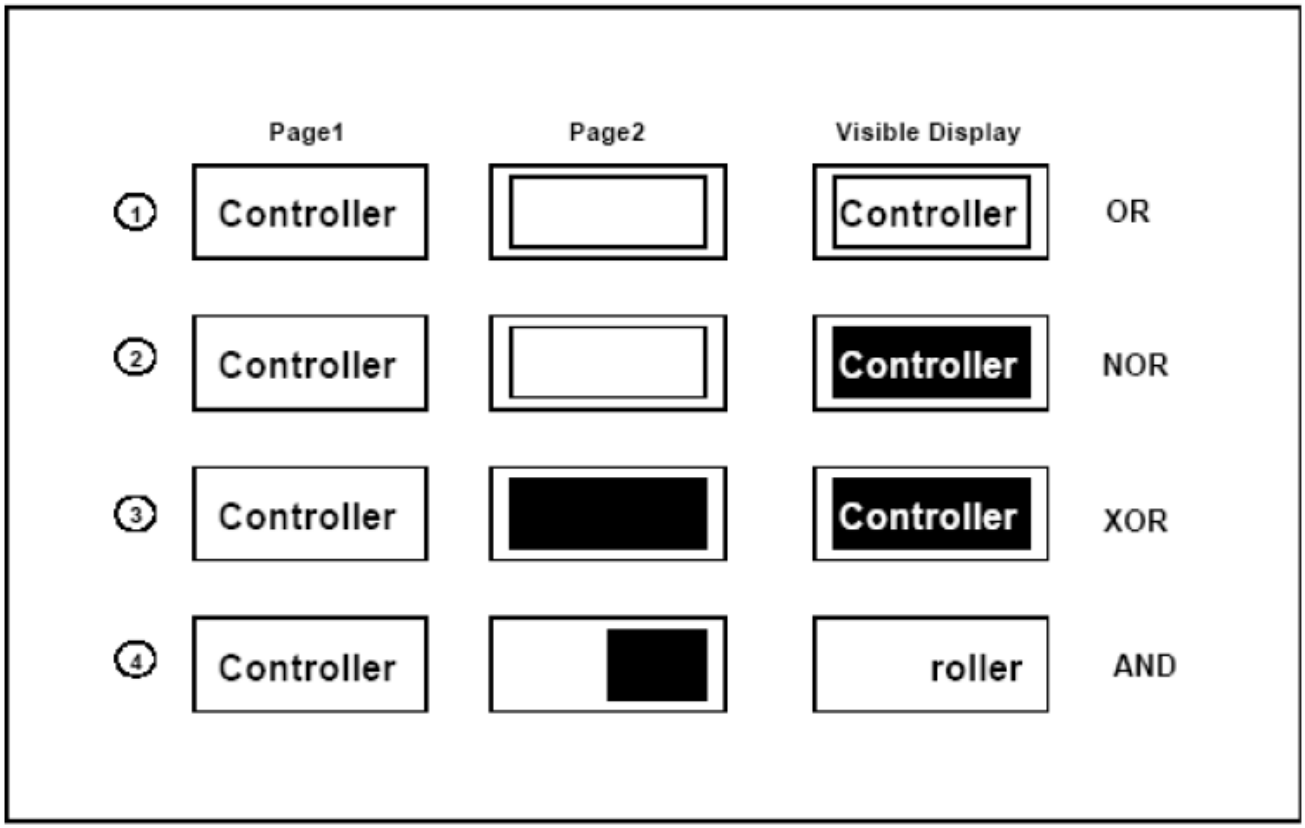


图7.19：图层显示效果

19、屏幕水平卷动及垂直卷动设定：

在屏幕所显示的画面可以作水平卷动，须由缓存器[03h]来做设定。该项功能可达到左右的水平卷动，每次移动的刻度为1 个Byte。另外，还可透过缓存器[71h, 72h]来设定屏幕的区块水平卷动，如图7.20。

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

Bit	Description	Default	Access
2	设定 **Common** 的自动卷动 1：致能 0：禁能	0h	R/W
1	设定 **Segment** 的自动平移 1：致能 0：禁能	0h	R/W
0	设定选择 **Common** 的卷动或是 **Segment** 的平移模式 1：Segment 的平移 0：Common 的卷动	0h	R/W

REG [71h] Shift action range, Begin Common Register (BGCM)

Bit	Description	Default	Access
7-0	在水平移动模式下，设定区块移动的启始 **Common** 位置	0h	R/W

REG [72h] Shift action range END Common Register (EDCM)

Bit	Description	Default	Access
7-0	在水平移动模式下，设定区块移动的结束 **Common** 位置	EFh	R/W

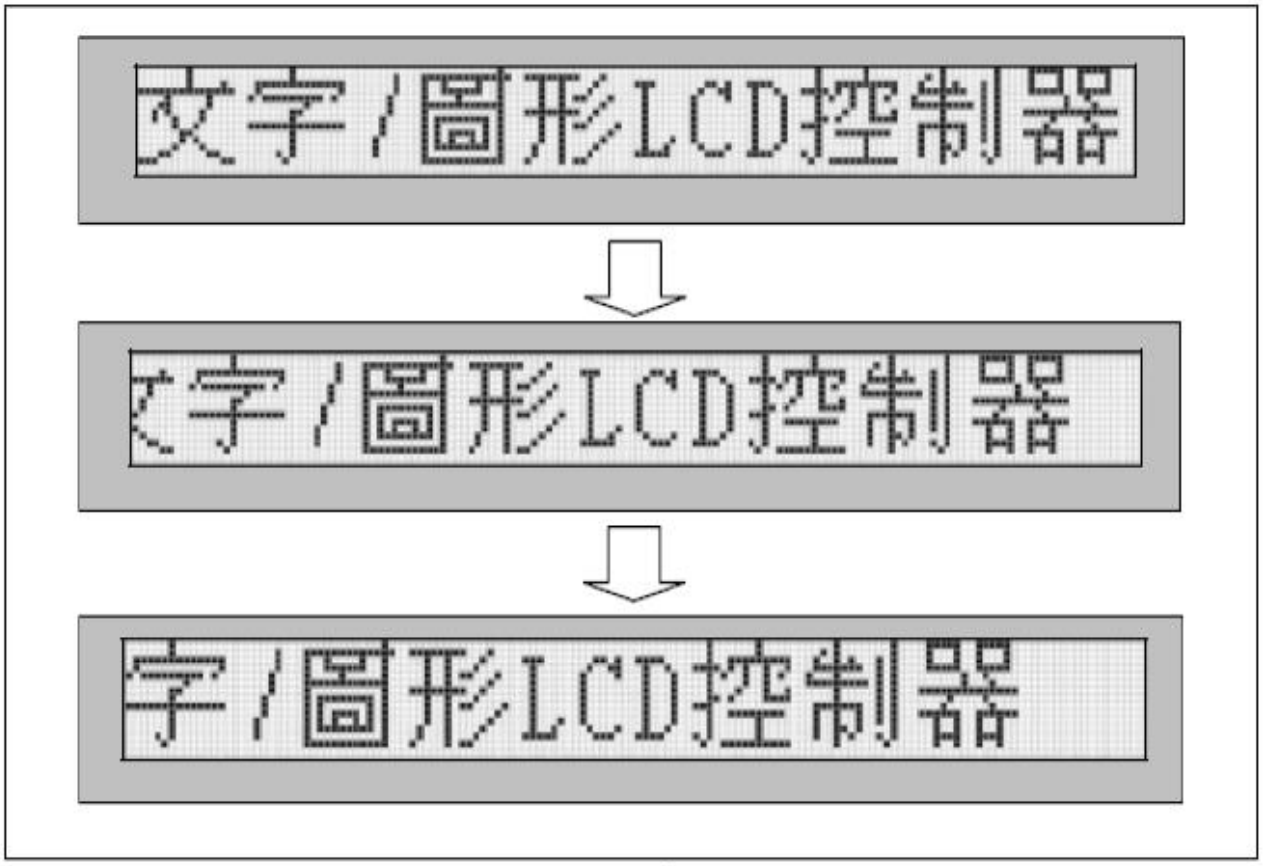


图7.20: 水平卷动的效果

在屏幕所显示的画面可以作垂直卷动，须由缓存器[03h]来做设定。该项功能可达到上下的垂直卷动，每次移动的刻度为1 个像素(Pixel)。如图7.21所示，可作卷动的效果。

