

KS0108 液晶显示控制驱动器的应用

KS0108 液晶显示控制驱动器是一种带有驱动输出的图形液晶显示控制器，它可直接与 8 位微处理器相连，它可与 KS0107 配合对液晶屏进行行、列驱动。本手册将有选择的介绍 KS0108，详细的叙述内置 KS0108 的液晶显示模块 128*32、128*64 和 192*64 的应用方法。

●关于 KS0108 的介绍

KS0108 是一种带有列驱动输出的液晶显示控制器，它可与行驱动器 KS0107 配合使用，组成液晶显示驱动控制系统。

一，KS0108 的特点

- 1、内藏 $64 \times 64 = 4096$ 位显示 RAM，RAM 中每位数据对应 LCD 屏上一个点的亮、暗状态；
- 2、KS0108 是列驱动器，具有 64 路列驱动输出；
- 3、KS0108 读、写操作时序与 68 系列微处理器相符，因此它可直接与 68 系列微处理器接口相连；
- 4、KS0108 的占空比为 1/48--1/64。

二，KS0108 与微处理器的接口信号

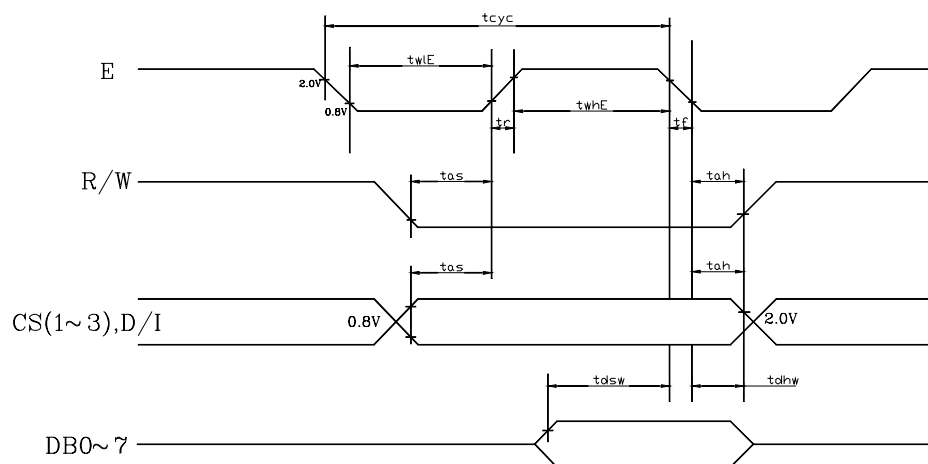
引脚符号	状态	引脚名称	功能
CS1,CS2,CS3	输入	芯片片选端	CS1 和 CS2 低电平选通，CS3 高电平选通。
E	输入	读写使能信号	在 E 下降沿，数据被锁存（写）入 KS0108；在 E 高电平期间，数据被读出
R/W	输入	读写选择信号	R/W=1 为读选通，R/W=0 为写选通
RS(也习惯叫做 D/I)	输入	数据、指令选择信号	RS=1 为数据操作 RS=0 为写指令或读状态
DB0-DB7	三态	数据总线	
RST	输入	复位信号	低电平有效，复位信号有效时，关闭液晶显示，使显示起始行为 0，RST 可跟 MPU 相连，由 MPU 控制；也可直接接 VDD，使之不起作用。

三，KS0108 的时序

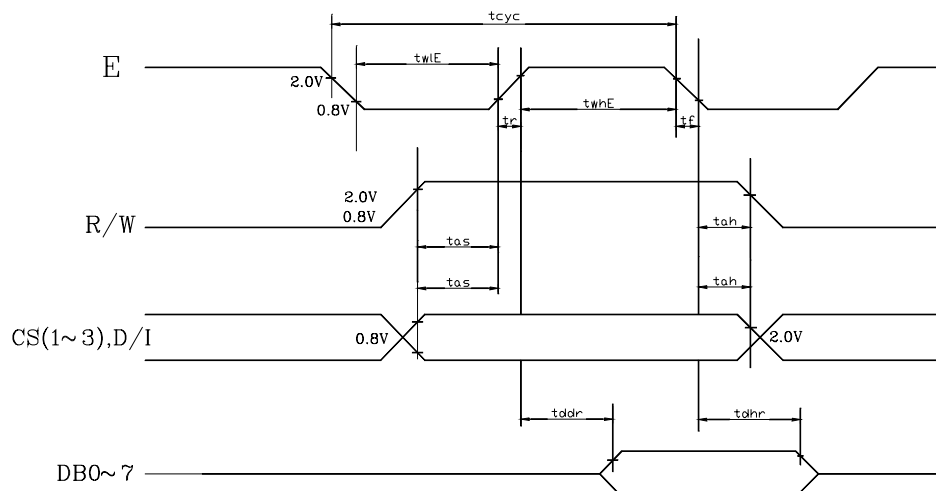
1，KS0108 具有能与 68 系列微处理器直接接口的时序，各种信号波形对照如下：

Characteristic	Symbol	Min.	Typ.	Max.	Unit
E 脉冲周期	tcyc	1000	---	---	ns
E 高电平宽度	twhE	450	---	---	ns
E 低电平宽度	twlE	450	---	---	ns
E 上升时间	tr	---	---	25	ns
E 下降时间	tf	---	---	25	ns
地址建立时间	tas	140	---	---	ns
地址保持时间	tah	10	---	---	ns
数据建立时间	tdsw	200	---	---	ns
数据延迟时间	tddr	----	---	320	ns
数据保持时间(写)	tdhw	10	---	---	ns
数据保持时间(读)	tdhr	20	---	---	ns

MPU Write Timong:



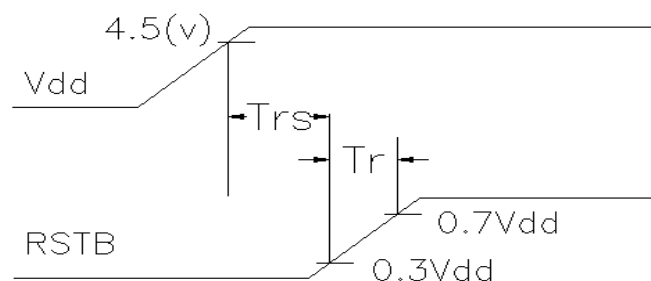
MPU Read Timong:



2, 复位时序

复位后，KS0108 显示关闭，显存地址归 0。复位条件如下表：

项目	符号	最小值	典型	最大	单位
复位时间	Trs	1.0	---	---	微秒
上升时间	Tr	---	---	200	纳秒



四. KS0108 显示 RAM 的地址结构

0	0	0	DB0						Page0							00H		
			DB1												01H			
			DB2												02H			
			DB3												03H			
			DB4												04H			
			DB5												05H			
			DB6												06H			
			DB7												07H			
0	0	1	DB0						Page1							08H		
			DB1												09H			
			DB2												0AH			
			DB3												0BH			
			DB4												0CH			
			DB5												0DH			
			DB6												0EH			
			DB7												0FH			
---	---	---		---	---	---	---		---	---	---	---	---		---			
1	1	1	DB0						Page6							38H		
			DB1												39H			
			DB2												3AH			
			DB3												3BH			
			DB4												3CH			
			DB5												3DH			
			DB6												3EH			
			DB7												3FH			
			00	01	02	03	04	05	-----				62	63				
			SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	-----				SEG63	SEG64				

