模组用户手册

我

LED 键盘模组

使用说明书

V1.0 – 2004.8..20

凌阳大学计划推广中心
北京海淀上地信息产业基地中黎科技园 1 号楼 6 层
TEL: 86-10-62981668 FAX: 86-10-62985972 E-mail:unsp@sunplus.com.cn http://www:unsp.com.cn

版权声明

目　　录

1 前言

现在单片机在各个领域都有广泛的应用，在单片机系统应用中我们经常需要将实时测得的信息输入到单片机，经过处理后送到输出设备以便显示或输出到执行机构实行对现场的控制等。基本一般的单片机系统中都扩展了按键 KEY 和显示单元（LED、数码管、LCD 等）。

我们这本书中介绍了基于单片机常用外围器件的扩展而设计的 LED 键盘模组，它集成了必要的 LED、KEY、数码管功能，可与任一款单片机进行软硬件接口设计。书中详细介绍了此模组的主要功能、接口说明以及使用方法等，并对模组中的元器件进行了介绍。

此模块简单实用，使用时用排线连接即可，配合 61 板使用最佳。我们针对 61 板和此模组也举出了应用实例和该模组的自检程序。

通过对该模组的使用，我们会掌握单片机常用外围器件如键盘、LED 和数码管的使用方法等，对单片机应用有了基础的了解，达到节省时间、快速入门的目的。

2 系统简介

2.1 基本特性与参数指标

LED 键盘模组集成 LED、KEY、数码管功能，可作为单片机常用外围器件的扩展模块。

LED 键盘模组采用 DC5V 供电。

2.2 主要功能

1. 扩展了 6 位 8 段数码管，最大显示数据为 999999；
2. 8 个发光二极管，可作为显示状态信息使用；
3. 8 个按键，可以组成 1*8KEY 也可组成 2*4KEY；
4. 一个电位器，可以提供 0－5V 的模拟电压信号或者 0—3.3V 的模拟电压,与模组输入的 VDD 有关。
5. 键盘 LED 模组接口简单，可方便与任何一款单片机进行软硬件接口设计。