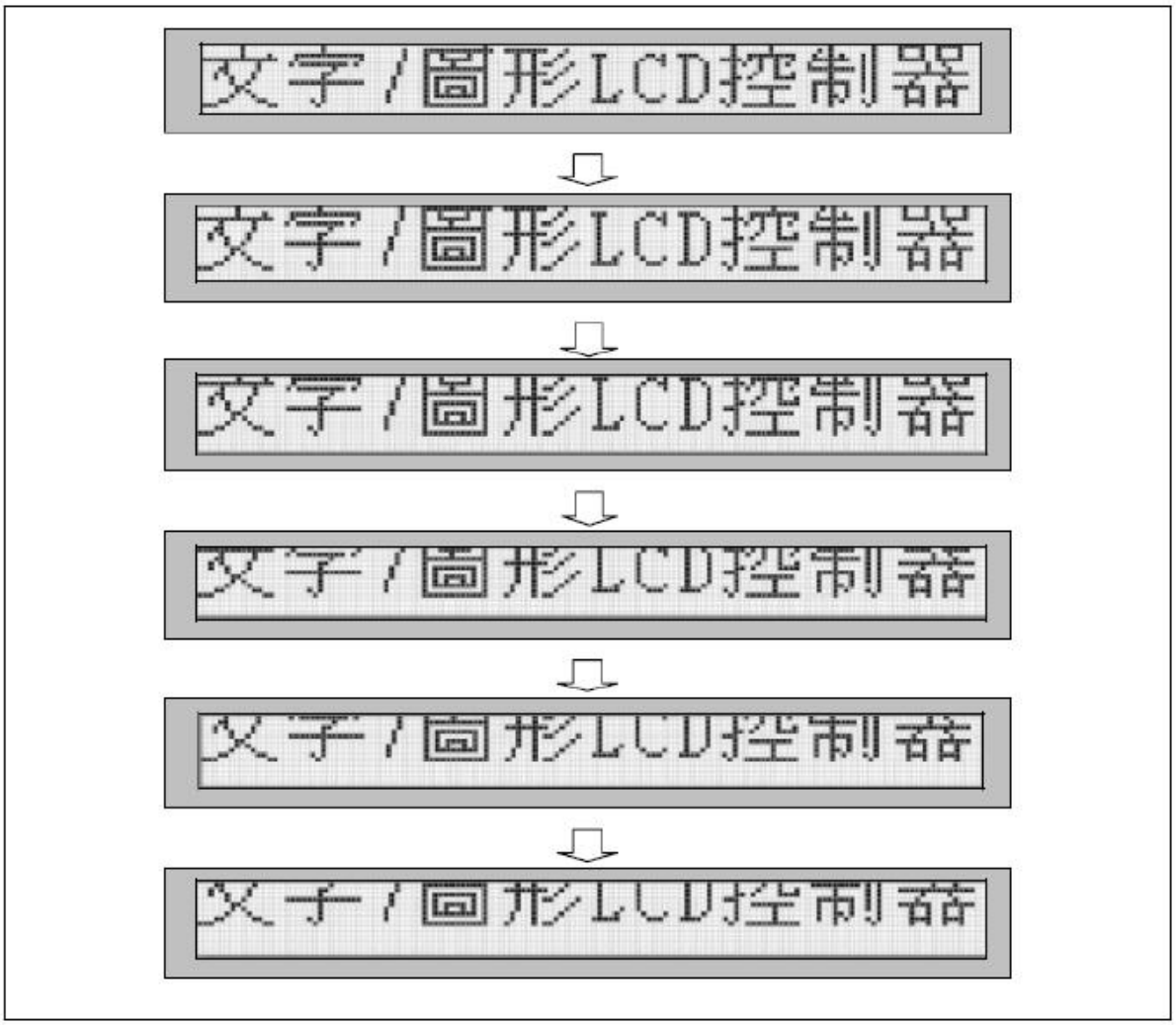


图7.21: 垂直卷动的效果

20、ASCII 区块选择设定:

此LCM内建四个ASCII 区块，包含许多文字、特殊符号或图形等，可供使用者直接取用，此功能可以由缓存器[F0h]的bit[1..0]来设定。如果使用者需要特殊符号或图形，亦可经由调整ROM Code 的方式来建立。下面我们将介绍这四个区块的Pattern(如图7.22～7.25)、选择方式及使用范例。

REG [F0h] Font Control Register (FNCR)

Bit	Description	Text/Graph	Default	Access
1-0	**4** 种 **ASCII** 区块选择 0 0：ASCII 选择区块 0，Latin_1 0 1：ASCII 选择区块 1，Latin_2 1 0：ASCII 选择区块 2，Latin_3 1 1：ASCII 选择区块 3，Latin_4	--	2h	R/W

1) ASCII 字形区块 0:

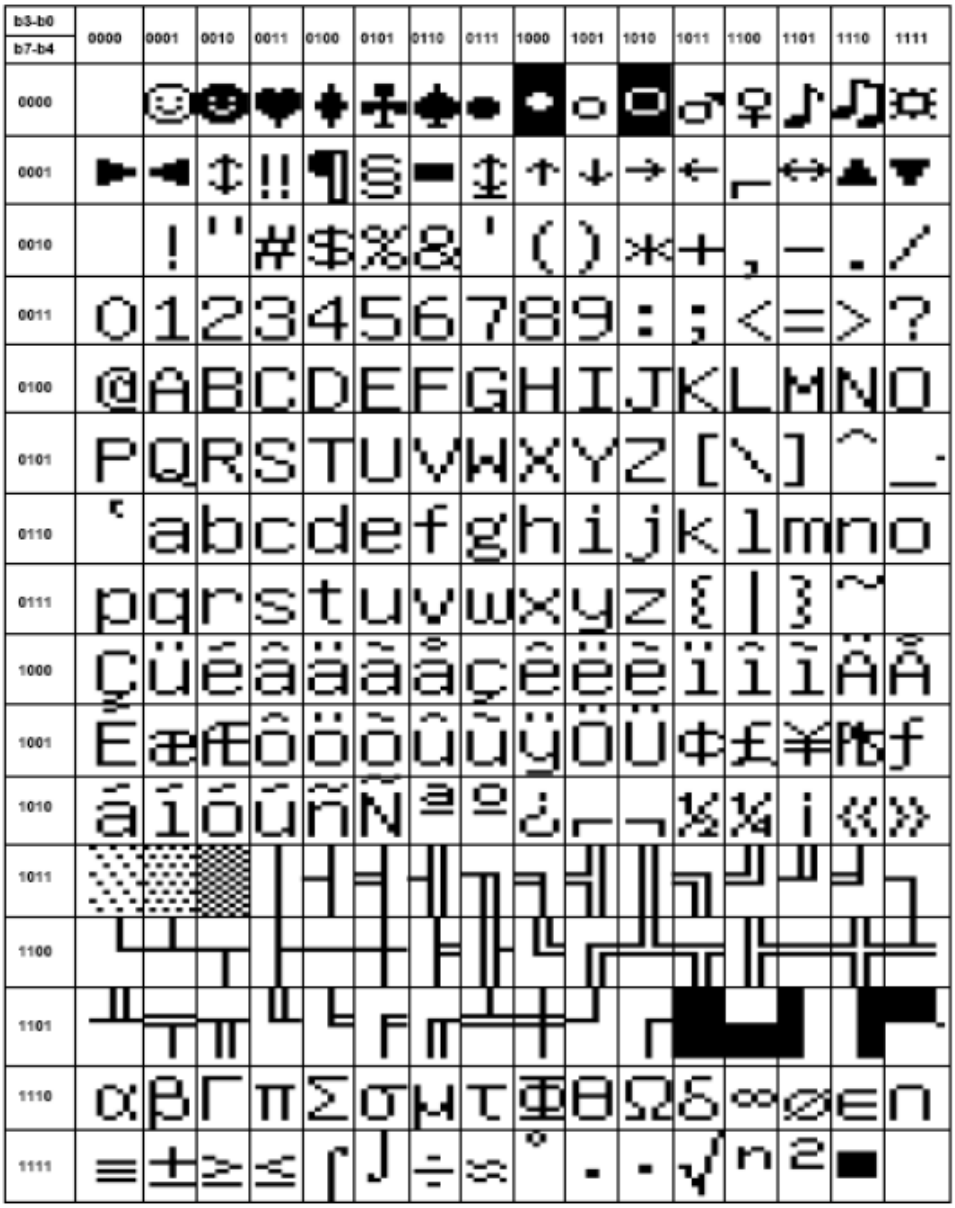


图7.22: 内建ASCII 区块Bank0

2) ASCII 字形区块 1:

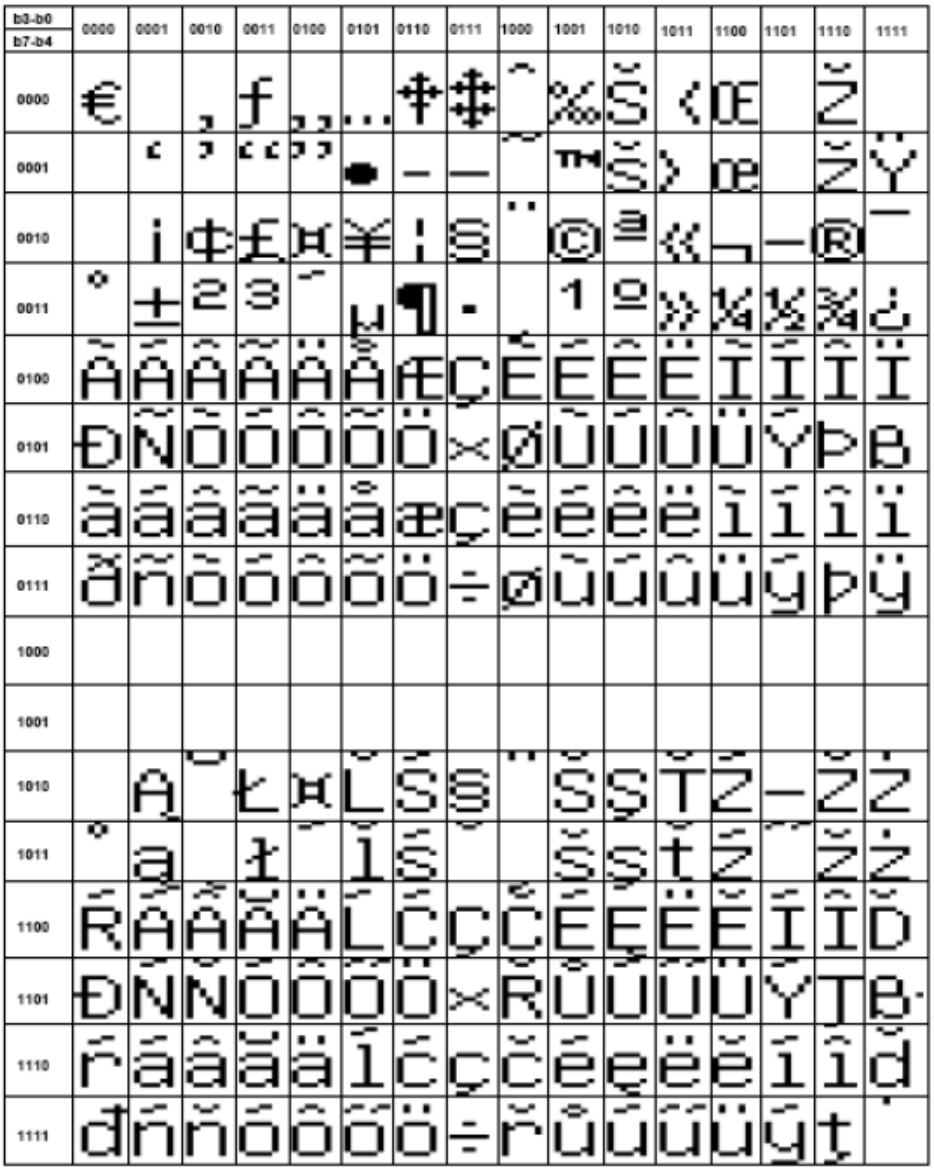


图7.23: 内建ASCII 区块Bank1

3) ASCII 字形区块 2:

区块2 的选择方式与上面相同，只要设定缓存器[F0h]的bit[1..0]，再将选择的Pattern 写入光标所在的位置既可。

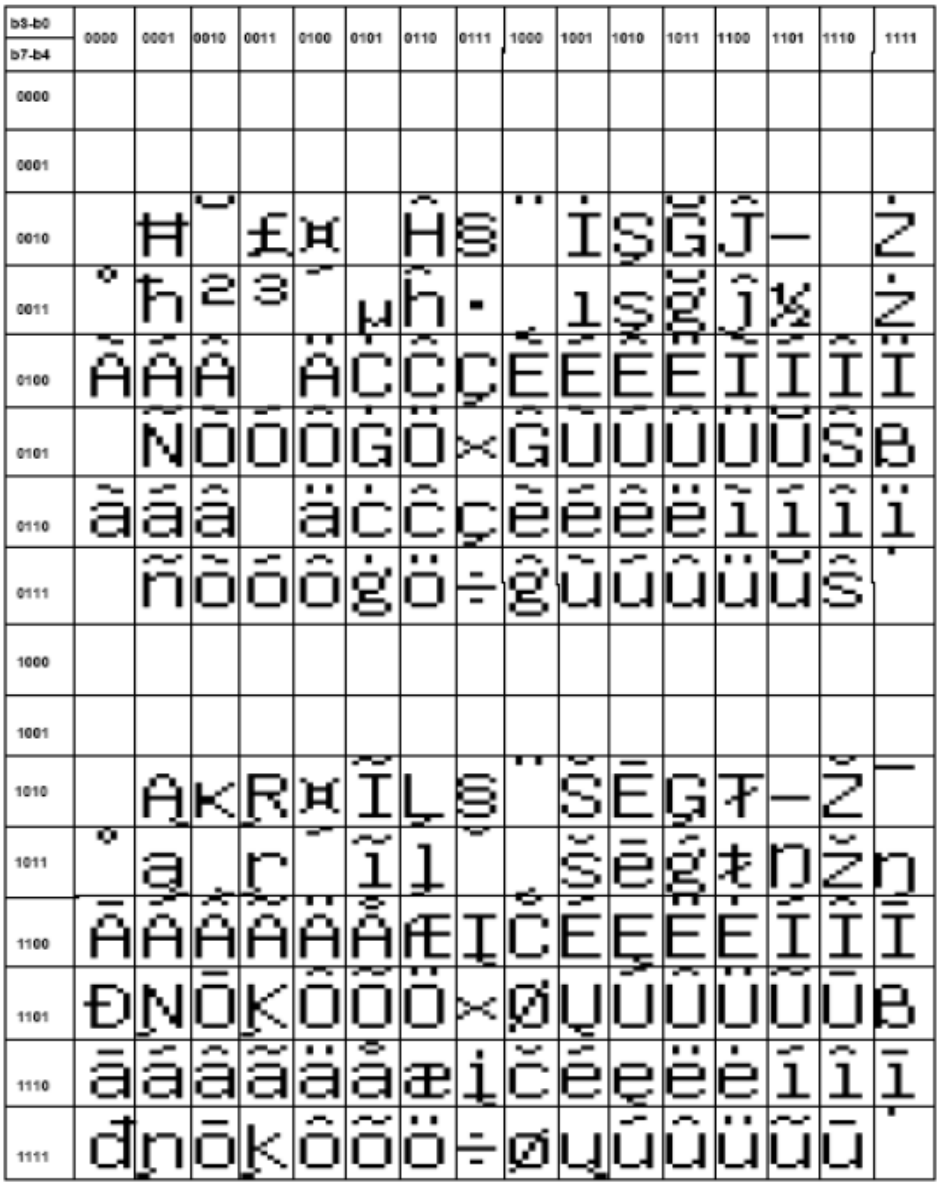


图7.24: 内建ASCII 区块Bank2

4) ASCII 字形区块 3:

区块3 的选择方式与上面相同，也只要设定缓存器[F0h]的bit[1..0]，再将选择的Pattern 写入光标所在的位置既可。在区块3 有许多空的Pattern，如果使用者需要少量的特殊符号或图形，可经由调整ROM Code 的方式填入Pattern 在此区块。

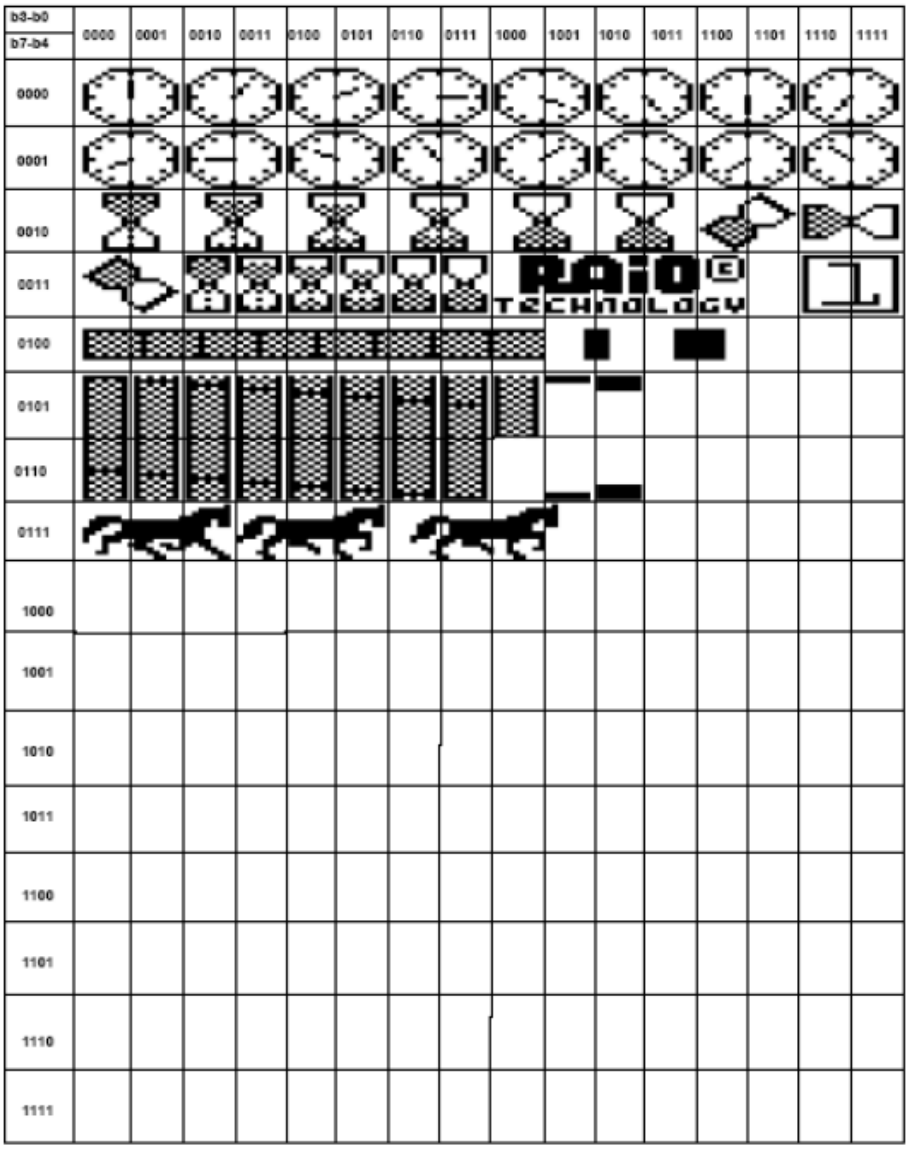


图7.25: 内建ASCII 区块Bank3

21、自行造字:

ZX240128GB 内建512Byte SRAM 可支持自行造字功能，最大字数为16 个全角中文字(16 × 16)。若用到特殊字，是字库内没有的字型，可利用该项功能，增加内建字库的内容，来提升MPU 的存取效率。下面是造字会用到的缓存器及范例:

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

Bit	Description	Default	Access
1-0	设定 **Read/ Write** 要在哪一个图层运行 0 0：存取 Page0 (512B SRAM)的 Display Data RAM 0 1：存取 Page1 (9.6KB SRAM)的 Display Data RAM 1 0：存取 Page2 (9.6KB SRAM)的 Display Data RAM 1 1：同时存取 Page1 和 Page2 的 Display Data RAM	1h	R/W

REG [60h] Cursor Position X Register (CPXR)

Bit	Description	Default	Access
5-0	设定光标 **Segment** 地址	0h	R/W

例 题:

```
Create_Font_Tab0: db 08h,1ch,1ch,ffh,7fh,1ch,3eh,3eh,
                     77h,41h,00h,00h,83h,7fh,3fh,0fh,  0Dh
Create_Font_Tab1: db 20h,10h,1ch,9eh,1eh,1fh,1fh,1fh,
                     1fh,3fh,7eh,feh,fch,f8h,f0h,c0h,  0Dh
Create_Font_Tab2: db FFh,  F0h,  0Dh
Test_Create_Font:
CALL Graphic_Mode          ; 设定成绘图模式
MOV A, #10h                ; Write to Page0→512Byte SRAM
CALL Write_R12
MOV A, #0h                 ; 对中文码 "FFF0" 进行造字
CALL Write_R60             ; 设定光标Segment 地址
Printf Create_Font_Tab0    ; 前16Byte
MOV A, #01h
CALL Write_R60             ; 设定光标Segment 地址(每16Byte 要加1)
Printf Create_Font_Tab1    ; 后 16Byte
CALL Text_Mode             ; 设定成文字模式
MOV A, #91h ; Page1
CALL Write_R12
Printf Create_Font_Tab2    ; 显示码为 "FFF0"的字样→图7.26
```

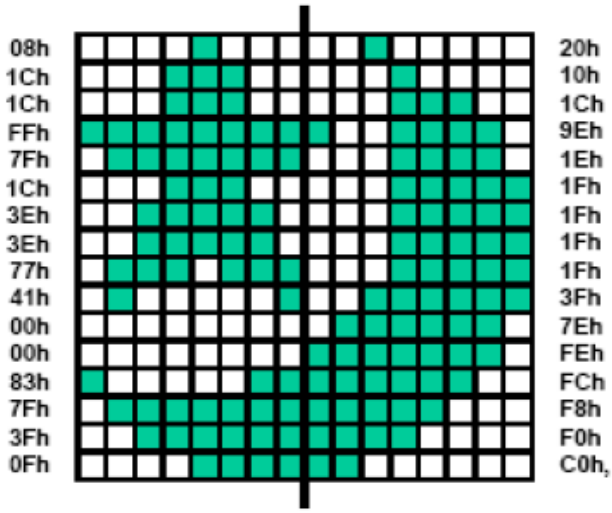


图7.26: 造字

每个全角16×16 中文字占32Byte ， 因此内建512Byte SRAM 可造16 个字，中文码内订为"FFF0～FFFF" 。上例为自建中文码为"FFF0"的字样，若是"FFF1"则写入Data 到Page0 之前的前16Byte 要先将缓存器[60h]设成"02h"，写入Data 到Page0 之前的后16Byte 要将缓存器[60h]设成"03h"，依此类推。

注:在可造字时须要先将Line Distance 设为0，也就是缓存器[11h]的Bit[3:0]设成0，造完字后就无此限定。

22、灰阶显示:

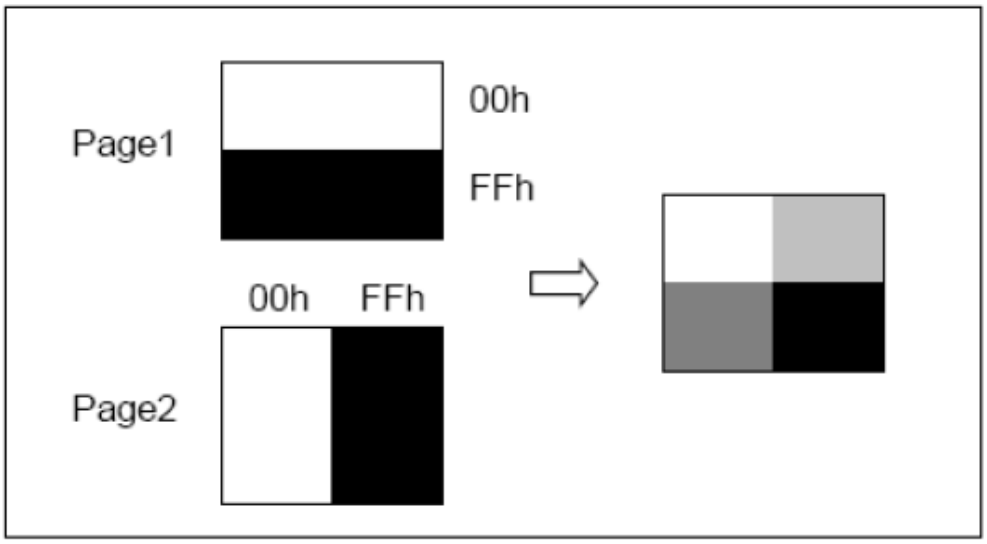
ZX240128GB 可利用分时显示的原理达到灰阶显示的效果，灰阶模式需要同时使用Page1 和Page2 的图层，在此模式下LCD 每一个点的灰阶效果决定于Display RAM Page1 与Page2 的值。对LCD 的同一点来说，[Page1, Page2] 可以为[0,0]、[1,0]、[0,1]、或 [1,1]，如果它们的显示不同将会产生不同的灰度效果，由于是利用分时显示的原理，为了达到良好的显示质量及避免闪烁必须将Frame Rate 或系统频率提高。

REG [E0h] Pattern Data Register (PNTR)

Bit	Description	Default	Access
7-0	**(1) Data Written to DDRAM** **(2) Display Times of Gray Mode** 在灰阶模式下(Register MAMR bit[6..4] = 000) ，此缓存器用来控制显示时间，如果 Frame Rate 固定，此缓存器 "1" 和 "0" 的数目代表显示 1 和 0 的比率。	0h	R/W

PNTR = 55h, AAh, 0Fh, F0h, CCh, 33h 或99h 皆表示缓存器Data 中“1”和“0”的数目一样，那么灰阶Level2 与Level3 的显示效果是一样的，如果设成这些值只能有3 阶的显示效果，必须让 ”1”的数目多于“0”的数目才能有4 灰阶显示的效果。

图7.27 是在屏幕上秀出四灰阶的基本概念，如果Display RAM 的Page1上半全部填”00”，下半部全部填”FF”，且Page2 的左半部全部填”00”，右半部全部填”FF”那么启动灰阶功能后可以在屏幕上秀出四个不同灰阶的方块。



7.27: 灰阶

八、附件:

1、指令时间:

此附件说明 ZX240128GB 在做读/写或是各种模式下写到内存所需的时间。可依

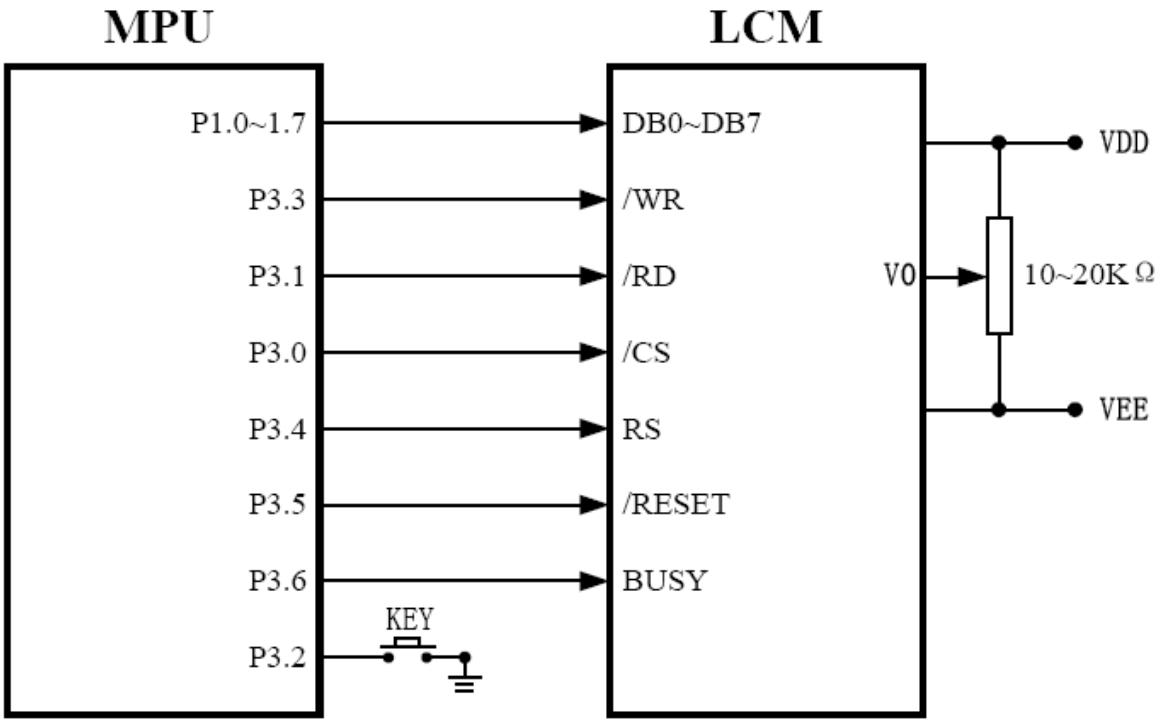
使用者所设定的不同系统频率(SYS_CLK)，来决定各个指令所需要的时间。例如，SYS_CLK=8MHz，每个Clock 的时间=1/SYS_CLK=125ns，而写入缓存器所需的Clock 为3 个机械周期，所以对缓存器做读取或写入时所需的时间约为125ns × 3 lock= 375ns，用以此方式来计算指令所需的时间。

下列是说明各个指令动作所需的机械周期时间:

- 写入缓存器的时间为3 个机械周期
- 读取缓存器的时间为3 个机械周期
- 写入内存的时间为3 个机械周期
- 在绘图模式下写入内存的时间为3 个机械周期
- 在中文字型下写入一个字到内存的时间为35 个机械周期
- 在ASCII 字型下写入一个字到内存的时间为19 个机械周期
- 硬件清除屏幕所需的机械周期时间，公式: 3 + (Com × Seg) / 8

2、程序举例(汇编语言程序):

1) 硬件连接图:



2) 软件举例:

```
;单片机 89C52 与240128GB的连接
;*****************************************************
;IO口定义
    _WR BIT P3.3
    _RD BIT P3.1
    CSL BIT P3.0
    RS  BIT P3.4
    RST BIT P3.5
    BUSY    BIT     P3.6
    DBUS    EQU P1
    KEY BIT P3.2

    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
;=========================================================================
;程序代码开始
    ORG 0000H
    AJMP    MAIN
    ORG 0040H
;=========================================================================
MAIN:
    LCALL   DELAY2
    MOV SP,#60H
    CLR KEY ;如果执行此语句,将全速运行,否则要按一次按键才能显示下一个画面
    CLR RST
    LCALL   DELAY2
    LCALL   DELAY2
    SETB    RST
TEST_PRO:
    LCALL   LCM_INIT    ;初始化
;---------------清除显示数据:文本和图形
    LCALL   CLR_P1
    LCALL   CLR_P2
;---------------全屏显示:横条
    LCALL   DELAY2
    MOV LCM_DATA1,#0FFH
    MOV LCM_DATA2,#00H
    LCALL   TEST_LCD
    LCALL   DELAY1
    JB  KEY,$    ;按键，按一下换一个画面
    MOV LCM_DATA1,#00H
    MOV LCM_DATA2,#0FFH
    LCALL   TEST_LCD
    LCALL   DELAY1
    JB  KEY,$    ;按键，按一下换一个画面
;---------------全屏显示:竖条
    MOV LCM_DATA1,#0AAH
    MOV LCM_DATA2,#0AAH
    LCALL   TEST_LCD
    LCALL   DELAY1
    JB  KEY,$    ;按键，按一下换一个画面
    MOV LCM_DATA1,#055H
    MOV LCM_DATA2,#055H
    LCALL   TEST_LCD
    LCALL   DELAY1
    JB  KEY,$    ;按键，按一下换一个画面
```

```
;---------------全屏显示:交叉点
    MOV LCM_DATA1,#0AAH
    MOV LCM_DATA2,#055H
    LCALL   TEST_LCD
    LCALL   DELAY1
    JB  KEY,$    ;按键，按一下换一个画面
    MOV LCM_DATA1,#055H
    MOV LCM_DATA2,#0AAH
    LCALL   TEST_LCD
    LCALL   DELAY1
    JB  KEY,$    ;按键，按一下换一个画面
;---------------全屏显示:黑点
    MOV LCM_DATA1,#0FFH       ;全显
    MOV LCM_DATA2,#0FFH
    LCALL   TEST_LCD
    LCALL   DELAY1
    JB  KEY,$    ;按键，按一下换一个画面
;---------------文本显示
    LCALL   CLR_P2      ;清屏图形区数据
    MOV DPTR,#TABOFTEXT
    LCALL   HZ_DIS ;显示汉字表中的汉字
    LCALL   DELAY1
    JB  KEY,$  ;按键，按一下换一个画面
    LCALL   DELAY1
;---------------图片显示
    LCALL   CLR_P1     ;清除文本
    MOV DPTR,#BMP000
    LCALL   BMP_DIS         ;画 BMP 图
    LCALL   DELAY1
    JB  KEY,$    ;按键，按一下换一个画面
    NOP
        LCALL CLR_P2      ;清屏画 BMP 图
    LJMP    TEST_PRO
;*****************************************************
LCM_INIT:
    LCALL   DELAY2
;   MOV REGNAME,#00H
;   MOV REGDATA,#0F0H
;   LCALL   WRITE_COM  ;软件复位
    LCALL   DELAY1

    MOV DPTR,#INITTAB
IIINIT:    CLR A
```

```
    MOVC    A,@A+DPTR
    MOV REGNAME,A
    CLR C
    SUBB    A,#0FFH
    JZ  OUTINT
    CLR A
    INC DPTR
    MOVC    A,@A+DPTR
    MOV REGDATA,A
    LCALL   WRITE_COM
    INC DPTR
    AJMP    IIINIT
OUTINT:

    RET
;=============================================
HZ_DIS:
    MOV REGNAME,#00H
    MOV REGDATA,#0CDH ;字符模式
    LCALL   WRITE_COM
    MOV COUNT1,#8  ;字符表中共8行汉字
    MOV LCM_X,#00H ;设显示行列地址
    MOV LCM_Y,#00H ;

HZ_DP0:    LCALL   LOCAL_XY   ;
HZ_DIS1:MOV    COUNT2,#30 ;一行字节计数(含标点15个汉字,合计30个字节)
HZ_DIS11:
    CLR A
    MOVC    A,@A+DPTR
    LCALL   WRITE_DAT
    INC DPTR
    DJNZ    COUNT2,HZ_DIS11
    MOV A,LCM_Y
    ADD A,#16       ;指向下一行地址
    MOV LCM_Y,A
    DJNZ    COUNT1,HZ_DP0
    RET
;=====================================================
BMP_DIS:
    MOV REGNAME,#00H
    MOV REGDATA,#0C5H ;图形模式
    LCALL   WRITE_COM
    MOV COUNT1,#128    ;垂直 =128点
```

```
    MOV LCM_X,#00H      ;画 BMP 图
    MOV LCM_Y,#00H
BMP_D1:     LCALL   LOCAL_XY        ;设显示定行列地址
    MOV COUNT2,#30      ;水平 30X8=240点
BMP_D2:     CLR A
    MOVC    A,@A+DPTR
    LCALL   WRITE_DAT
    INC DPTR
    DJNZ    COUNT2,BMP_D2
    INC LCM_Y       ;指向下一行地址
    DJNZ    COUNT1,BMP_D1
    RET
;=========================================================
CLR_P1:         ;清除文本层显示数据
    MOV REGNAME,#00H
    MOV REGDATA,#0CDH ;文本模式
    LCALL   WRITE_COM
    MOV LCM_X,#00H
    MOV LCM_Y,#00H
    LCALL   LOCAL_XY    ;设行列地址
    MOV COUNT1,#16 ;垂直16行文本
CLR_P11:
    MOV COUNT2,#40 ;水平40个字节宽
CLR_P12:
    MOV A,#020H
    LCALL   WRITE_DAT
    DJNZ    COUNT2,CLR_P12
    DJNZ    COUNT1,CLR_P11
    RET
;=========================================================
CLR_P2:         ;清除图形层显示数据
    MOV REGNAME,#00H
    MOV REGDATA,#0C5H ;图形模式
    LCALL   WRITE_COM
    MOV LCM_X,#00H
    MOV LCM_Y,#00H
    LCALL   LOCAL_XY    ;设行列地址
    MOV COUNT1,#240     ;垂直 =240行
CLR_P21:
    MOV COUNT2,#40 ;水平40X8=320点
CLR_P22:
    CLR A
    MOV A,#00H
```

```
    LCALL   WRITE_DAT
    DJNZ    COUNT2,CLR_P22
    DJNZ    COUNT1,CLR_P21
    RET
;=========================================================
TEST_LCD:  ;图形模式下测试LCD屏
    MOV REGNAME,#00H
    MOV REGDATA,#0C5H
    LCALL   WRITE_COM  ;图形模式
    MOV COUNT1,#64
    MOV LCM_Y,#00H
TEST_1:     MOV LCM_X,#00H
    LCALL   LOCAL_XY
    MOV COUNT2,#30
TEST_2:     MOV A,LCM_DATA1
    LCALL   WRITE_DAT
    DJNZ    COUNT2,TEST_2

    INC LCM_Y
    LCALL   LOCAL_XY
    MOV COUNT2,#30
TEST_3:     MOV A,LCM_DATA2
    LCALL   WRITE_DAT
    DJNZ    COUNT2,TEST_3
    INC LCM_Y
    DJNZ    COUNT1,TEST_1
    RET
;=============================================================
LOCAL_XY:
    MOV REGNAME,#60H   ;设定显示列地址
    MOV REGDATA,LCM_X
    LCALL   WRITE_COM
    MOV REGNAME,#70H   ;设定显示行地址
    MOV REGDATA,LCM_Y
    LCALL   WRITE_COM
    RET
;=============================================================
WRITE_COM:
    MOV A,REGNAME
    LCALL   WRITE_REG
    MOV A,REGDATA
    LCALL   WRITE_REG
    RET
```

```
WRITE_REG:
    MOV DBUS,A
    CLR CSL
    SETB    _RD
    CLR RS
    CLR _WR
    NOP
    NOP
    NOP
    NOP
    SETB    _WR
    SETB    RS
    SETB    CSL
    RET

WRITE_DAT:
    MOV DBUS,A
    CLR CSL
    SETB    _RD
    SETB    RS
    CLR _WR
    NOP
    NOP
    NOP
    NOP
    SETB    _WR
    SETB    RS
    SETB    CSL
    RET

READ_RS:
    MOV A,REGNAME       ;读缓存器
    LCALL   WRITE_REG
    MOV DBUS,#0FFH
    CLR CSL
    SETB    _WR
    CLR RS
    NOP
    MOV A,DBUS
    SETB    RS
    SETB    CSL
    MOV REGDATA1,A
```

```
    RET
;=============================================================
DELAY1:    MOV R5,#09H
DEL11:
    LCALL  DELAY2
    DJNZ   R5,DEL11
    RET
;=============================================================
DELAY2:    MOV R6,#0
DEL21: MOV R7,#0
DEL22: DJNZ    R7,DEL22
    DJNZ   R6,DEL21
    RET
;=========================================
DELAY: MOV R5,#10
    DJNZ   R5,$
    RET
;=============================================================
INITTAB:   ;初始化参数表:前面的是寄存器地址,后面的是参数.最后以0FFH为标志做结
    DB  000H,0C9H  ;LCD基本显示功能设定 图形模式
    DB  001H,0F0H  ;中断准位(高电平) / 设定系统频率8MHz

    DB  002H,010H  ;LCD内存读写速度与功能设定(读字形ROM)
    DB  003H,080H  ;LCD特殊显示功能
    DB  010H,06BH  ;LCD基本显示功能设定 2
    DB  011H,000H  ;光标高度和行距设定
    DB  012H,091H  ;显示层设定：高位8灰度，9水平单一层，A单二层，B同时一二层
                ;
    DB  020H,01EH  ;实际显示窗口垂直结束地址(LCD屏的分辨率30*8=240)
    DB  030H,07FH  ;实际显示窗口水平结束地址(128)
    DB  040H,000H  ;实际显示窗口垂直起始地址
    DB  050H,000H  ;实际显示窗口水平起始地址
                ;
    DB  021H,01EH  ;工作区窗口垂直结束地址(用户指定的显示窗口)
    DB  031H,07FH  ;工作区窗口水平结束地址
    DB  041H,000H  ;工作区窗口垂直起始地址
    DB  051H,000H  ;工作区窗口水平起始地址
                ;
    DB  060H,000H  ;地址定位
    DB  061H,000H  ;
    DB  070H,000H  ;
    DB  071H,000H  ;
    DB  072H,000H  ;
```

```
                ;
    DB  080H,0AAH  ;光标闪烁时间
    DB  081H,004H  ;保留
                ;
    DB  091H,000H  ;保留
    DB  090H,00FH  ;
                ;
    DB  0A0H,011H  ;键、触摸屏、光标行列状态
    DB  0A1H,000H  ;
    DB  0A2H,000H  ;
    DB  0A3H,000H  ;
                   ;
    DB  0B0H,027H  ;
    DB  0B1H,0EFH  ;
                   ;
    DB  0C0H,000H  ;
    DB  0C1H,00AH  ;
    DB  0C8H,080H  ;
    DB  0C9H,080H  ;
    DB  0CAH,000H  ;
                ;
    DB  0D0H,080H  ;
    DB  0E0H,000H  ;自动写此寄存器的数据到DDRAM(图形模式有效)
    DB  0F0H,0A0H  ;字符字体控制
    DB  0F1H,00FH  ;字符点阵大小控制
    DB  0FFH       ;结束标志
TABOFTEXT:
    DB "内置中文字库液晶模块，并行接口"
    DB "分辩率有240*128,320*240 等两款"
    DB "显示8行中文/每行15个标准文字或"
    DB "15行每行20个标准文字....."
    DB "五四运动是中国近代史上划时代的"
    DB "里程碑，它以辛亥革命所不曾有的"
    DB "姿态，展开了彻底地反对帝国主义"
    DB "和封建主义的斗争，标志着中国新"
    DB "民主主义革命的开端，为中国共产"
    DB "党的建立作了思想上干部上的准备"



BMP000:
;--  调入了一幅图像：D:\program\bmp\240128GB.bmp  --
;--  宽度x高度=240x128  --
```

```
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,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB  0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0C0H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,003H,0C0H,07FH,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,0FEH,003H,0C1H,080H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,001H,083H,0C2H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,043H,0C4H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,023H,0C8H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,013H,0C8H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,013H,0D0H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,0BH,0D0H,000H,000H,000H