五、KS0108 的指令系统

KS0108 的指令系统比较简单，总共只有七种。现分别介绍如下。

1、显示开/关指令

R/W	RS	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	1	1	1	1	1/0

当 DB0=1 时，LCD 显示 RAM 中的内容；DB0=0 时，关闭显示。

2、显示起始行（ROW）设置指令

R/W	RS	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	1	显示起始行（0-63）					

该指令设置了对应液晶屏最上一行的显示 RAM 的行号，有规律的改变显示起始行，可以使 LCD 实现显示滚屏的效果。

3、页（PAGE）设置指令

R/W	RS	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	0	1	1	1	1	页号（0-7）	

显示 RAM 共 64 行，分 8 页，每页 8 行。

4、列地址（Y Address）设置指令

R/W	RS	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	1	显示列地址（0-63）					

设置了页地址和列地址，就唯一确定了显示 RAM 中的一个单元，这样 MPU 就可以用读、写指令读出该单元中的内容或向该单元写进一个字节数据。

5、读状态指令

R/W	RS	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	0	BUSY	0	ON/OFF	REST	0	0	0	0

该指令用来查询 KS0108 的状态，各参量含义如下：

BUSY：1-内部在工作 0-正常状态

ON/OFF：1-显示关闭 0-显示打开

REST：1-复位状态 0-正常状态

在 BUSY 和 REST 状态时，除读状态指令外，其它指令均不对 KS0108 产生作用。

在对 KS0108 操作之前要查询 BUSY 状态，以确定是否可以对 KS0108 进行操作。

6、写数据指令

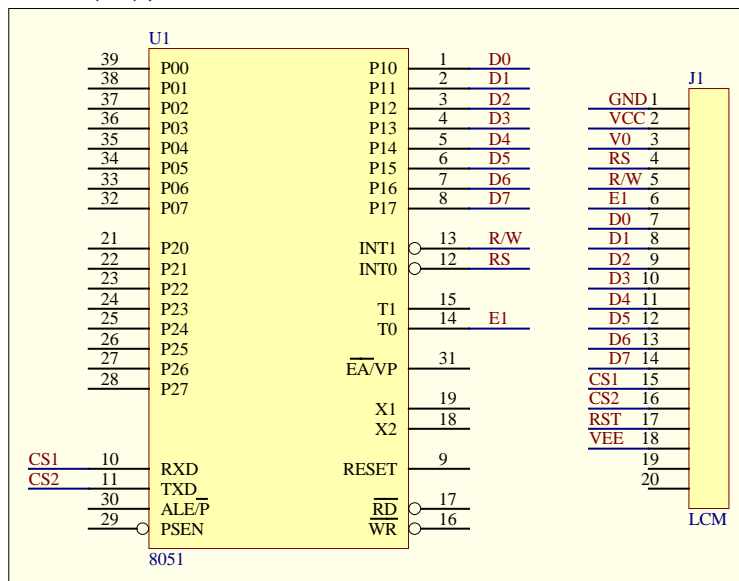
R/W	RS	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	1	写数据							

7、读数据指令

R/W	RS	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	1	读显示数据							

读、写数据指令每执行完一次读、写操作，列地址就自动增一，必须注意的是，进行读操作之前，必须有一次空读操作，紧接着再读才会读出所要读的单元中的数据。

2, 间接访问 (I/O 口控制)



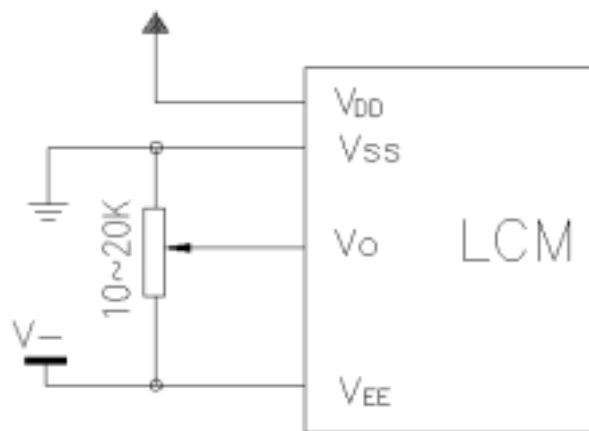
3, 片选

128*64 液晶显示模块的片选信号，高电平有效还是低电平有效取决于引出的KS0108的哪个片选。市场上常见的产品，高有效和低有效的都有，请查具体产品的规格书。

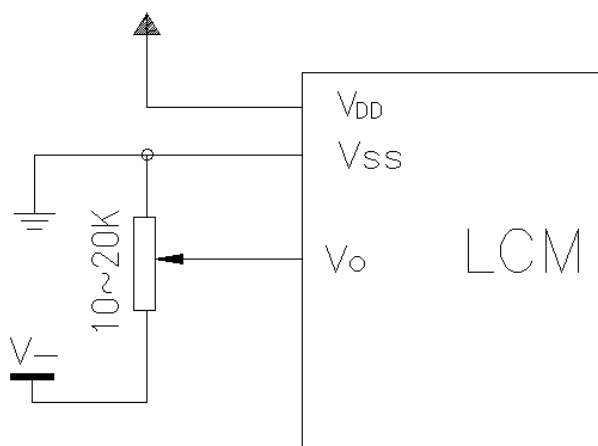
三, 电源供电和对比度调节

1, 双电源产品

双电源是指客户需要给液晶模块提供两路电压，一路是逻辑电压 V_{DD} ，即给液晶模块的逻辑电路供电的，一般是+5V（或+3V）。另一路是给液晶屏驱动用的，1/64 占空的液晶屏一般需要 8~15V 之间的电压驱动。所以需要客户提供一路负电压 (V_{EE})，-5V~ -10V，这样 V_{DD} 和 V_{EE} 之间有 10~15V 的压降，用做液晶的驱动电压。这种方式下的电源接法如下， V_- 就是客户要提供的负压，电位器起调节显示深浅的作用。

