

图形点阵液晶显示模块使用手册

CM12832-1SLYA

广州市瑞通科技有限公司

广州新赛格电子城三楼B3313档

TEL : 020 - 88403268

FAX : 020 - 87580480

http : www.rtlcd.com

E-mail : xiezhong2@163.com

一、概述

CM12832-1 是一种图形点阵液晶显示器,它主要由行驱动器/列驱动器及 128×32 全点阵液晶显示器组成。可完成图形显示,也可以显示 8×2 个(16×16 点阵)汉字。

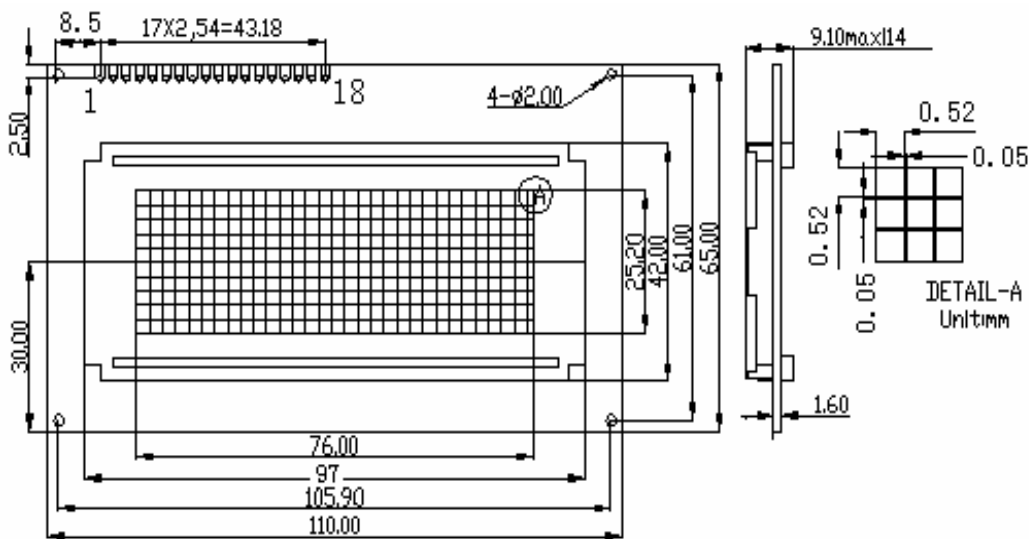
主要技术参数和性能:

1. 电源: VDD: +5V;
2. 显示内容: $128(\text{列}) \times 32(\text{行})$ 点
3. 全屏幕点阵
4. 七种指令
5. 与 CPU 接口采用 8 位数据总线并行输入输出和 8 条控制线
6. 占空比 1/32
7. 工作温度: $-10 \sim +55$, 存储温度: $-20 \sim +70$
8. 显示模式: 黄绿膜、灰膜、蓝膜、黑白膜
9. 背光特性: LED 背光 (黄绿色、蓝色、白色、红色)
10. 模块封装方式: COB
11. 视角方向: $6:00$
12. 功耗:

带格式的: 项目符号和编号

外形尺寸图

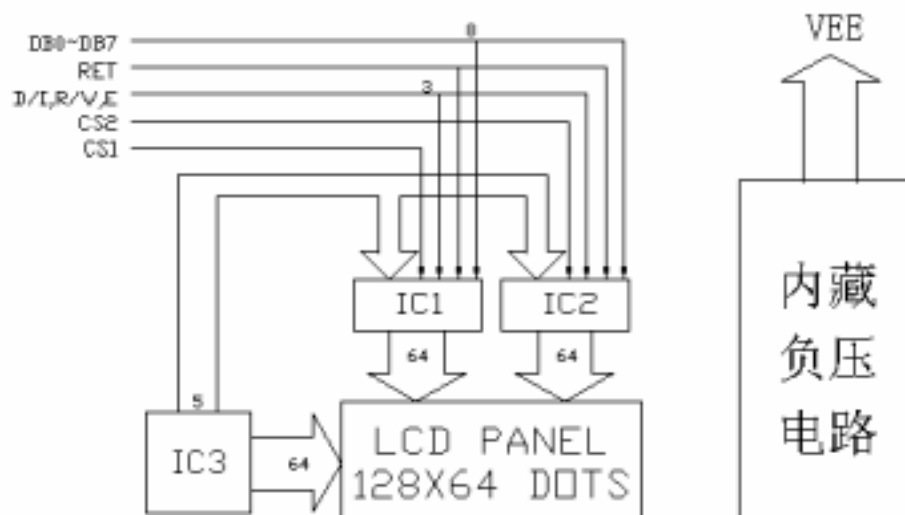
1. 外形尺寸图



2. 外形尺寸

ITEM	NOMINAL DIMEN	UNIT
模块体积	110.0 × 65.0 × 14	mm
视域	76 × 25.2	mm
行列点阵数	128 × 32	dots
点距离	0.05 × 0.05	mm
点大小	0.52 × 0.52	mm

三. 模块主要硬件构成说明 (结构框图)



注：IC2 控制模块的左半屏，IC1 控制模块的右半屏

IC3 为行驱动器。IC1，IC2 为列驱动器。IC1，IC2，IC3 含有以下主要功能器件。了解如下器件有利于对 LCD 模块之编程。

LCD 模块之编程。

1. 指令寄存器 (IR)

IR 是用来寄存指令码，与数据寄存器寄存数据相对应。当 D/I=1 时，在 E 信号下降沿的作用下，指令码写入 IR。

2. 数据寄存器 (DR)

DR 是用来寄存数据的，与指令寄存器寄存指令相对应。当 D/I=1 时，在 E 信号的下降沿作用下，图形显示数据写入 DR，或在 E 信号高电平作用下由 DR 读到 DB7~DB0 数据总线。DR 和 DDRAM 之间的数据传输是模块内部自动执行的。

3. 忙标志:BF

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

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

4. 显示控制触发器 DFF

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

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

5.XY 地址计数器

XY 地址计数器是一个 9 位计数器。高三位是 X 地址计数器，低 6 位为 Y 地址计数器。XY 地址计数器实际上是作为 DDRAM 的地址指针。X 地址计数器为 DDRAM 的页指针，Y 地址计数器为 DDRAM 的 Y 地址指针。

X 地址计数器是没有记数功能的，只能用指令设置。

Y 地址计数器具有循环记数功能，各显示数据写入后，Y 地址自动加 1，Y 地址指针从 0 到 63。

6.显示数据 RAM(DDRAM)

DDRAM 是存储图形显示数据的。数据为 1 表示显示选择，数据为 0 表示显示非选择。DDRAM 与地址和显示位置的关系见 DDRAM 地址表（见第 6 页）。

7.Z 地址计数器

Z 地址计数器是一个 6 位计数器，此计数器具备循环记数功能，它是用于显示行扫描同步。当一行扫描完成，此地址计数器自动加 1，指向下一行扫描数据，RST 复位后 Z 地址计数器为 0。

Z 地址计数器可以用指令 DISPLAY START LINE 预置。因此，显示屏幕的起始行就由此指令控制，即 DDRAM 的数据从哪一行开始显示在屏幕的第一行。此模块的 DDRAM 共 32 行，屏幕可以循环滚动显示 32 行。