DB  000H,000H,000H,000H,000H,000H,0C0H,000H,007H,000H,000H,000H,000H,000H,000H,000H
DB  070H,000H,000H,000H,000H,000H,000H,000H,000H,0BH,0D0H,000H,000H,000H,000H,030H
DB  000H,0E0H,000H,000H,0C0H,001H,087H,000H,003H,080H,000H,001H,0C0H,000H,070H,000H
DB  000H,000H,000H,018H,000H,000H,0CH,0BH,0D0H,060H,07EH,001H,0C0H,030H,000H,070H
DB  000H,000H,0CCH,001H,086H,000H,003H,080H,000H,01FH,0E0H,000H,060H,000H,01BH,0F8H
DB  000H,0CH,000H,007H,0FFH,0BH,0D0H,077H,0CFH,000H,0C0H,030H,000H,003H,080H,000H
DB  0CCH,001H,086H,000H,003H,080H,000H,031H,080H,000H,067H,0E0H,0FFH,0B8H,000H,006H
DB  000H,000H,006H,0BH,0D0H,0DH,098H,000H,0CDH,0B0H,001H,0FFH,0C0H,07EH,0D8H,007H
DB  083H,0C0H,003H,000H,000H,003H,000H,0FH,0FCH,000H,0DBH,038H,000H,0C7H,000H,000H
DB  006H,0BH,0D0H,0DH,0B8H,000H,0CDH,0B0H,03FH,060H,000H,0E6H,0F8H,007H,09EH,000H
DB  003H,0FCH,000H,0EH,000H,018H,0C0H,000H,0F3H,0F0H,000H,0E3H,000H,003H,0E6H,0BH
DB  0D1H,0C3H,018H,000H,0CDH,0B0H,000H,060H,000H,067H,080H,0FH,0E6H,078H,03FH,01CH
DB  000H,006H,018H,001H,08CH,000H,0F3H,030H,000H,0E1H,080H,006H,006H,0BH,0D0H,0C7H
DB  0C0H,003H,0EDH,0B0H,000H,0FEH,000H,060H,038H,0FH,08FH,0C0H,033H,018H,000H,007H
DB  0FEH,001H,0FEH,000H,0DBH,0F0H,001H,0C1H,0C0H,000H,006H,0BH,0D0H,000H,0FCH,000H
DB  0CDH,0B0H,007H,0E6H,000H,0F1H,0E0H,019H,0B9H,080H,03FH,0D8H,007H,0FEH,000H,003H
DB  0C6H,000H,0FBH,08CH,001H,080H,0F0H,001H,0E6H,0BH,0D0H,03FH,0E0H,000H,0CDH,0B0H
DB  006H,066H,000H,063H,080H,019H,0C1H,0F0H,01BH,018H,000H,007H,000H,007H,0C6H,000H
DB  0DBH,098H,003H,018H,07CH,03FH,0E6H,0BH,0D0H,061H,0C0H,000H,0FDH,0B0H,006H,066H
DB  000H,061H,09EH,003H,0BFH,080H,01BH,038H,000H,003H,000H,0EH,0F6H,000H,0C3H,0F0H
DB  006H,038H,000H,018H,0E6H,0BH,0D0H,063H,0F0H,001H,0CDH,0B0H,006H,07EH,000H,07BH
DB  0F7H,007H,099H,080H,01FH,0F0H,000H,003H,000H,0CH,0C6H,000H,0C3H,030H,0CH,030H
DB  000H,018H,0C6H,0BH,0D0H,0E7H,0F8H,0FH,019H,0B0H,006H,06EH,000H,07FH,066H,01DH
DB  08DH,080H,01BH,003H,000H,003H,000H,038H,0FEH,000H,0C3H,01CH,018H,067H,000H,0FH
```

```
DB  0C6H, 00BH, 0D1H, 0CEH, 0DEH, 006H, 018H, 030H, 000H, 060H, 003H, 0CFH, 066H, 001H, 081H, 080H
DB  003H, 003H, 000H, 003H, 000H, 061H, 0E6H, 000H, 0C3H, 06FH, 080H, 0C3H, 080H, 00CH, 006H, 00BH
DB  0D0H, 0D8H, 0CFH, 080H, 030H, 030H, 000H, 060H, 000H, 00CH, 066H, 001H, 081H, 080H, 001H, 083H
DB  080H, 007H, 000H, 001H, 086H, 000H, 0C3H, 0C0H, 001H, 0FFH, 080H, 000H, 006H, 00BH, 0D0H, 030H
DB  0C0H, 000H, 060H, 030H, 000H, 060H, 000H, 00CH, 00EH, 001H, 081H, 080H, 001H, 0FFH, 080H, 01EH
DB  000H, 001H, 086H, 000H, 0C3H, 080H, 001H, 081H, 080H, 000H, 03EH, 00BH, 0D0H, 000H, 0C0H, 000H
DB  000H, 030H, 000H, 060H, 000H, 000H, 00CH, 001H, 083H, 080H, 000H, 000H, 000H, 00EH, 000H, 001H
DB  08EH, 000H, 003H, 000H, 000H, 000H, 000H, 000H, 00EH, 00BH, 0D0H, 000H, 000H, 000H, 000H, 030H
DB  000H, 000H, 000H, 000H, 000H, 000H, 003H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00EH, 000H
DB  000H, 000H, 000H, 000H, 000H, 000H, 00CH, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB  000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB  000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB  000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB  000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB  000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB  000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH
DB  0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 03EH, 000H, 081H, 0E0H, 000H, 008H
DB  002H, 003H, 0E0H, 078H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H
DB  000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 041H, 001H, 082H, 010H, 020H, 010H, 00EH, 004H
DB  010H, 084H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H
DB  000H, 000H, 000H, 000H, 000H, 000H, 041H, 002H, 082H, 010H, 010H, 020H, 002H, 004H, 011H, 002H
DB  000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H
DB  000H, 000H, 000H, 000H, 041H, 004H, 084H, 008H, 008H, 040H, 002H, 004H, 011H, 002H, 000H, 000H
DB  000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB  000H, 000H, 001H, 004H, 084H, 008H, 004H, 080H, 002H, 000H, 011H, 002H, 000H, 000H, 000H, 000H
DB  000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB  002H, 008H, 084H, 008H, 003H, 000H, 002H, 000H, 020H, 084H, 000H, 000H, 000H, 000H, 000H, 000H
DB  000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 002H, 008H
DB  084H, 008H, 003H, 000H, 002H, 000H, 020H, 078H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB  000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 004H, 010H, 084H, 008H
DB  004H, 080H, 002H, 000H, 040H, 084H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH
DB  0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 008H, 01FH, 0E4H, 008H, 008H, 040H
DB  002H, 000H, 081H, 002H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H
DB  000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 010H, 000H, 084H, 008H, 010H, 020H, 002H, 001H
DB  001H, 002H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H
DB  000H, 000H, 000H, 000H, 000H, 000H, 021H, 000H, 082H, 010H, 020H, 010H, 002H, 002H, 011H, 002H
DB  000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H
DB  000H, 000H, 000H, 000H, 041H, 000H, 082H, 010H, 040H, 008H, 002H, 004H, 010H, 084H, 000H, 000H
DB  000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB  000H, 000H, 07FH, 003H, 0E1H, 0E0H, 000H, 000H, 00FH, 087H, 0F0H, 078H, 000H, 000H, 000H, 000H
DB  000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB  000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
```

```
DB   000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH
DB   0D0H, 000H, 000H, 000H, 000H, 000H, 040H, 000H, 080H, 000H, 080H, 000H, 040H, 010H, 010H, 000H
DB   000H, 000H, 000H, 000H, 000H, 008H, 024H, 008H, 020H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H
DB   000H, 000H, 000H, 000H, 040H, 000H, 040H, 000H, 040H, 000H, 020H, 008H, 008H, 003H, 0FCH, 007H
DB   0FCH, 007H, 0FEH, 008H, 024H, 008H, 020H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H
DB   000H, 008H, 042H, 000H, 040H, 00FH, 0FFH, 00FH, 0FFH, 089H, 0FFH, 082H, 004H, 004H, 004H, 000H
DB   000H, 009H, 0FFH, 008H, 020H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 00FH
DB   0FFH, 03FH, 0FFH, 088H, 001H, 008H, 040H, 002H, 048H, 003H, 0FCH, 007H, 0FCH, 000H, 000H, 008H
DB   024H, 008H, 0FEH, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 008H, 042H, 002H
DB   008H, 010H, 002H, 008H, 040H, 022H, 048H, 002H, 004H, 004H, 004H, 000H, 000H, 03EH, 0FEH, 03EH
DB   022H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 008H, 042H, 002H, 008H, 007H
DB   0F8H, 00FH, 0FFH, 014H, 09FH, 003H, 0FCH, 007H, 0FCH, 01FH, 0FFH, 088H, 082H, 008H, 022H, 000H
DB   000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 008H, 042H, 002H, 008H, 000H, 010H, 008H
DB   090H, 014H, 091H, 002H, 004H, 001H, 010H, 000H, 040H, 01CH, 0FEH, 008H, 022H, 000H, 000H, 000H
DB   000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 008H, 042H, 001H, 010H, 000H, 020H, 009H, 010H, 005H
DB   0AAH, 000H, 000H, 001H, 010H, 000H, 040H, 01AH, 082H, 008H, 022H, 000H, 000H, 000H, 000H, 00BH
DB   0D0H, 000H, 000H, 000H, 000H, 008H, 042H, 001H, 010H, 000H, 040H, 00BH, 0FFH, 00AH, 0E6H, 01FH
DB   09FH, 091H, 012H, 004H, 048H, 028H, 0FEH, 00BH, 0FFH, 080H, 000H, 000H, 000H, 00BH, 0D0H, 000H
DB   000H, 000H, 000H, 00FH, 0FEH, 000H, 0A0H, 01FH, 0FFH, 089H, 010H, 008H, 094H, 010H, 090H, 089H
DB   012H, 004H, 044H, 028H, 010H, 008H, 020H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H
DB   000H, 008H, 042H, 000H, 040H, 000H, 040H, 008H, 010H, 038H, 088H, 01FH, 09FH, 085H, 014H, 008H
DB   042H, 009H, 0FFH, 08EH, 050H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H
DB   040H, 000H, 0A0H, 000H, 040H, 00FH, 0FFH, 088H, 08CH, 010H, 090H, 085H, 018H, 010H, 043H, 008H
DB   010H, 038H, 048H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 040H, 001H
DB   018H, 000H, 040H, 008H, 010H, 008H, 094H, 010H, 090H, 081H, 010H, 020H, 041H, 008H, 028H, 010H
DB   086H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 040H, 006H, 007H, 080H
DB   040H, 010H, 010H, 008H, 0A2H, 01FH, 09FH, 0BFH, 0FFH, 080H, 040H, 008H, 026H, 000H, 083H, 080H
DB   000H, 000H, 000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 040H, 038H, 002H, 001H, 040H, 010H
DB   010H, 008H, 0C3H, 090H, 090H, 080H, 000H, 001H, 040H, 008H, 043H, 081H, 001H, 000H, 000H, 000H