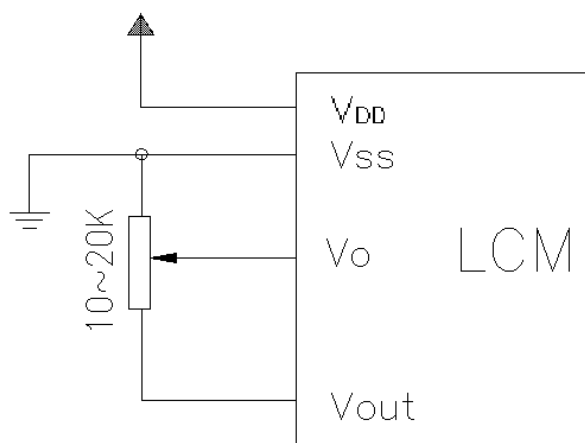


也有些产品的接口将 VEE 端省略了，只有 Vo 端，其电源接法如下：



2. 单电源产品

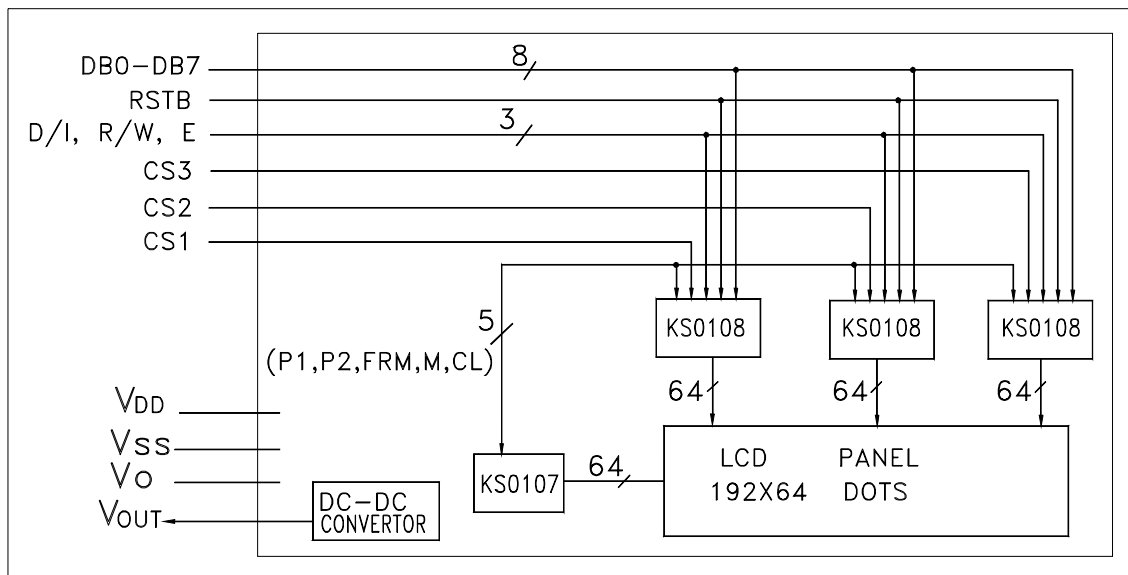
单电源产品是指客户之需给液晶模块提供一路电压，即逻辑电压 V_{DD}，一般+5V（或+3V），液晶模块内部集成了 DC-DC 转换电路，而液晶屏的驱动电压则由 DC-DC 转换电路提供。一般这类产品的接口中，没有 VEE 端子，而取代之的是 V_{OUT} 端子，即液晶模块内部的 DC-DC 转换电路生成的负电压的输出端子，一般电压为-5V 或-10V 左右。这种产品一般仍需要客户外接电位器来调节显示深浅。其供电电路如下：



● 192*64 图形点阵模块的应用

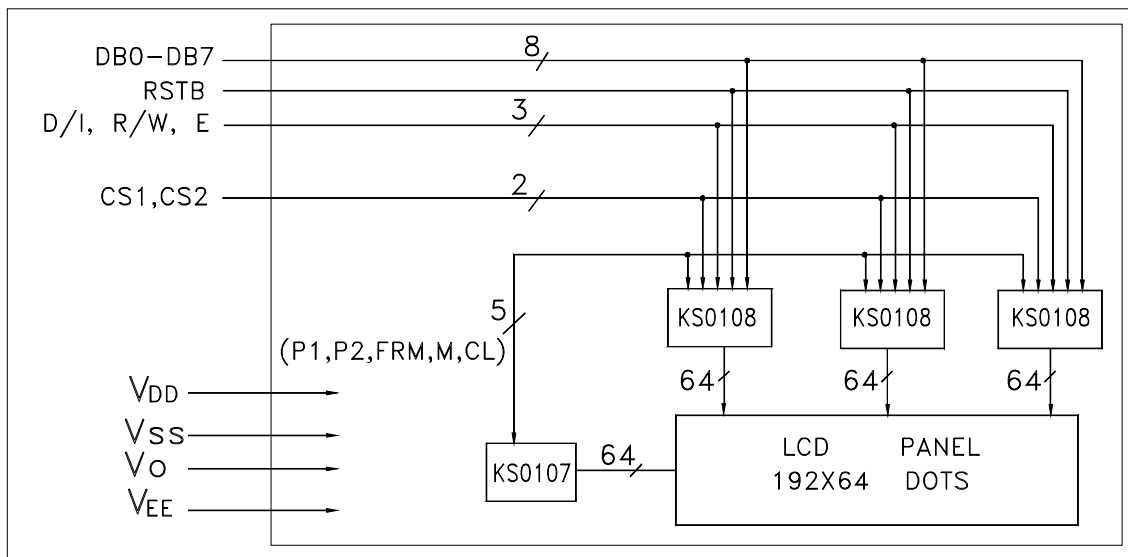
一，逻辑图

1，三个片选端的产品



这种 192*64 液晶显示模块的每一片 KS0108 都引出一个片选信号，高电平有效还是低电平有效取决于引出的 KS0108 的哪个片选。

2，两个片选信号的产品

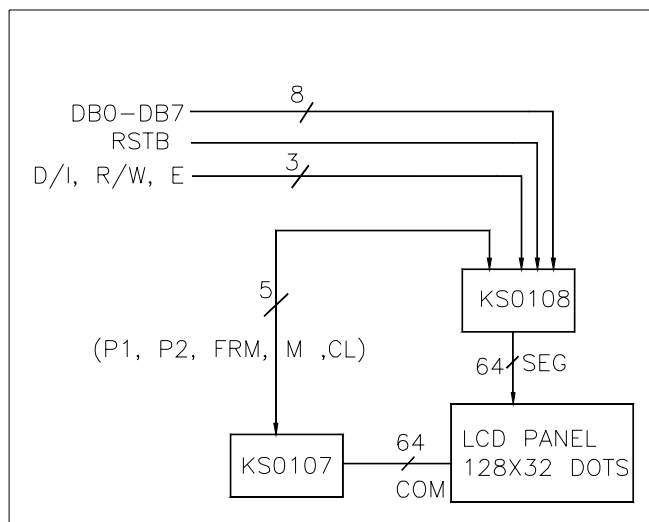


这种 192*64 液晶显示模块的两个片选信号分别接到三个 KS0108 的不同的片选端子上，一般此类产品的选通方法：CS1、CA2=00 时，选通左边 KS0108，CS1、CA2=01 时，选通中间的 KS0108，CS1、CA2=10 时，选通右边 KS0108。（以上为常见产品的片选关系，可能某些型号的产品中间和右边的 IC 的选通值相反，请注意看具体产品的规格书。）

二，192*64 产品的接口及供电电路请参考 128*64 产品的例图。

● 128*32 图形点阵模块的应用

128*32 图形点阵模块的逻辑图：



如图所示，内置 KS0108 的 128*32 的液晶显示模块，其实相当于 64*64 的模块，将显示内容分成 64*32 的上下两部分，而把下半部分移到上半部分的后边构成的。

一般在 128*64 的产品中，片选信号如果没有引到接口上，说明在模块内部已经接到相应的有效电平上了。

其他方面的应用请参考 128*64 及 129*64 模块的应用。

● 以 192*64 图形点阵模块为例介绍 KS0108 的软件编程

一、 直接访问方式

(一) 接口电路

(二) 直接访问方式驱动子程序

A11=/CSB, A10=/CSA, A9=R/W, A8=A0

COM	EQU	20H	; 指令寄存器
DAT	EQU	21H	; 数据寄存器
CWADD1	EQU	2000H	; 写指令代码地址 (左)
CRADD1	EQU	2001H	; 读状态字地址 (左)
DWADD1	EQU	2002H	; 写显示数据地址 (左)
DRADD1	EQU	2003H	; 读显示数据地址 (左)
CWADD2	EQU	2004H	; 写指令代码地址 (中)
CRADD2	EQU	2005H	; 读状态字地址 (中)
DWADD2	EQU	2006H	; 写显示数据地址 (中)
DRADD2	EQU	2007H	; 读显示数据地址 (中)
CWADD3	EQU	2008H	; 写指令代码地址 (右)
CRADD3	EQU	2009H	; 读状态字地址 (右)
DWADD3	EQU	200AH	; 写显示数据地址 (右)