四．模块的外部接口

外部接口信号如下表 2 所示：

表 2

管脚号	管脚名称	LEVER	管脚功能描述
1	VLED-	0	LED(+5V)或 EL 背光源
2	VLED+	5.0V	LED(0V)或 EL 背光源
3	V0	5.0V 0V	液晶显示器驱动电压
4	VDD	5.0V	电源电压
5	VSS	0	电源地
6	E	H/L	R/W=“L”，E 信号下降沿锁存 DB7 DB0 R/W=“H”，E=“H” DDRAM 数据读到 DB7 DB0
7	D/I	H/L	D/I=“H”，表示 DB7 DB0 为显示数据 D/I=“L”，表示 DB7 DB0 为显示指令数据
8	R/W	H/L	R/W=“H”，E=“H” 数据被读到 DB7 DB0 R/W=“L”，E=“H→L” 数据被写到 IR 或 DR
9	CS	H/L	L:选择芯片(右半屏)信号 H:选择芯片(左半屏)信号
10	RET	H/L	复位信号,低电平复位
11	DB0	H/L	数据线
12	DB1	H/L	数据线
13	DB2	H/L	数据线
14	DB3	H/L	数据线
15	DB4	H/L	数据线
16	DB5	H/L	数据线
17	DB6	H/L	数据线
18	DB7	H/L	数据线

五、指令说明

表 3

指令	指令码										功能
	RW	DI	D7	D6	D5	D4	D3	D2	D1	D0	
显示 ON/OFF	0	0	0	0	1	1	1	1	1	1/0	控制显示器的开关，不影响 DDRAM 中数据和内部状态
显示起始行	0	0	1	1	显示起始行 (0...63)						指定显示屏从 DDRAM 中哪一行开始显示数据
设置 X 地址	0	0	1	0	1	1	1	X : 0...7			设置 DDRAM 中的页地址 (X 地址)
设置 Y 地址	0	0	0	1	Y 地址(0-63)						设置地址(Y 地址)
读状态	1	0	B U S Y	0	ON/ OFF	RST	0	0	0	0	RST 1:复位 0 : 正常 ON/OFF 1:显示开 0: 显示关 BUSY 0 : READY 1 : IN OPERATION
写显示数据	0	1	显示数据								将数据线上的数据 DB7~DB0 写入 DDRAM
读显示数据	1	1	显示数据								将数据线上的数据 DB7~DB0 写入 DDRAM

1. 显示开关控制(DISPLAY ON/OFF)

代码	R/W	D/I	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
形式	0	0	0	0	1	1	1	1	1	D

D=1:开显示(DISPLAY ON) 意即显示器可以进行各种显示操作

D=0:关显示(DISPLAY OFF) 意即不能对显示器进行各种显示操作

2. 设置显示起始行(DISPLAY START LINE)

代码	R/W	D/I	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
形式	0	0	1	1	A5	A4	A3	A2	A1	A0

前面在 Z 地址计数器一节已经描述了显示起始行是由 Z 地址计数器控制的。

A5~A0 6 位地址自动送入 Z 地址计数器 起始行的地址可以是 0~63 的任意一行。

例如:

选择 A5~A0 是 62,则起始行与 DDRAM 行的对应关系如下:

DDRAM 行: 62 63 0 1 2 3 28 29

屏幕显示行: 1 2 3 4 5 6 31 32

3. 设置页地址 (SET PAGE "X ADDRESS")

代码	R/W	D/I	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
形式	0	0	1	0	1	1	1	A2	A1	A0

所谓页地址就是 DDRAM 的行地址,8 行为一页,模块共 64 行即 8 页,A2~A0 表示 0~7 页。读写数据对地址没有影响,页地址由本指令或 RST 信号改变复位后页地址为 0。页地址与 DDRAM 的对应关系见 DDRAM 地址表。

4. 设置 Y 地址(SET Y ADDRESS)

代码	R/W	D/I	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
形式	0	0	0	1	A5	A4	A3	A2	A1	A0

此指令的作用是将 A5~A0 送入 Y 地址计数器,作为 DDRAM 的 Y 地址指针。在对 DDRAM 进行读写操作后,Y 地址指针自动加 1,指向下一个 DDRAM 单元。

DDRAM 地址表:

表 4

CS2=1						CS1=1					
Y=	0	1		62	63	0	1		62	63	行号
X=	DB0	DB0	DB0	DB0	DB0	DB0	DB0	DB0	DB0	DB0	0
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
0	DB7	DB7	DB7	DB7	DB7	DB7	DB7	DB7	DB7	DB7	7
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
↓	DB0	DB0	DB0	DB0	DB0	DB0	DB0	DB0	DB0	DB0	8
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
	DB7	DB7	DB7	DB7	DB7	DB7	DB7	DB7	DB7	DB7	55
X=7	DB0	DB0	DB0	DB0	DB0	DB0	DB0	DB0	DB0	DB0	56
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
	DB7	DB7	DB7	DB7	DB7	DB7	DB7	DB7	DB7	DB7	63

5. 读状态(STATUS READ)

代码	R/W	D/I	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
形式	0	1	BUSY	0	ON/OFF	RET	0	0	0	0

当 R/W=1 D/I=0 时,在 E 信号为“H”的作用下,状态分别输出到数据总线(DB7~DB0)的相应位。

BF: 前面已叙述过(见 BF 标志位一节)。

ON/OFF: 表示 DFF 触发器的状态(见 DFF 触发器一节)。

RST: RST=1 表示内部正在初始化,此时组件不接受任何指令和数据。

6. 写显示数据(WRITE DISPLAY DATE)

代码	R/W	D/I	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
形式	0	1	D7	D6	D5	D4	D3	D2	D1	D0

D7~D0 为显示数据,此指令把 D7~D0 写入相应的 DDRAM 单元,Y 地址指针自动加 1。

7. 读显示数据(READ DISPLAY DATE)

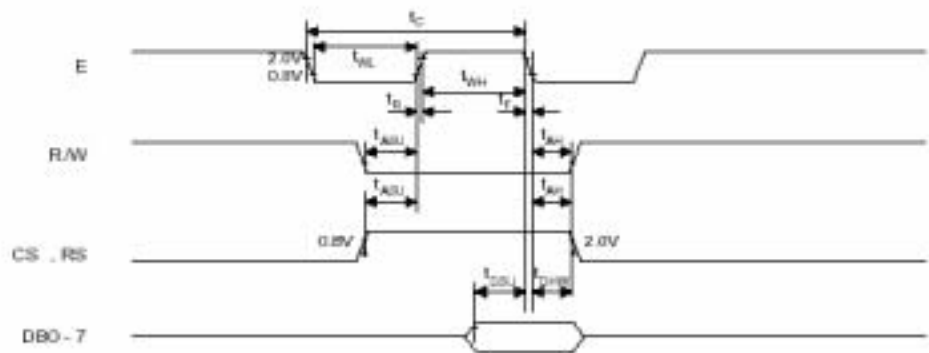
代码	R/W	D/I	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
形式	1	1	D7	D6	D5	D4	D3	D2	D1	D0

此指令把 DDRAM 的内容 D7~D0 读到数据总线 DB7~DB0,Y 地址指针自动加 1。

[Return](#)