2.3 结构框图

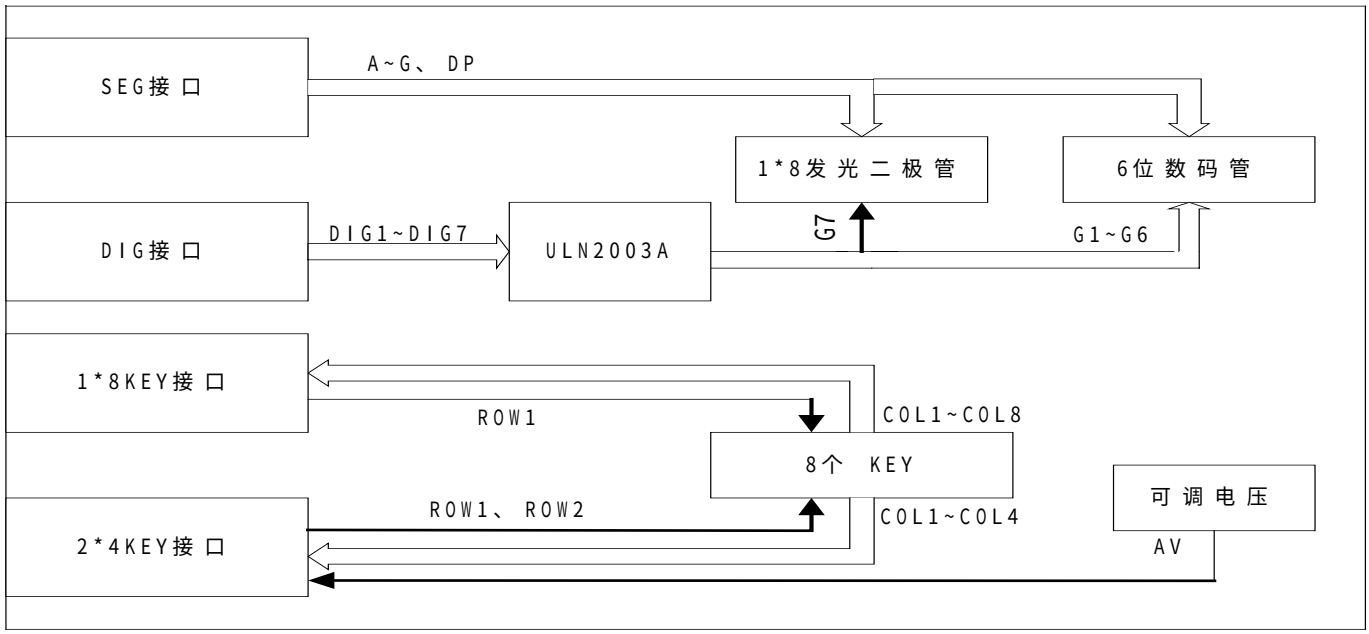


图 1　LED 键盘模组结构框图

2.4 布局框图

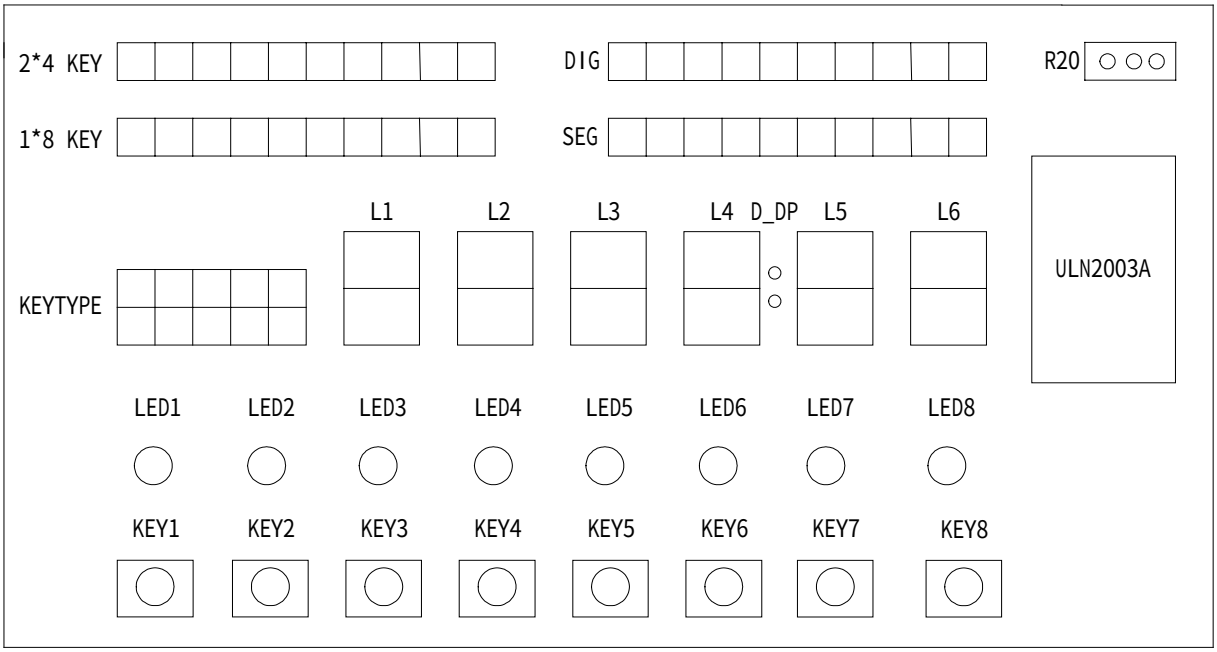


图 2 LED 键盘模组的布局框图

2.5 系统环境

LED 键盘模组供电电压 DC5V，也可用 DC3.3V。

2.6 注意事项

LED 键盘模组可作为任何一款具有外部 IO 端口的单片机进行外部扩展使用。