DRADD2 EQU 200BH ; 读显示数据地址 (右)
1 写指令代码子程序 (左)

```
PRL0:  PUSH    DPL                ; 片选设置为 "00"
        PUSH    DPH
        MOV     DPTR, #CRADD1    ; 设置读状态字地址
PRL01:  MOVX    A, @DPTR          ; 读状态字
        JB      ACC.7, PRL01     ; 判 "忙" 标志为 "0" 否, 否在读
        MOV     DPTR, #CWADD1    ; 设置写指令代码地址
        MOV     A, COM           ; 取指令代码
        MOVX    @DPTR, A        ; 写指令代码
        POP     DPH
        POP     DPL
        RET
```

2 写显示数据子程序 (左)

```
PRL1:   PUSH    DPL                ; 片选设置为 "00"
        PUSH    DPH
        MOV     DPTR, #CRADD1    ; 设置读状态字地址
PRL11:  MOVX    A, @DPTR          ; 读状态字
        JB      ACC.7, PRL11     ; 判 "忙" 标志为 "0" 否, 否在读
        MOV     DPTR, #DWADD1    ; 设置写显示数据地址
        MOV     A, DAT           ; 取数据
        MOVX    @DPTR, A        ; 写数据
        POP     DPH
        POP     DPL
        RET
```

3 读显示数据子程序 (左)

```
PRL2:   PUSH    DPL                ; 片选设置为 "00"
        PUSH    DPH
        MOV     DPTR, #CRADD1    ; 设置读状态字地址
PRL21:  MOVX    A, @DPTR          ; 读状态字
        JB      ACC.7, PRL21     ; 判 "忙" 标志为 "0" 否, 否在读
        MOV     DPTR, #DRADD1    ; 设置读显示数据地址
        MOVX    A, @DPTR        ; 读数据
        MOV     DAT, A          ; 存数据
        POP     DPH
        POP     DPL
        RET
```

4 写指令代码子程序 (中)

```
PRM0:   PUSH    DPL                ; 片选设置为 "01"
        PUSH    DPH
        MOV     DPTR, #CRADD2    ; 设置读状态字地址
PRM01:  MOVX    A, @DPTR          ; 读状态字
        JB      ACC.7, PRM01     ; 判 "忙" 标志为 "0" 否, 否在读
        MOV     DPTR, #CWADD2    ; 设置写指令代码地址
        MOV     A, COM           ; 取指令代码
```

```

MOVX    @DPTR, A      ; 写指令代码
POP      DPH
POP      DPL
RET

```

5 写显示数据子程序（中）

```

PRM1:   PUSH    DPL      ; 片选设置为“01”
        PUSH    DPH
        MOV     DPTR, #CRADD2 ; 设置读状态字地址
PRM11:  MOVX    A, @DPTR   ; 读状态字
        JB      ACC.7, PRM11 ; 判“忙”标志为“0”否，否在读
        MOV     DPTR, #DWADD2 ; 设置写显示数据地址
        MOV     A, DAT      ; 取数据
        MOVX    @DPTR, A   ; 写数据
        POP     DPH
        POP     DPL
        RET

```

6 读显示数据子程序（中）

```

PRM2:   PUSH    DPL      ; 片选设置为“01”
        PUSH    DPH
        MOV     DPTR, #CRADD2 ; 设置读状态字地址
PRM21:  MOVX    A, @DPTR   ; 读状态字
        JB      ACC.7, PRM21 ; 判“忙”标志为“0”否，否在读
        MOV     DPTR, #DRADD2 ; 设置读显示数据地址
        MOVX    A, @DPTR   ; 读数据
        MOV     DAT, A     ; 存数据
        POP     DPH
        POP     DPL
        RET

```

7 写指令代码子程序（右）

```

PRR0:   PUSH    DPL      ; 片选设置为“10”
        PUSH    DPH
        MOV     DPTR, #CRADD3 ; 设置读状态字地址
PRR01:  MOVX    A, @DPTR   ; 读状态字
        JB      ACC.7, PRR01 ; 判“忙”标志为“0”否，否在读
        MOV     DPTR, #CWADD3 ; 设置写指令代码地址
        MOV     A, COM      ; 取指令代码
        MOVX    @DPTR, A   ; 写指令代码
        POP     DPH
        POP     DPL
        RET

```

8 写显示数据子程序（右）

```

PRR1:   PUSH    DPL      ; 片选设置为“10”
        PUSH    DPH
        MOV     DPTR, #CRADD3 ; 设置读状态字地址
PRR11:  MOVX    A, @DPTR   ; 读状态字
        JB      ACC.7, PRR11 ; 判“忙”标志为“0”否，否在读

```

```

MOV    DPTR, #DWADD3    ; 设置写显示数据地址
MOV    A, DAT            ; 取数据
MOVX   @DPTR, A          ; 写数据
POP    DPH
POP    DPL
RET

```

9 读显示数据子程序（右）

```

PRR2:  PUSH    DPL                ; 片选设置为“10”
        PUSH    DPH
        MOV     DPTR, #CRADD3    ; 设置读状态字地址
PRR21: MOVX   A, @DPTR            ; 读状态字
        JB      ACC.7, PRR21     ; 判“忙”标志为“0”否，否在读
        MOV     DPTR, #DRADD3    ; 设置读显示数据地址
        MOVX   A, @DPTR            ; 读数据
        MOV     DAT, A           ; 存数据
        POP     DPH
        POP     DPL
        RET

```

二.间接控制方式接口电路

（二）间接控制方式驱动子程序

```

CSA     EQU     P3.0    ; 片选/CSA
CSB     EQU     P3.1    ; 片选/CSB
RS      EQU     P3.2    ; 寄存器选择信号
R/W     EQU     P3.3    ; 读写选择信号
E       EQU     P3.4    ; 使能信号

```

1、写指令代码子程序（左）

```

PRL0:  CLR     CSA                ; 片选设置为“00”
        CLR     CSB
        CLR     RS                ; RS=0
        SETB    R/W              ; R/W=1
PRL01: MOV     P1, #0FFH          ; P1口置“1”
        SETB    E                ; E=1
        MOV     A, P1            ; 读状态字
        CLR     E                ; E=0
        JB      ACC.7, PRL01     ; 判“忙”标志为“0”否，否在读
        CLR     R/W              ; RW=0
        MOV     P1, COM          ; 写指令代码
        SETB    E                ; E=1

```

CLR E ; E=0

RET

2、 写显示数据子程序（左）

PRL1: CLR CSA ; 片选设置为“00”
CLR CSB
CLR RS ; RS=0
SETB R/W ; R/W=1
PRL11: MOV P1, #0FFH ; P1 口置“1”
SETB E ; E=1
MOV A, P1 ; 读状态字.
CLR E ; E=0
JB ACC.7, PRL11 ; 判“忙”标志为“0”否, 否在读
SETB RS ; RS=1
CLR R/W ; RW=0
MOV P1, DAT ; 写数据
SETB E ; E=1
CLR E ; E=0
RET