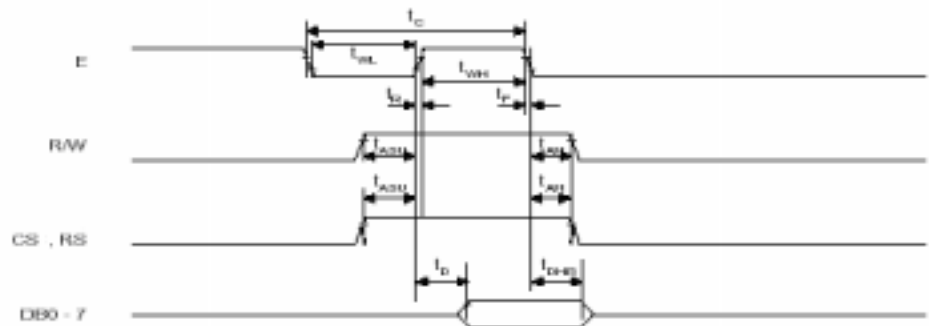
六、读写操作时序
1. 写操作时序

图 3



2. 读操作时序

图 4



3. 读写时序参数表

Characteristic	Symbol	Min	Typ	Max	Unit
E cycle	t_C	1000	-	-	ns
E high level width	t_{WH}	450	-	-	ns
E low level width	t_{WL}	450	-	-	ns
E rise time	t_R	-	-	25	ns
E fall time	t_F	-	-	25	ns
Address set-up time	t_{ASU}	140	-	-	ns
Address hold time	t_{AH}	10	-	-	ns
Data set-up time	t_{DSU}	200	-	-	ns
Data delay time	t_D	-	-	320	ns
Data hold time (write)	t_{DHW}	10	-	-	ns
Data hold time (read)	t_{DHR}	20	-	-	ns

[Return](#)

七．应用举例

CM12832-1 与单片机 8031 的一种接口如图 5. 所示：

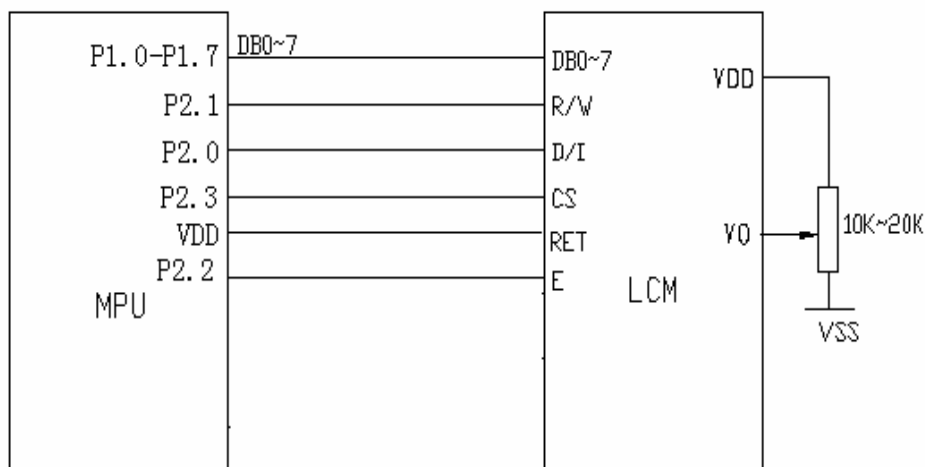


图 5

```

ORG    0000H
LJMP   MAIN
ORG    0003H
LJMP   ZHONGDUAN
ORG    0035H
ZHONGDUAN:
LCALL  DELAY
CLR    EA
JB     P3.2,$
SETB   EA
LCALL  DELAY
RETI
  
```

```

CS1    EQU    P2.3
CS2    EQU    P2.4
DI     EQU    P2.0
RW     EQU    P2.1
E      EQU    P2.2
DATA1  EQU    50H
DATA2  EQU    51H
DATA3  EQU    52H
  
```

*****写左半屏命令*****

WRITE_COMMAND_L:

```

    SETB  CS1
; CLR    CS2
    CLR   RW
    CLR   DI
    MOV   P1,DATA1
    SETB  E
    CLR   E
    CLR   CS1
    RET

;*****写左半屏数据*****
WRITE_DATA_L:
    SETB  CS1
; CLR    CS2
    CLR   RW
    SETB  DI
    MOV   P1,DATA2
    SETB  E
    CLR   E
    CLR   CS1
    RET

;*****写右半屏命令*****
WRITE_COMMAND_R:
    SETB  CS2
; CLR    CS1
    CLR   RW
    CLR   DI
    MOV   P1,DATA1
    SETB  E
    CLR   E
    CLR   CS2
    RET

;*****写右半屏数据*****
WRITE_DATA_R:
    SETB  CS2
; CLR    CS1
    CLR   RW
    SETB  DI
    MOV   P1,DATA2
    SETB  E
    CLR   E
    CLR   CS2
    RET