3 硬件说明

3.1 电路原理图

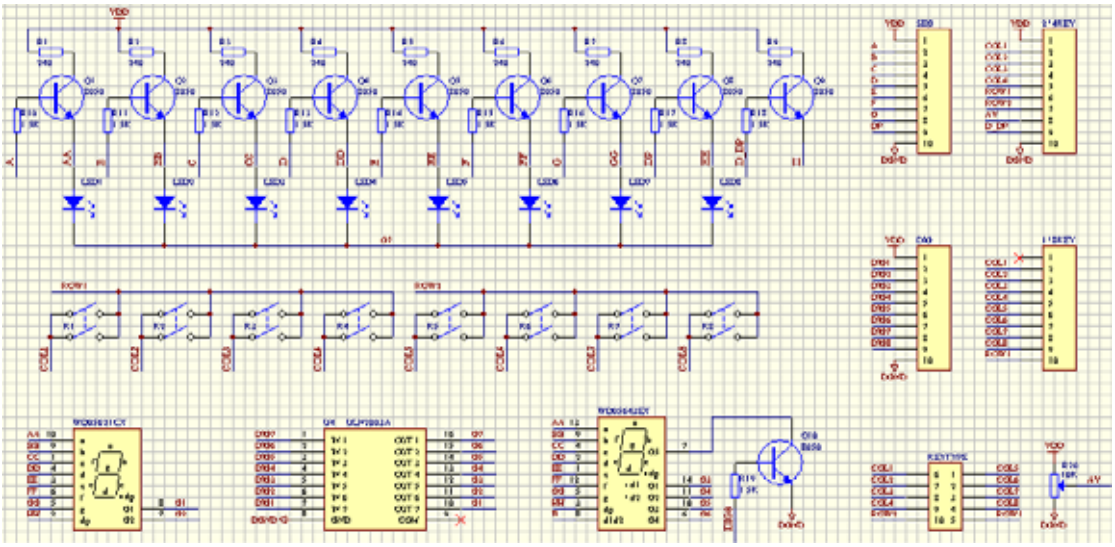


图 3 LED 键盘模组的电路原理图

3.2 主要元器件

1. ULN2003A

其内部为三极管阵列，其 IN 脚相当于三极管的 B 极，OUT 较相当于三极管的 C 极。若 IN 脚输入高电平，对应的 OUT 脚接地；IN 脚输入低电平，对应的 OUT 脚截止输出。