3 读显示数据子程序（左）

PRL2: CLR CSA ; 片选设置为“00”
CLR CSB
CLR RS ; RS=0
SETB R/W ; R/W=1
PRL21: MOV P1, #0FFH ; P1 口置“1”
SETB E ; E=1
MOV A, P1 ; 读状态字.
CLR E ; E=0
JB ACC.7, PRL21 ; 判“忙”标志为“0”否, 否在读
SETB RS ; RS=1
MOV P1, #0FFH ; P1 口置“1”
SETB E ; E=1
MOV DAT, P1 ; 读数据
CLR E ; E=0
RET

4、 写指令代码子程序（中）

PRM0: CLR CSA ; 片选设置为“01”
SETB CSB
CLR RS ; RS=0
SETB R/W ; R/W=1
PRM01: MOV P1, #0FFH ; P1 口置“1”
SETB E ; E=1
MOV A, P1 ; 读状态字.
CLR E ; E=0

```
JB      ACC.7, PRM01 ; 判“忙”标志为“0”否，否在读
CLR     R/W          ; RW=0
MOV     P1, COM      ; 写指令代码
SETB    E            ; E=1
CLR     E            ; E=0
RET
```

5、 写显示数据子程序（中）

```
PRM1:   CLR     CSA          ; 片选设置为“01”
        SETB    CSB
        CLR     RS          ; RS=0
        SETB    R/W         ; R/W=1
PRM11:  MOV     P1, #0FFH    ; P1 口置“1”
        SETB    E          ; E=1
        MOV     A, P1       ; 读状态字.
        CLR     E          ; E=0
        JB      ACC.7, PRM11 ; 判“忙”标志为“0”否，否在读
        SETB    RS         ; RS=1
        CLR     R/W         ; RW=0
        MOV     P1, DAT     ; 写数据
        SETB    E          ; E=1
        CLR     E          ; E=0
        RET
```

6、读显示数据子程序（中）

```
PRM2:   CLR     CSA          ; 片选设置为“01”
        SETB    CSB
        CLR     RS          ; RS=0
        SETB    R/W         ; R/W=1
PRM21:  MOV     P1, #0FFH    ; P1 口置“1”
        SETB    E          ; E=1
        MOV     A, P1       ; 读状态字.
        CLR     E          ; E=0
        JB      ACC.7, PRM21 ; 判“忙”标志为“0”否，否在读
        SETB    RS         ; RS=1
        MOV     P1, #0FFH    ; P1 口置“1”
        SETB    E          ; E=1
        MOV     DAT, P1     ; 读数据
        CLR     E          ; E=0
        RET
```

7、 写指令代码子程序（右）

```
PRR0:   SETB    CSA          ; 片选设置为“10”
        CLR     CSB
        CLR     RS          ; RS=0
        SETB    R/W         ; R/W=1
```

```
PRR01: MOV    P1, #0FFH    ; P1 口置 “1”
        SETB   E           ; E=1
        MOV    A, P1       ; 读状态字.
        CLR    E           ; E=0
        JB     ACC.7, PRR01 ; 判 “忙” 标志为 “0” 否, 否在读
        CLR    R/W         ; RW=0
        MOV    P1, COM     ; 写指令代码
        SETB   E           ; E=1
        CLR    E           ; E=0
        RET
```

8、 写显示数据子程序（右）

```
PRR1:  SETB    CSA          ; 片选设置为 “10”
        CLR    CSB
        CLR    RS           ; RS=0
        SETB   R/W          ; R/W=1
PRR11: MOV     P1, #0FFH    ; P1 口置 “1”
        SETB   E           ; E=1
        MOV    A, P1       ; 读状态字.
        CLR    E           ; E=0
        JB     ACC.7, PRR11 ; 判 “忙” 标志为 “0” 否, 否在读
        SETB   RS          ; RS=1
        CLR    R/W         ; RW=0
        MOV    P1, DAT     ; 写数据
        SETB   E           ; E=1
        CLR    E           ; E=0
        RET
```

9、 读显示数据子程序（右）

```
PRR2:  SETB    CSA          ; 片选设置为 “10”
        CLR    CSB
        CLR    RS           ; RS=0
        SETB   R/W          ; R/W=1
PRR21: MOV     P1, #0FFH    ; P1 口置 “1”
        SETB   E           ; E=1
        MOV    A, P1       ; 读状态字.
        CLR    E           ; E=0
        JB     ACC.7, PRR21 ; 判 “忙” 标志为 “0” 否, 否在读
        SETB   RS          ; RS=1
        MOV    P1, #0FFH    ; P1 口置 “1”
        SETB   E           ; E=1
        MOV    DAT, P1     ; 读数据
        CLR    E           ; E=0
        RET
```

三、应用子程序

在使用该程序之前应根据使用的系统调用相应的驱动子程序，修改口地址，若使用 128*64，则将（左）PRL1 驱动子程序屏蔽。

1、初始化子程序

```
INT:  MOV      COM, #0C0H      ; 设置显示起始行为第一行
      LCALL    PRL0
      LCALL    PRM0
      LCALL    PRR0
      MOV      COM, #3FH      ; 开显示设置
      LCALL    PRL0
      LCALL    PRM0
      LCALL    PRR0
      RET
```

2、清显示 RAM 区（清屏）子程序

```
CLEAR: MOV      R4, #00H      ; 页面地址暂存器设置
CLEAR1: MOV     A, R4
      ORL      A, #0B8H      ; “或” 页面地址设置代码
      MOV      COM, A        ; 页面地址设置
      LCALL    PRL0
      LCALL    PRM0
      LCALL    PRR0
      MOV      COM, #40H      ; 列地址设置为 “0”
      LCALL    PRL0
      LCALL    PRM0
      LCALL    PRR0
      MOV      R3, #40H      ; 一页清 64 个字节
CLEAR2: MOV     DAT, #00H      ; 显示数据为 “0”
      LCALL    PRL1
      LCALL    PRM1
      LCALL    PRR1
      DJNZ     R3, CLEAR2      ; 页内字节清零循环
      INC      R4              ; 页地址暂存器加 1
      CJNE     R4, #08H, CLEAR1 ; RAM 区清零循环
      RET
```