```

```

;*****延时程序*****
MS40:    MOV     R7,#0E8H
MS2:     MOV     R6,#0FFH
MS1:     DJNZ    R6,MS1
          DJNZ    R7,MS2
          RET
DELAY:   MOV     R5,#07H
DELAY1:  LCALL   MS40
          DJNZ    R5,DELAY1
          RET
;*****

```

```

MAIN:
    MOV     P2,#00H
    MOV     SP,#60H
    SETB    EA
    SETB    EX0
    SETB    ITO
    MOV     DATA1,#3EH
    LCALL   WRITE_COMMAND_L
    LCALL   WRITE_COMMAND_R
    MOV     DATA1,#3FH
    LCALL   WRITE_COMMAND_L
    LCALL   WRITE_COMMAND_R
    MOV     DATA1,#0C0H
    LCALL   WRITE_COMMAND_L
    LCALL   WRITE_COMMAND_R
    MOV     DATA1,#0B8H    ;PAGE ADDRESS
    LCALL   WRITE_COMMAND_L
    LCALL   WRITE_COMMAND_R
    MOV     DATA1,#40H    ;Y ADDRESS
    LCALL   WRITE_COMMAND_L
    LCALL   WRITE_COMMAND_R
    MOV     R1,#8
    MOV     DATA3,#0B8H
    MOV     DATA1,DATA3
M21:  MOV     DATA1,DATA3
    LCALL   WRITE_COMMAND_L
    LCALL   WRITE_COMMAND_R
    MOV     DATA1,#40H
    LCALL   WRITE_COMMAND_L
    LCALL   WRITE_COMMAND_R

```

```

        MOV        R2,#64
L21:    MOV        DATA2,#0AAH
        LCALL     WRITE_DATA_L
        LCALL     WRITE_DATA_R
        DJNZ      R2,L21
        INC       DATA3
        DJNZ      R1,M21
        LCALL     DELAY

        MOV        R1,#8
        MOV        DATA3,#0B8H
        MOV        DATA1,DATA3
M22:    MOV        DATA1,DATA3
        LCALL     WRITE_COMMAND_L
        LCALL     WRITE_COMMAND_R
        MOV        DATA1,#40H
        LCALL     WRITE_COMMAND_L
        LCALL     WRITE_COMMAND_R
        MOV        R2,#64
L22:    MOV        DATA2,#55H
        LCALL     WRITE_DATA_L
        LCALL     WRITE_DATA_R
        DJNZ      R2,L22
        INC       DATA3
        DJNZ      R1,M22
        LCALL     DELAY

        MOV        R1,#8
        MOV        DATA3,#0B8H
        MOV        DATA1,DATA3
M23:    MOV        DATA1,DATA3
        LCALL     WRITE_COMMAND_L
        LCALL     WRITE_COMMAND_R
        MOV        DATA1,#40H
        LCALL     WRITE_COMMAND_L
        LCALL     WRITE_COMMAND_R
        MOV        R2,#32
L23:    MOV        DATA2,#0FFH
        LCALL     WRITE_DATA_L
        LCALL     WRITE_DATA_R
        MOV        DATA2,#00H
        LCALL     WRITE_DATA_L
        LCALL     WRITE_DATA_R
        DJNZ      R2,L23