ULN2003A 元件图如下：

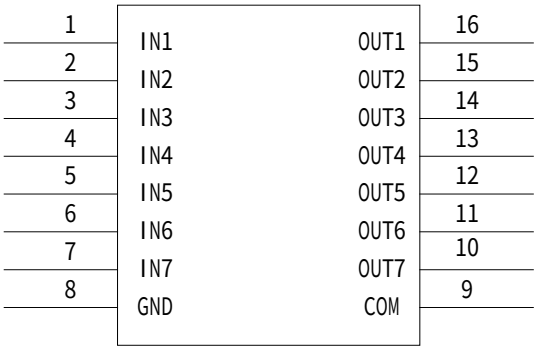


图 4 ULN2003A 元件图

图中：IN1~IN7 为输入信号，OUT1~OUT7 为输出信号。输入信号高有效。

2. LG5621AH

共阴极 2 位数码管。

LG5621AH 元件图如下：

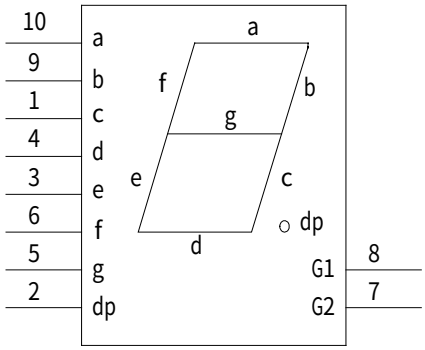


图 5　LG5621AH 元件图

图中：a~dp 为数码管的段信号，G1、G2 为 2 位数码管的位信号。段信号高有效，位信号低有效。

3. LG5641AH

共阴极 4 位数码管。

LG5641AH 元件图如下：

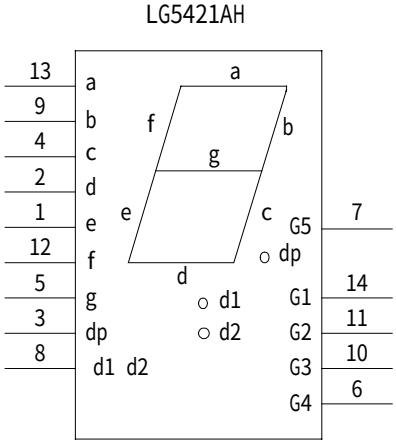


图 6　LG5641AH 元件图

图中：a~dp 为数码管的段信号，d1、d2 为时钟冒号的段信号；G1~G4 为 4 位数码管的位信号，G5 为时钟冒号的位信号。段信号高有效，位信号低有效。

3.3　接口说明

接口说明：

LED 模组使用十分简单，直接用排线与 61 板 I/O 接口相连即可。

1. KEYTYPE 选择接口

此接口是选择 2*4KEY 还是 1*8KEY，具体选择方式如下：

从左至右看，将第 1 至 4 短接块连接，选择了 2*4KEY；只将第 5 个短接块短接，选择 1*8KEY。

选择 2*4KEY 接口连接示意图如下：

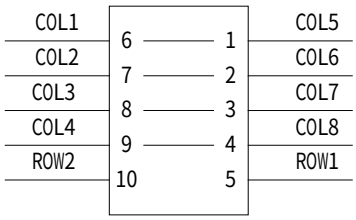


图 7　选择 2*4KEY 时 KEYTYPE 接口连接示意图

选择 1*8KEY 接口连接示意图如下：

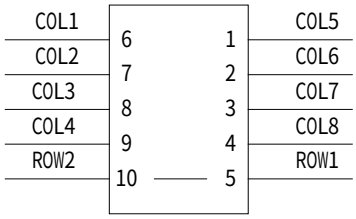


图 8　选择 1*8KEY 时 KEYTYPE 接口连接示意图

2. 2*4KEY 接口

2*4KEY 的第 1 组行定义为 ROW1，第 2 组行定义为 ROW2，4 列定义为 COL1~COL4。

使用时 COL1 是 K1 和 K5 的输入，COL2 是 K2 和 K6 的输入，COL3 是 K3 和 K7 的输入，COL4 是 K4 和 K8 的输入。用户可自行选择 ROW1 与 ROW2 接至 VDD 还是 GND，但同一时间只能使用一组。AV 是模拟电压输出端，通过调整 R20 可以改变 AV 的值，AV 的最大输出值与 VDD 相同。D_DP 是第 4 位数码管后时钟冒号的位信号控制端，见图 2 中的 D_DP。

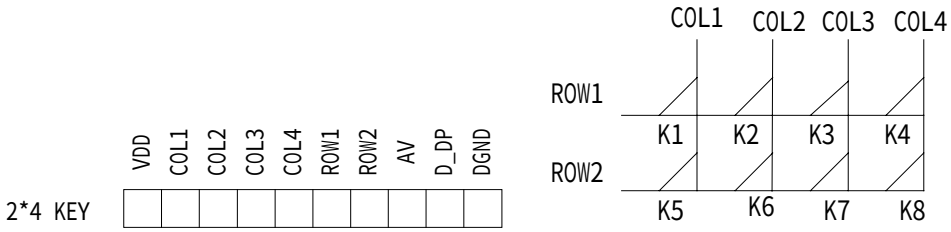


图 9　2*4KEY 接口图及使用示意图

使用 2*4KEY 时相应 KEYTYPE 连接如图 7，即将 KEYTYPE 接口的前 4 个短接块短接。

3. 1*8KEY 接口

1*8KEY 的 8 列分别定义为 COL1~COL8，1 行定义为 ROW1。使用时 COL1~COL8 是 K1~K8 的列输入，ROW1 是 K1~K8 的行输入。1*8KEY 接口见图 10。

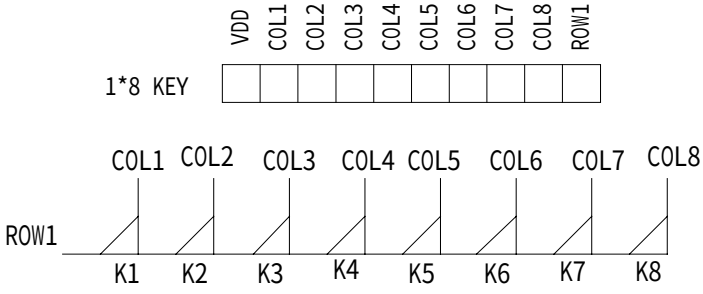


图 10　1*8KEY 接口图及使用示意图

使用 1*8KEY 时相应 KEYTYPE 连接如图 8，即只须将 KEYTYPE 接口的第 5 个短接块短接。用户可自行选择 ROW1 接至 VDD 还是 GND。