示例程序：

```
MAIN:  MOV      SP, #60H
      ANL      P3, #0E0H
      LCALL    INT
      LCALL    CLEAR
```

3、西文字符写入子程序

```
COLUMN EQU      30H  ; 列地址寄存器（0-191）
PAGE    EQU      31H  ; 页地址寄存器 D2, D1, D0: 页地址
                        ; D7: 字符体 D7=0 为 6X8 点阵
```



```

                                ; D7=1 为 8X8 点阵
CODE EQU 32H ; 字符代码寄存器
COUNT EQU 33H ; 计数器
CW__PR: MOV DPTR, #CTAB ; 确定字符字模块首地址
MOV A, CODE ; 取代码
MOV B, #08H ; 字模块宽度为 8 个字节
MUL AB ; 代码 X8
ADD A, DPL ; 字符字模块首地址
MOV DPL, A ; =字模库首地址+代码 X8
MOV AB
ADDC A, DPH
MOV DPH, A
MOV CODE, #00H ; 借用为间址寄存器
MOV A, PAGE ; 读页地址寄存器
JB ACC.7, CW_1 ; 判字符体
MOV COUNT, #06H ; 6X8 点阵
LJMP CW_2
CW_1: MOV COUNT, #08H ; 8X8 点阵
CW_2: ANL A, #07H ; 取页地址值
ORL A, #0B8H ; “或” 页地址指令代码
MOV COM, A ; 写页地址指针
LCALL PRL0
LCALL PRM0
LCALL PRR0
MOV A, COLUMN ; 读列地址寄存器
CLR C
SUBB A, #40H ; 列地址-64
JC CW_3 ; <0 为左屏显示区域
MOV COLUMN, A
SUBB A, #40H ; 列地址-64
JC CW_21 ; <0 为中屏显示区域
MOV COLUMN, A ; ≥0 为右屏显示区域
MOV A, PAGE
SETB ACC.5 ; 设置区域标志位
MOV PAGE, A ; “00” 为左, “01” 为中, “10” 为右
LJMP CW_3
CW_21: MOV A, PAGE
SETB ACC.4 ; 设置区域标志位
MOV PAGE, A
CW_3: MOV COM, COLUMN ; 设置列地址值
ORL COM, #40H ; “或” 列地址指令标志位
MOV A, PAGE ; 判区域标志以确定设置哪个控制器
ANL A, #30H

```

```

        CJNE    A, #10H, CW_31 ; “01” 为中区
        LCALL   PRM0
        LJMP    CW_4
CW_31:  CJNE    A, #20H, CW_32 ; “10” 为右区
        LCALL   PRR0
        LJMP    CW_4
CW_32:  LCALL   PRL0           ; “00” 为左区
CW_4:   MOV     A, CODE         ; 取间址寄存器值
        MOVC    A, @A+DPTR     ; 取字符字模数据
        MOV     DAT, A         ; 写数据
        MOV     A, PAGE        ; 判区域标志
        ANL     A, #30H
        CJNE    A, #10H, CW_41 ; “01” 为中区
        LCALL   PRM1
        LJMP    CW_5
CW_41:  CJNE    A, #20H, CW_42 ; “10” 为右区
        LCALL   PRR1
        LJMP    CW_5
CW_42:  LCALL   PRL1           ; “00” 为左区
CW_5:   INC     CODE           ; 间址加 1
        INC     COLUMN         ; 列地址加 1
        MOV     A, COLUMN      ; 判列地址是否超出区域范围
        CJNE    A, #40H, CW_6
CW_6:   JC      CW_9           ; 未超出则继续
        MOV     COLUMN, #00H
        MOV     A, PAGE        ; 超出则判在何区域
        JB      ACC.5, CW_9     ; 在右区域则退出
        JB      ACC.4, CW_61    ; 判在左或中区
        SETB    ACC.4          ; 在左区则转中区
        MOV     PAGE, A
        MOV     COM, #40H      ; 设置中区列地址为 “0”
        LCALL   PRM0
        LJMP    CW_9
CW_61:  SETB    ACC.5          ; 在中区则转右区
        CLR     ACC.4
        MOV     PAGE, A
        MOV     COM, #40H      ; 设置右区列地址为 “0”
        LCALL   PRR0
CW_9:   DJNZ    COUNT, CW_4    ; 循环
        RET

```

西文显示演示程序段

```

        MOV     PAGE, #05H     ; 6X8 点阵字体, 第 4 页
        MOV     COLUMN, #30H   ; 起始列为第 4 列

```

```

MOV     CODE, #34H      ; 字符代码
LCALL   CW_PR
MOV     PAGE, #05H      ;
MOV     COLUMN, #3CH    ;
MOV     CODE, #45H
LCALL   CW_PR
MOV     PAGE, #05H      ;
MOV     COLUMN, #48H    ;
MOV     CODE, #4CH
LCALL   CW_PR
MOV     PAGE, #05H      ;
MOV     COLUMN, #54H    ;
MOV     CODE, #1AH
LCALL   CW_PR
MOV     R7, #00H
MOV     R6, 60H

```

```

LOOP:   MOV     A, R7
        MOV     DPTR, #TAB1
        MOVC    A, @A+DPTR
        MOV     CODE,A
        MOV     PAGE,#85H      ; 8X8 点阵字体, 第 4 页
        MOV     COLUMN, R6
        LCALL   CW_PR
        INC     R7
        MOV     A, #06H
        ADD     A, R6
        MOV     R6, A
        CJNE    R7, #08H, LOOP
        SJMP     $

```

```

TAB1:   DB  16H, 12H, 17H, 18H, 10H, 18H, 16H, 16H

```

西文字符库

```

CTAB:   DB 000H,000H,000H,000H,000H,000H,000H,000H ; “ ” =00H
        DB 000H,000H,000H,04FH,000H,000H,000H,000H ; “!”=01H
        DB 000H,000H,007H,000H,007H,000H,000H,000H, ; “”” =02H
        DB 000H,014H,07FH,014H,07FH,014H,000H,000H ; “#” =03H
        DB 000H,024H,02AH,07FH,02AH,012H,000H,000H ; “$” =04H
        DB 000H,023H,013H,008H,064H,062H,000H,000H ; “%” =05H
        DB 000H,036H,049H,055H,022H,050H,000H,000H ; “&” =06H
        DB 000H,000H,005H,003H,000H,000H,000H,000H ; “^” =07H
        DB 000H,000H,01CH,022H,041H,000H,000H,000H ; “(” =08H