```

```

        INC      DATA3
        DJNZ     R1,M23
        LCALL    DELAY

        MOV      R1,#8
        MOV      DATA3,#0B8H
        MOV      DATA1,DATA3
M24:    MOV      DATA1,DATA3
        LCALL    WRITE_COMMAND_L
        LCALL    WRITE_COMMAND_R
        MOV      DATA1,#40H
        LCALL    WRITE_COMMAND_L
        LCALL    WRITE_COMMAND_R
        MOV      R2,#32
L24:    MOV      DATA2,#00H
        LCALL    WRITE_DATA_L
        LCALL    WRITE_DATA_R
        MOV      DATA2,#0FFH
        LCALL    WRITE_DATA_L
        LCALL    WRITE_DATA_R
        DJNZ     R2,L24
        INC      DATA3
        DJNZ     R1,M24
        LCALL    DELAY

        MOV      R1,#8
        MOV      DATA3,#0B8H
        MOV      DATA1,DATA3
M25:    MOV      DATA1,DATA3
        LCALL    WRITE_COMMAND_L
        LCALL    WRITE_COMMAND_R
        MOV      DATA1,#40H
        LCALL    WRITE_COMMAND_L
        LCALL    WRITE_COMMAND_R
        MOV      R2,#32
L25:    MOV      DATA2,#0AAH
        LCALL    WRITE_DATA_L
        LCALL    WRITE_DATA_R
        MOV      DATA2,#55H
        LCALL    WRITE_DATA_L
        LCALL    WRITE_DATA_R
        DJNZ     R2,L25
        INC      DATA3
        DJNZ     R1,M25

```

```

        LCALL    DELAY

M26:    MOV      R1,#8
        MOV      DATA3,#0B8H
        MOV      DATA1,DATA3
        MOV      DATA1,DATA3
        LCALL    WRITE_COMMAND_L
        LCALL    WRITE_COMMAND_R
        MOV      DATA1,#40H
        LCALL    WRITE_COMMAND_L
        LCALL    WRITE_COMMAND_R
        MOV      R2,#32
L26:    MOV      DATA2,#55H
        LCALL    WRITE_DATA_L
        LCALL    WRITE_DATA_R
        MOV      DATA2,#0AAH
        LCALL    WRITE_DATA_L
        LCALL    WRITE_DATA_R
        DJNZ     R2,L26
        INC      DATA3
        DJNZ     R1,M26
        LCALL    DELAY

        MOV      DPTR,#CHINESE1
        MOV      R1,#4
        MOV      DATA3,#0B8H
        MOV      DATA1,DATA3
M27:    MOV      DATA1,DATA3
        LCALL    WRITE_COMMAND_L
        MOV      DATA1,#40H
        LCALL    WRITE_COMMAND_L
        MOV      R2,#64
L27:    CLR      A
        MOVC     A,@A+DPTR
        MOV      DATA2,A
        LCALL    WRITE_DATA_L
        INC      DPTR
        DJNZ     R2,L27
        INC      DATA3
        DJNZ     R1,M27

        MOV      DPTR,#CHINESE2
        MOV      R1,#4

```

```

        MOV     DATA3,#0B8H
        MOV     DATA1,DATA3
M28:    MOV     DATA1,DATA3
        LCALL   WRITE_COMMAND_R
        MOV     DATA1,#40H
        LCALL   WRITE_COMMAND_R
        MOV     R2,#64
L28:
        CLR     A
        MOVC    A,@A+DPTR
        MOV     DATA2,A
        LCALL   WRITE_DATA_R
        INC     DPTR
        DJNZ    R2,L28
        INC     DATA3
        DJNZ    R1,M28
        LCALL   DELAY
        LJMP    MAIN

```

CHINESE1:

```

DB
0FFH,0FFH,003H,013H,053H,093H,033H,0DBH,00BH,08BH,06BH,043H,003H,083H
,043H,033H
DB
023H,003H,003H,003H,003H,003H,0F3H,053H,053H,053H,053H,053H,0F3H
,003H,003H
DB
003H,003H,083H,093H,093H,093H,0F3H,08BH,08BH,003H,013H,063H,003H,003H
,0FBH,003H
DB
003H,003H,043H,043H,043H,0FBH,043H,043H,003H,043H,043H,043H,0FBH,043H
,043H,043H
DB
0FFH,0FFH,000H,004H,084H,045H,034H,0FEH,015H,024H,0C4H,044H,001H,010H
,088H,044H
DB
03BH,012H,000H,0F8H,028H,028H,02BH,029H,0F9H,001H,001H,0F9H,029H,02BH
,028H,028H
DB
0F8H,000H,020H,010H,00CH,003H,0FFH,002H,024H,020H,021H,016H,010H,010H
,0FFH,008H
DB
008H,000H,008H,008H,004H,0FFH,002H,001H,000H,006H,09AH,062H,063H,092H
,00EH,002H