DB   000H, 00BH, 0D0H, 000H, 000H, 000H, 000H, 000H, 040H, 000H, 000H, 000H, 080H, 020H, 010H, 008H
DB   081H, 000H, 000H, 000H, 000H, 000H, 080H, 009H, 081H, 002H, 000H, 000H, 000H, 000H, 000H, 00BH
DB   0D0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D1H, 0FDH, 0F9H, 0C0H, 000H, 000H, 030H, 0FCH
DB   0FCH, 0FCH, 000H, 078H, 0FCH, 030H, 078H, 020H, 078H, 0FCH, 0FCH, 000H, 000H, 000H, 000H, 000H
```

```
DB   000H,000H,000H,000H,000H,00BH,0D1H,024H,084H,080H,000H,000H,048H,088H,080H,080H
DB   000H,084H,088H,048H,084H,0E0H,084H,088H,080H,000H,000H,000H,000H,000H,000H,000H
DB   000H,000H,000H,00BH,0D0H,020H,090H,080H,000H,000H,084H,088H,080H,080H,000H,084H
DB   088H,084H,084H,020H,084H,088H,080H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,00BH,0D0H,020H,090H,080H,000H,000H,084H,010H,080H,080H,000H,084H,010H,084H
DB   008H,020H,084H,010H,080H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH
DB   0D0H,020H,0F0H,080H,000H,000H,084H,010H,0B0H,0B0H,000H,008H,010H,084H,030H,020H
DB   048H,010H,0B0H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,020H
DB   090H,080H,000H,000H,084H,020H,0C8H,0C8H,0FEH,008H,020H,04CH,008H,020H,030H,020H
DB   0C8H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,020H,090H,080H
DB   060H,000H,084H,020H,004H,004H,000H,010H,020H,034H,004H,020H,048H,020H,004H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,020H,080H,080H,060H,000H
DB   084H,020H,004H,004H,000H,020H,020H,004H,004H,020H,084H,020H,004H,000H,000H,000H
DB   000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,020H,084H,080H,000H,000H,084H,020H
DB   084H,084H,000H,040H,020H,004H,084H,020H,084H,020H,084H,000H,000H,000H,000H,000H
DB   000H,000H,000H,000H,000H,00BH,0D0H,020H,084H,084H,060H,000H,048H,020H,088H,088H
DB   000H,084H,020H,048H,088H,020H,084H,020H,088H,000H,000H,000H,000H,000H,000H,000H
DB   000H,000H,000H,00BH,0D0H,071H,0F9H,0FCH,060H,000H,030H,020H,070H,070H,000H,0FCH
DB   020H,070H,070H,0F8H,078H,020H,070H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,00BH,0D0H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH
DB   0D0H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,0FCH,010H,0E7H
DB   000H,000H,018H,07EH,07EH,07EH,000H,03CH,07EH,018H,03CH,010H,03CH,01CH,004H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,042H,010H,042H,000H,000H
DB   024H,044H,040H,040H,000H,042H,044H,024H,042H,070H,042H,024H,00CH,000H,000H,000H
DB   000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,048H,018H,024H,000H,000H,042H,044H
DB   040H,040H,000H,042H,044H,042H,042H,010H,042H,040H,014H,000H,000H,000H,000H,000H
DB   000H,000H,000H,000H,000H,00BH,0D0H,048H,028H,024H,000H,000H,042H,008H,040H,040H
DB   000H,042H,008H,042H,004H,010H,042H,040H,024H,000H,000H,000H,000H,000H,000H,000H
DB   000H,000H,000H,00BH,0D0H,078H,028H,018H,000H,000H,042H,008H,058H,058H,000H,004H
DB   008H,042H,018H,010H,024H,058H,024H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,00BH,0D0H,048H,024H,018H,000H,000H,042H,010H,064H,064H,07FH,004H,010H,026H
DB   004H,010H,018H,064H,044H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH
DB   0D0H,048H,03CH,018H,030H,000H,042H,010H,002H,002H,000H,008H,010H,01AH,002H,010H
DB   024H,042H,044H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,040H
DB   044H,024H,030H,000H,042H,010H,002H,002H,000H,010H,010H,002H,002H,010H,042H,042H
DB   07EH,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,040H,042H,024H
DB   000H,000H,042H,010H,042H,042H,000H,020H,010H,002H,042H,010H,042H,042H,004H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,040H,042H,042H,030H,000H
DB   024H,010H,044H,044H,000H,042H,010H,024H,044H,010H,042H,024H,004H,000H,000H,000H
```

```
DB   000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,0E0H,0E7H,0E7H,030H,000H,018H,010H
DB   038H,038H,000H,07EH,010H,038H,038H,07CH,03CH,018H,01EH,000H,000H,000H,000H,000H
DB   000H,000H,000H,000H,000H,00BH,0D0H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,000H,000H,00BH,0D0H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,00BH,0D0H,0FCH,000H,0EEH,000H,030H,070H,000H,000H,000H,030H,000H,000H,010H
DB   01CH,03CH,038H,03CH,010H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH
DB   0D0H,042H,000H,06CH,000H,030H,010H,000H,000H,000H,030H,000H,000H,070H,024H,042H
DB   044H,042H,070H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,048H
DB   000H,06CH,000H,000H,010H,000H,000H,000H,000H,010H,000H,010H,040H,042H,05AH,042H
DB   010H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,048H,000H,06CH
DB   000H,000H,010H,018H,000H,000H,000H,010H,000H,010H,040H,042H,0AAH,042H,010H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,078H,000H,06CH,03CH,070H
DB   010H,018H,0EEH,0C6H,070H,07CH,03CH,010H,058H,024H,0AAH,004H,010H,000H,01CH,03CH
DB   0FEH,000H,000H,000H,000H,000H,000H,00BH,0D0H,048H,07FH,054H,042H,010H,010H,000H
DB   032H,042H,010H,010H,042H,010H,064H,018H,0AAH,004H,010H,000H,022H,042H,049H,000H
DB   000H,000H,000H,000H,000H,00BH,0D0H,048H,000H,054H,01EH,010H,010H,000H,020H,042H
DB   010H,010H,07EH,010H,042H,024H,0AAH,008H,010H,000H,040H,042H,049H,000H,000H,000H
DB   000H,000H,000H,00BH,0D0H,040H,000H,054H,022H,010H,010H,000H,020H,042H,010H,010H
DB   040H,010H,042H,042H,0B4H,010H,010H,000H,040H,042H,049H,000H,000H,000H,000H,000H
DB   000H,00BH,0D0H,042H,000H,054H,042H,010H,010H,000H,020H,042H,010H,010H,040H,010H
DB   042H,042H,042H,020H,010H,000H,040H,042H,049H,000H,000H,000H,000H,000H,000H,00BH
DB   0D0H,042H,000H,054H,042H,010H,010H,018H,020H,046H,010H,010H,042H,010H,024H,042H
DB   044H,042H,010H,060H,022H,042H,049H,000H,000H,000H,000H,000H,000H,00BH,0D0H,0FCH
DB   000H,0D6H,03FH,07CH,07CH,018H,0F8H,03BH,07CH,00CH,03CH,07CH,018H,03CH,038H,07EH
DB   07CH,060H,01CH,03CH,0EDH,000H,000H,000H,000H,000H,000H,00BH,0D0H,000H,000H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,000H,000H,000H,000H,000H
DB   000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,000H,000H,000H,000H,000H,000H,00BH,0D0H,0E7H,000H,000H,000H,000H,002H,002H
DB   000H,000H,000H,000H,000H,000H,030H,000H,000H,070H,000H,006H,000H,000H,000H,000H
DB   000H,000H,000H,000H,000H,00BH,0D0H,042H,000H,000H,000H,000H,002H,002H,000H,000H
DB   000H,000H,000H,000H,030H,000H,000H,010H,000H,002H,000H,000H,000H,000H,000H,000H
DB   000H,000H,000H,00BH,0D0H,042H,010H,010H,000H,000H,004H,004H,000H,000H,000H,000H
DB   000H,000H,000H,010H,000H,010H,000H,002H,000H,000H,000H,000H,000H,000H,000H,000H
DB   000H,00BH,0D0H,042H,010H,010H,000H,018H,004H,004H,000H,000H,000H,000H,000H,000H
DB   000H,010H,000H,010H,000H,002H,000H,000H,000H,000H,000H,000H,000H,000H,000H,00BH
DB   0D0H,042H,07CH,07CH,0D8H,018H,008H,008H,0D7H,0D7H,0D7H,000H,0EEH,0C6H,070H,07CH
DB   03CH,010H,01CH,01EH,000H,01CH,03CH,0FEH,000H,000H,000H,000H,000H,00BH,0D0H,07EH
DB   010H,010H,064H,000H,008H,008H,092H,092H,092H,000H,032H,042H,010H,010H,042H,010H
DB   022H,022H,000H,022H,042H,049H,000H,000H,000H,000H,000H,00BH,0D0H,042H,010H,010H
DB   042H,000H,010H,010H,092H,092H,092H,000H,020H,042H,010H,010H,07EH,010H,040H,042H
```

```
DB   000H, 040H, 042H, 049H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 042H, 010H, 010H, 042H, 000H
DB   010H, 010H, 0AAH, 0AAH, 0AAH, 000H, 020H, 042H, 010H, 010H, 040H, 010H, 040H, 042H, 000H, 040H
DB   042H, 049H, 000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 042H, 010H, 010H, 042H, 000H, 020H, 020H
DB   0AAH, 0AAH, 0AAH, 000H, 020H, 042H, 010H, 010H, 040H, 010H, 040H, 042H, 000H, 040H, 042H, 049H
DB   000H, 000H, 000H, 000H, 000H, 00BH, 0D0H, 042H, 010H, 010H, 044H, 018H, 020H, 020H, 044H, 044H
DB   044H, 060H, 020H, 046H, 010H, 010H, 042H, 010H, 022H, 026H, 060H, 022H, 042H, 049H, 000H, 000H
DB   000H, 000H, 000H, 00BH, 0D0H, 0E7H, 00CH, 00CH, 078H, 018H, 040H, 040H, 044H, 044H, 044H, 060H
DB   0F8H, 03BH, 07CH, 00CH, 03CH, 07CH, 01CH, 01BH, 060H, 01CH, 03CH, 0EDH, 000H, 000H, 000H, 000H
DB   000H, 00BH, 0C8H, 000H, 000H, 000H, 040H, 000H, 040H, 040H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 013H
DB   0C8H, 000H, 000H, 000H, 0E0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 013H, 0C4H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 023H, 0C2H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 043H, 0C1H, 080H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 001H, 083H, 0C0H, 07FH, 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, 0FEH, 003H, 0C0H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H, 000H
DB   000H, 000H, 000H, 003H, 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, 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
     END
```