```

DB 000H,000H,041H,022H,01CH,000H,000H,000H ; “)” =09H
DB 000H,014H,008H,03EH,008H,014H,000H,000H ; “*” =0AH
DB 000H,008H,008H,03EH,0H08,008H,000H,000H ; “+” =0BH
DB 000H,000H,050H,030H,000H,000H,000H,000H ; “;” =0CH
DB 000H,008H,008H,008H,008H,000H,000H,000H ; “-” =0DH
DB 000H,000H,060H,060H,000H,000H,000H,000H ; “.” =0EH
DB 000H,020H,010H,008H,004H,002H,000H,000H ; “/” =0FH
DB 000H,03EH,051H,049H,045H,03EH,000H,000H ; “0” =10H
DB 000H,000H,042H,07FH,040H,000H,000H,000H ; “1” =11H
DB 000H,042H,061H,051H,049H,046H,000H,000H ; “2” =12H
DB 000H,021H,041H,045H,04BH,031H,000H,000H ; “3” =13H
DB 000H,018H,014H,012H,07FH,010H,000H,000H ; “4” =14H
DB 000H,027H,045H,045H,045H,039H,000H,000H ; “5” =15H
DB 000H,03CH,04AH,049H,049H,030H,000H,000H ; “6” =16H
DB 000H,001H,001H,079H,005H,003H,000H,000H ; “7” =17H
DB 000H,036H,049H,049H,049H,036H,000H,000H ; “8” =18H
DB 000H,006H,049H,049H,029H,01EH,000H,000H ; “9” =19H
DB 000H,000H,036H,036H,000H,000H,000H,000H ; “:” =1AH
DB 000H,000H,056H,036H,000H,000H,000H,000H ; “;” =1BH
DB 000H,008H,014H,022H,041H,000H,000H,000H ; “<” =1CH
DB 000H,014H,014H,014H,014H,014H,000H,000H ; “=” =1DH
DB 000H,000H,041H,022H,014H,008H,000H,000H ; “>” =1EH
DB 000H,002H,001H,051H,009H,006H,000H,000H ; “?” =1FH
DB 000H,032H,049H,079H,041H,03EH,000H,000H ; “@” =20H
DB 000H,07EH,011H,011H,011H,07EH,000H,000H ; “A” =21H
DB 000H,041H,07FH,049H,049H,036H,000H,000H ; “B” =22H
DB 000H,03EH,041H,041H,041H,022H,000H,000H ; “C” =23H
DB 000H,041H,07EH,041H,041H,003H,000H,000H ; “D” =24H
DB 000H,07EH,049H,049H,049H,049H,000H,000H ; “E” =25H
DB 000H,07FH,009H,009H,009H,001H,000H,000H ; “F” =26H
DB 000H,03EH,041H,041H,049H,07AH,000H,000H ; “G” =27H
DB 000H,07FH,008H,008H,008H,07FH,000H,000H ; “H” =28H
DB 000H,000H,041H,07FH,041H,000H,000H,000H ; “I” =29H
DB 000H,020H,040H,041H,03FH,001H,000H,000H ; “J” =2AH
DB 000H,07FH,008H,014H,022H,041H,000H,000H ; “K” =2BH
DB 000H,07FH,040H,040H,040H,040H,000H,000H ; “L” =2CH
DB 000H,07FH,002H,00CH,002H,07FH,000H,000H ; “M” =2DH
DB 000H,07FH,006H,008H,030H,07FH,000H,000H ; “N” =2EH
DB 000H,03EH,041H,041H,041H,03EH,000H,000H ; “O” =2FH
DB 000H,07FH,009H,009H,009H,006H,000H,000H ; “P” =30H
DB 000H,03EH,041H,051H,021H,05EH,000H,000H ; “Q” =31H
DB 000H,07FH,009H,019H,029H,046H,000H,000H ; “R” =32H
DB 000H,026H,049H,049H,049H,032H,000H,000H ; “S” =33H
DB 000H,001H,001H,07FH,001H,001H,000H,000H ; “T” =34H

DB 000H,03FH,040H,040H,040H,03FH,000H,000H ; “U” =35H
DB 000H,01FH,020H,040H,020H,01FH,000H,000H ; “V” =36H
DB 000H,07FH,020H,018H,020H,07FH,000H,000H ; “W” =37H
DB 000H,063H,014H,008H,014H,063H,000H,000H ; “X” =38H
DB 000H,007H,008H,070H,008H,007H,000H,000H ; “Y” =39H
DB 000H,061H,051H,049H,045H,043H,000H,000H ; “Z” =3AH
DB 000H,000H,07FH,041H,041H,000H,000H,000H ; “[” =3BH
DB 000H,002H,004H,008H,010H,020H,000H,000H ; “\” =3CH
DB 000H,000H,041H,041H,07FH,000H,000H,000H ; “]” =3DH
DB 000H,004H,002H,001H,002H,004H,000H,000H ; “^” =3EH
DB 000H,040H,040H,000H,040H,040H,000H,000H ; “-” =3FH
DB 000H,001H,002H,004H,000H,000H,000H,000H ; “¹” =40H
DB 000H,020H,054H,054H,054H,078H,000H,000H ; “a” =41H
DB 000H,07FH,048H,044H,044H,038H,000H,000H ; “b” =42H
DB 000H,038H,044H,044H,044H,028H,000H,000H ; “c” =43H
DB 000H,038H,044H,044H,048H,07FH,000H,000H ; “d” =44H
DB 000H,038H,054H,054H,054H,018H,000H,000H ; “e” =45H
DB 000H,000H,008H,07EH,009H,002H,000H,000H ; “f” =46H
DB 000H,00CH,052H,052H,04CH,03EH,000H,000H ; “g” =47H
DB 000H,07FH,008H,004H,004H,078H,000H,000H ; “h” =48H
DB 000H,000H,044H,07DH,040H,000H,000H,000H ; “i” =49H
DB 000H,020H,040H,044H,03DH,000H,000H,000H ; “j” =4AH
DB 000H,000H,07FH,010H,028H,044H,000H,000H ; “k” =4BH
DB 000H,000H,041H,07FH,040H,000H,000H,000H ; “l” =4CH
DB 000H,07CH,004H,078H,004H,078H,000H,000H ; “m” =4DH
DB 000H,07CH,008H,004H,004H,078H,000H,000H ; “n” =4EH
DB 000H,038H,044H,044H,044H,038H,000H,000H ; “o” =4FH
DB 000H,07EH,00CH,012H,012H,00CH,000H,000H ; “p” =50H
DB 000H,00CH,012H,012H,00CH,07EH,000H,000H ; “q” =51H
DB 000H,07CH,008H,004H,004H,008H,000H,000H ; “r” =52H
DB 000H,058H,054H,054H,054H,064H,000H,000H ; “s” =53H
DB 000H,004H,03FH,044H,040H,020H,000H,000H ; “t” =54H
DB 000H,03CH,040H,040H,03CH,040H,000H,000H ; “u” =55H
DB 000H,01CH,020H,040H,020H,01CH,000H,000H ; “v” =56H
DB 000H,03CH,040H,030H,040H,03CH,000H,000H ; “w” =57H
DB 000H,044H,028H,010H,028H,044H,000H,000H ; “s” =58H
DB 000H,01CH,0A0H,0A0H,090H,07CH,000H,000H ; “y” =59H
DB 000H,044H,064H,054H,04CH,044H,000H,000H ; “z” =5AH
DB 000H,000H,008H,036H,041H,000H,000H,000H ; “{” =5BH
DB 000H,000H,000H,077H,000H,000H,000H,000H ; “|” =5CH
DB 000H,000H,041H,036H,008H,000H,000H,000H ; “}” =5DH
DB 000H,002H,001H,002H,004H,002H,000H,000H ; “-” =5FH
DB 000H,0FFH,0FFH,0FFH,0FFH,0FFH,000H,000H ; “ ” =60H

4 中文字符写入子程序

```

        COLUMN EQU 30H ; 列地址寄存器 (0-191)
        PAGE EQU 31H ; 页地址寄存器 D1, D0: 页地址
        CODE EQU 32H ; 字符代码寄存器
        COUNE EQU 33H ; 计数器
CCW_PR: MOV DPTR, #CCTAB ; 确定字符字模块首地址
        MOV A, CODE ; 取代码
        MOV B, #20H ; 字模块宽度为 32 个字节
        MUL AB ; 代码 X32
        ADD A, DPL ; 字符字模块首地址
        MOV DPL, A ; =字模库首地址+代码 X32
        MOV AB
        ADDC A, DPH
        MOV DPH, A
        PUSH COLUMN ; 列地址入栈
        PUSH COLUMN ; 列地址入栈
        MOV CODE, 00H ; 代码寄存器借用为间址寄存器
CCW_1: MOV COUNT, #10H ; 计数器设置为 16
        MOV A, PAGE ; 读页地址寄存器
        ANL A, #07H
        ORL A, 0B8H ; “或” 页地址设置代码
        MOV COM, A ; 写页地址设置指令
        LCALL PRL0
        LCALL PRM0
        LCALL PRR0
        POP COLUMN ; 取列地址值
        MOV A, COLUMN ; 读列地址寄存器
        CLR C
        SUBB A, #40H ; 列地址-64
        JC CCW_2 ; <0 为左屏显示区域
        MOV COLUMN, A
        SUBB A, #40H ; 列地址-64
        JC CCW_11 ; <0 为中屏显示区域
        MOV COLUMN, A ; ≥0 为右屏显示区域
        MOV A, PAGE
        SETB ACC.5 ; 设置区域标志位
        MOV PAGE, A ; “00” 为左, “01” 为中, “10” 为右
        LJMP CCW_2
CCW_11: MOV A, PAGE
        SETB ACC.4 ; 设置区域标志位
        MOV PAGE, A
CCW_2: MOV COM, COLUMN ; 设置列地址值
        ORL COM, #40H ; “或” 列地址指令标志位
        MOV A, PAGE ; 判区域标志以确定设置哪个控制器