```

DB
0FFH,0FFH,000H,001H,000H,000H,000H,007H,000H,000H,002H,0C2H,021H,011H
,010H,010H

DB
030H,000H,000H,003H,0C1H,031H,0C1H,001H,003H,000H,010H,013H,0F1H,011H
,011H,001H

DB
003H,000H,000H,000H,000H,000H,007H,000H,010H,010H,0F0H,010H,010H,000H
,007H,010H

DB
010H,0F0H,010H,012H,004H,013H,0F0H,072H,082H,011H,0F0H,010H,0C0H,020H
,011H,013H

DB
0FFH,0FFH,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C7H,0C8H,0D0H
,0D0H,0D0H

DB
0C8H,0C0H,0D0H,0DCH,0D3H,0C2H,0D3H,0DCH,0D0H,0C0H,0D0H,0D0H,0DFH,0D0H
,0D0H,0C0H

DB
0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0F0H,0E0H,0E0H,0DFH,0C0H,0C0H,0C0H
,0C0H,0D0H

DB
0D0H,0DFH,0D0H,0D0H,0C0H,0D0H,0DFH,0D0H,0C3H,0CCH,0DFH,0C0H,0C7H,0C8H
,0D0H,0D2H

CHINESE2:

DB
043H,003H,003H,023H,023H,023H,023H,0A3H,063H,03BH,023H,023H,023H,023H
,023H,023H

DB
023H,003H,0F3H,013H,093H,073H,013H,003H,0F3H,053H,053H,053H,053H,053H
,0F3H,003H

DB
003H,003H,003H,003H,003H,083H,073H,023H,003H,003H,073H,083H,003H,003H
,003H,003H

DB
083H,093H,093H,093H,093H,093H,093H,093H,093H,093H,013H,013H,0F3H,003H
,0FFH,0FFH

DB
000H,000H,010H,008H,004H,002H,0FFH,049H,049H,049H,049H,049H,049H,0FFH
,000H,000H

DB
000H,000H,0FFH,000H,011H,022H,01CH,000H,0FFH,002H,006H,01AH,062H,092H
,00BH,000H

DB
000H,000H,008H,004H,082H,041H,020H,01CH,088H,080H,080H,0A0H,0C3H,086H
,00CH,004H

DB
000H,0FCH,024H,024H,024H,024H,024H,024H,07CH,000H,000H,000H,0FFH,000H
,0FFH,0FFH

DB
031H,000H,000H,000H,000H,000H,007H,000H,000H,000H,011H,0F2H,096H,043H
,030H,010H

DB
000H,010H,0F7H,010H,010H,090H,030H,000H,007H,002H,001H,000H,000H,000H
,001H,003H

DB
011H,010H,0F0H,011H,013H,001H,001H,011H,011H,0F0H,010H,010H,003H,001H
,000H,000H

DB
000H,000H,000H,000H,000H,000H,000H,000H,000H,000H,001H,002H,001H,000H
,0FFH,0FFH

DB
0CEH,0C2H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0D0H,0DFH,0D1H,0C3H
,0DCH,0D0H

DB
0C0H,0D0H,0DFH,0D1H,0D1H,0D3H,0D8H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H
,0C0H,0F0H

DB
0E0H,0E0H,0DFH,0C0H,0C0H,0C0H,0C0H,0D0H,0D0H,0DFH,0D0H,0D0H,0C0H,0C0H
,0C0H,0C0H

DB
0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H,0C0H
,0FFH,0FFH

END