4. DIG 接口

6 位数码管 L1~L6 的段发光管阳极和 8 个 LED 指示灯的阳极并联，并且 8 个 LED 指示灯采用共阴极方式。6 位数码管 L1~L6 的阴极和 8 个 LED 的共阴极分别用 DIG 接口的 DIG1~DIG7 控制，第 4 位数码管后时钟冒号 D_DP（见图 2）的位信号用 DIG8 控制，位信号均为高有效。DIG 接口详见图 11。

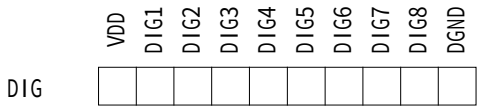


图 11　DIG 接口图

5. SEG 接口

A~G、DP 是 6 位数码管 L1~L6 的段信号和 8 个 LED 指示灯的阳极控制信号输入端。控制信号为高有效。

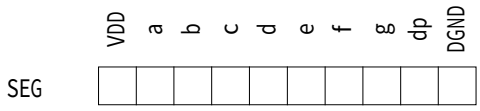
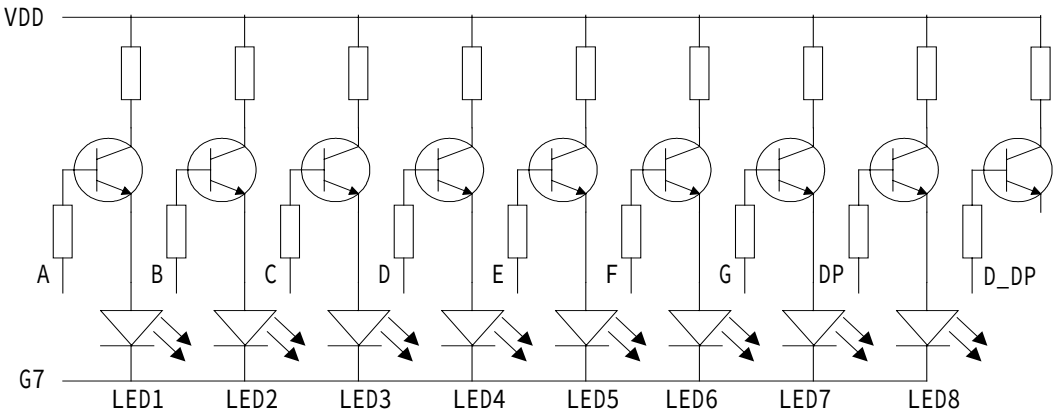