```

```

        ANL      A, #30H
        CJNE     A, #10H, CCW_31 ; “01” 为中区
        LCALL    PRM0
        LJMP     CCW_4
CCW_31:  CJNE     A, #20H, CCW_32 ; “10” 为右区
        LCALL    PRR0
        LJMP     CCW_4
CCW_32:  LCALL    PRL0           ; “00” 为左区
CCW_4:   MOV      A, CODE        ; 取间址寄存器值
        MOVC     A, @A+DPTR     ; 取汉字字模数据
        MOV      DAT, A         ; 写数据
        MOV      A, PAGE        ; 判区域标志
        ANL      A, #30H
        CJNE     A, #10H, CCW_41 ; “01” 为中区
        LCALL    PRM1
        LJMP     CCW_5
CCW_41:  CJNE     A, #20H, CCW_42 ; “10” 为右区
        LCALL    PRR1
        LJMP     CCW_5
CCW_42:  LCALL    PRL1           ; “00” 为左区
CCW_5:   INC      CODE          ; 间址寄存器加 1
        INC      COLUMN        ; 列地址寄存器加 1
        MOV      A, COLUMN      ; 判列地址是否超出区域范围
        CJNE     A, #40H, CCW_6
CCW_6:   JC       CCW_7         ; 未超出则继续
        MOV      COLUMN, #00H
        MOV      A, PAGE        ; 超出则判在何区域
        JB       ACC.5, CCW_9    ; 在右区域则退出
        JB       ACC.4, CCW_61   ; 判在左或中区
        SETB     ACC.4          ; 在左区则转中区
        MOV      PAGE, A
        MOV      COM, #40H      ; 设置中区列地址为 “0”
        LCALL    PRM0
        LJMP     CCW_7
CCW_61:  SETB     ACC.5          ; 在中区则转右区
        CLR      ACC.4
        MOV      PAGE, A
        MOV      COM, #40H      ; 设置右区列地址为 “0”
        LCALL    PRR0
CCW_7:   DJNZ     COUNT, CCW_4   ; 当页循环
        MOV      A, PAGE        ; 读页地址寄存器
        JB       ACC.7, CCW_9    ; 判完成标志 D7 位, “1” 则完成退出
        INC      A              ; 否则页地址加 1

```

```

        SETB    ACC.7           ; 置完成位为“1”
        ANL     A, #0CFH        ; 清区域标志
        MOV     PAGE, A
        MOV     CODE, #10H      ; 间址寄存器设置为 16
        LJMP    CCW_1           ; 大循环
CCW_9:   RET

```

中文演示程序段

```

MOV     PAGE, #02H
MOV     COLUMN, #35H
MOV     CODE, #00H
LCALL   CCW_PR
MOV     PAGE, #02H
MOV     COLUMN, #4BH
MOV     CODE, #01H
LCALL   CCW_PR
MOV     PAGE, #02H
MOV     COLUMN, #63H
MOV     CODE, #02H
LCALL   CCW_PR
MOV     PAGE, #02H
MOV     COLUMN, #7BH
MOV     CODE, #03H
LCALL   CCW_PR
SJMP    $

```

CCTAB:

```

HZ0:   DB  08H, 0AH, 0EAH, 0AAH, 0AAH, 0AAH, 0AFH, 0E0H
        DB  0AFH, 0AAH, 0AAH, 0AAH, 0FAH, 28H, 0CH, 00H
        DB  20H, 0A0H, 0ABH, 6AH, 2AH, 3EH, 2AH, 2BH
        DB  2AH, 3EH, 2AH, 6AH, 0ABH, 0A0H, 20H, 00H      ;冀

HZ1:   DB  40H, 42H, 0CCH, 00H, 00H, 0F8H, 88H, 88H
        DB  88H, 08H, 0FFH, 08H, 0AH, 0CCH, 08H, 00H
        DB  00H, 00H, 3FH, 90H, 48H, 3FH, 08H, 10H
        DB  4FH, 20H, 13H, 1CH, 63H, 80H, 0E0H, 00H      ;诚

HZ2:   DB  00H, 0F8H, 48H, 48H, 48H, 48H, 0FFH, 48H
        DB  48H, 48H, 48H, 0FCH, 08H, 00H, 00H, 00H
        DB  00H, 07H, 02H, 02H, 02H, 02H, 3FH, 42H
        DB  42H, 42H, 42H, 47H, 40H, 70H, 00H, 00H      ;电

HZ3:   DB  80H, 80H, 82H, 82H, 82H, 82H, 82H, 0E2H
        DB  0A2H, 92H, 8AH, 86H, 80H, 0C0H, 80H, 00H

```


DB 00H, 00H, 00H, 00H, 00H, 40H, 80H, 7FH
DB 00H, 00H, 00H, 00H, 00H, 00H, 00H, 00H ;子

汉字的显示是国内应用图形液晶显示模块的目的之一。由于 KS0108 显示 RAM 的特性，所以不能将计算机内的汉字库的汉字提出直接使用，需要将其旋转 90 度后再写入。字库的提取软件可以与冀诚公司业务部门联系索要。提取汉字字模软件，将汉字从计算机内汉字库提取旋转 90 度后生成专用的用户字库。其生成字库的格式为前 16 个字节为上半部 16*8 点阵字模数据，后 16 个字节为下半部 16*8 点阵字模数据。该程序提供单字节汉字代码寄存器，所以只能建立 256 个汉字库。若要选择显示更多的汉字，就需要使用双字节汉字代码寄存器。这时只需要修改一下程序的前 8 条即可实现。

冀诚公司销售网络:

大陆工厂

地址: 河北省石家庄市新石北路 368 号 邮政编码: 050091

电话: +86-311-3856940 传真: +86-311-3851891

电子邮件: sales@gemtech-hb.com

深圳宾康实业有限公司

地址: 深圳福田区华强北路宝华大厦 A1205 室

邮政编码: 518031 电话: +86-755-83742919 传真: +86-755-83742632

冀诚电子杭州分公司

地址: 杭州市文苑路 522 号 4-2-601

电话: +86-571-88988297

香港代理

宾康有限公司

地址: 香港九龙湾宏照道 11 号 宝隆中心 B 座 3 字楼 12 室

电话: +852-27518691 传真: +852-27965670

电子邮件: pantage@asiansources.com

新加坡代理

大欣工业私人有限公司

地址: 41 Kallang Pudding Road #06-06 Golden Wheel Building Singapore 349316s

电话: +65-7463198 传真: +65-7468348

电子邮件: tassinsingnet.com.sg