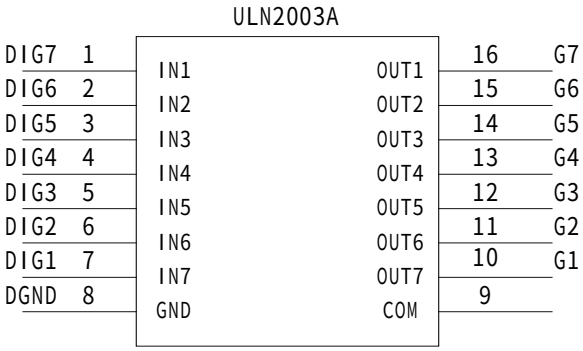
图 12　SEG 接口图

实验前请仔细阅读附录，要了解数码管和LED的结构，在用排线连接系统开发板和模组进行实验时一定要注意方向问题：板子的V_{CC}与模组的VDD是同一点。

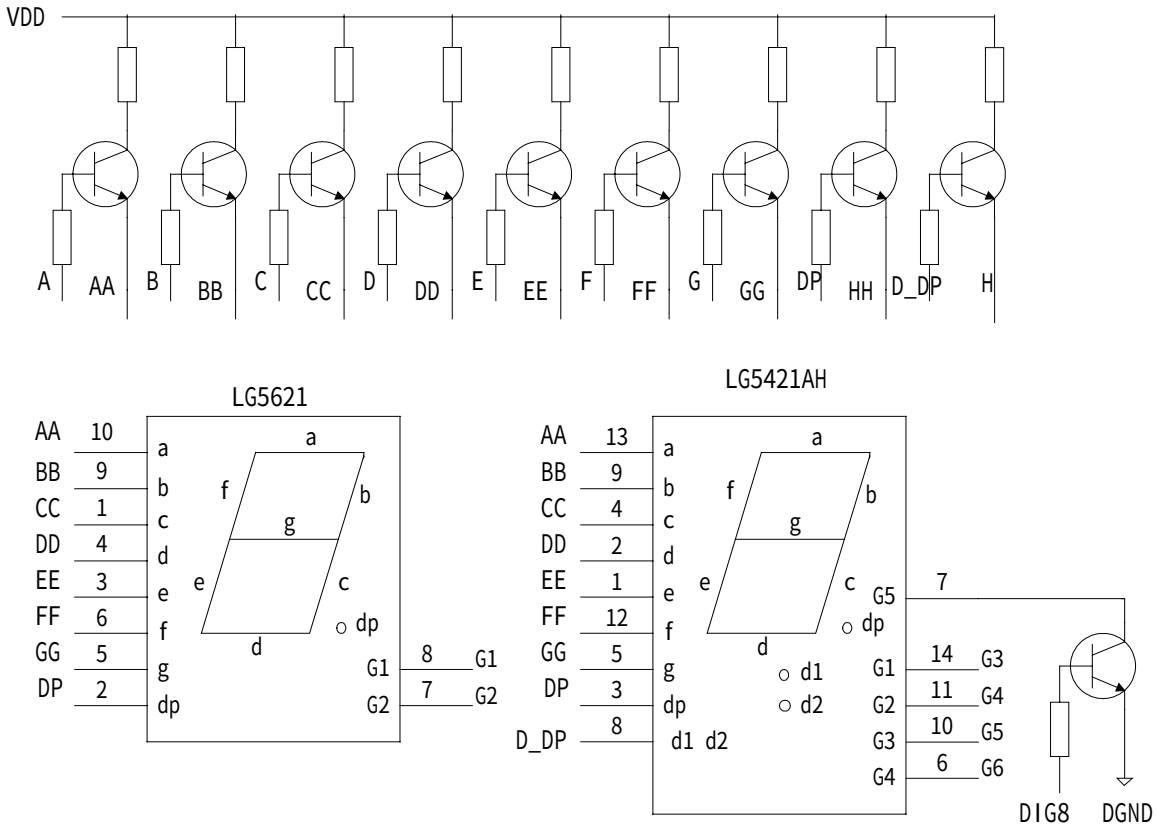
使用提示：

LED 的使用说明：LED 的使用主要涉及 DIG 接口（见图 11）和 SEG 接口（见图 12）。8 个 LED（ LED1~LED8）采用共阴极设计共阴极即位信号用 DIG 接口中的 DIG7 控制，阳极即段信号由 SEG 接口的 A~G、DP 控制。使用时首先将其位信号 DIG7 设置成有效状态“ 高 ”，即选择了该元件，然后将其段信号 A~G、DP 设置成需要的状态，即选择了显示信息，段信号“ 高 ”有效。相应原理见下图。





数码管的使用说明：数码管的使用主要涉及 DIG 接口（见图 11）和 SEG 接口（见图 12）。数码管 L1~L6 采用共阴极设计共阴极即位信号用 DIG 接口中的 DIG~DIG6 控制，阳极即段信号由 SEG 接口的 A~G、DP 控制。使用时首先将其位信号 DIG~DIG6 设置成有效状态“ 高 ”，即选择了相应数码管，然后将其段信号 A~G、DP 设置成需要的状态，即选择了显示信息，段信号 “ 高 ” 有效。相应原理见下图。



时钟冒号 D_DP（见图 2）的使用说明：D_DP 的位信号是 DIG 接口（见图 11）中的 DIG8，段信号是 2*4KEY 接口（见图 9）中的 D_DP。想点亮此时钟冒号只需将 DIG8 与 D_DP 都设置成 “ 高 ” 即可。原理见上图。

按键的使用说明：

1*8 按键使用举例：见图 10，使用按键功能时需检测获取按键信息的相应 IO 端口状态。例如，将 1*8KEY 接口中的控制 ROW1 的 IO 口设置成低电平输出，然后获取按键信息 COL1~COL8 的 IO 端口设置成高电平输入口，当有键按下，相应的 IO 口将得到一个低电平， 通过得到的低电平即可判断是哪个按键按下。

2*4 按键使用举例：见图 9，使用按键功能时需检测获取按键信息的相应 IO 端口状态。例如，将 2*4KEY

接口中的控制 ROW1 的 IO 口设置成低电平输出，控制 ROW2 的 IO 口设置成高电平输出，然后获取按键信息 COL1~COL4 的 IO 端口设置成高电平输入口，当有键按下，相应的 IO 口将得到一个低电平， 通过得到的低电平即可判断是哪个按键按下。

4 应用举例

以下范例以 SPCE061A 指令集编写，可与凌阳 16 位单片机 SPCE061A 实验仪、35 板、61 板直接连接使用。

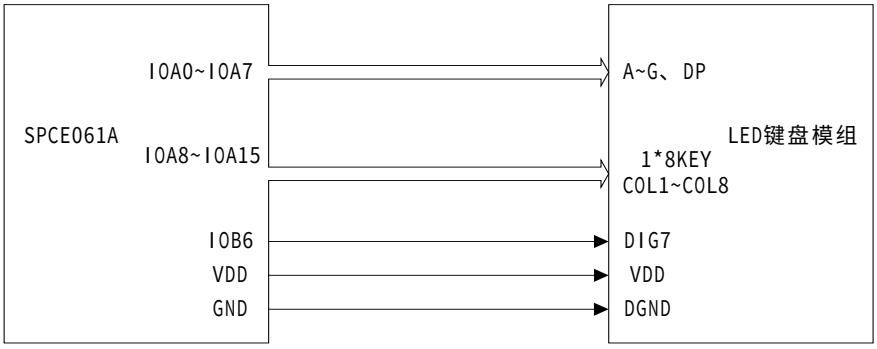
此范例是 LED 键盘模组与 61 板配套使用。

程序范例：

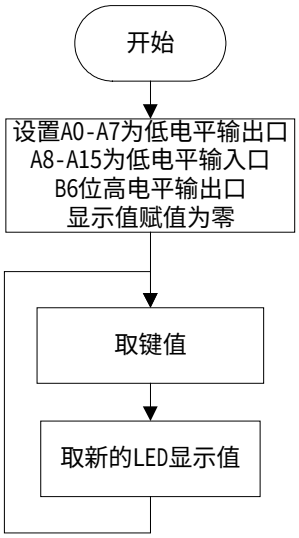
此范例实现的功能是：按键点亮发光二极管。主程序调用键盘扫描子程序 GetKey ()，即可读取按键。主程序根据按键值修改发光二极管显示值，然后赋给相应端口，即可完成发光二极管显示。

硬件连接是：IOA 低 8 位接至 SEG 接口控制 LED 的导通，IOA 高 8 位连接到 LED 键盘模组的 1*8KEY 接口管脚上，读取相应按键值， 1*8key 中 ROW1 与 DGND 相连，IOB6 连至 DIG7 通过 ULN2003A 控制 8 个 LED 的共阴极电平状态。

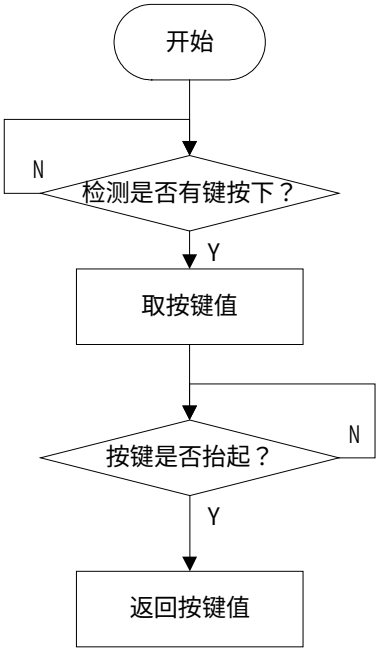
硬件连接图如下：



主程序流程图如下：



取按键值程序流程图如下：



程序由两部分组成，主程序和取按键值程序。程序代码如下：

主程序代码：

```
#define P_IOA_Data          (volatile unsigned int *)0x7000
#define P_IOA_Buffer        (volatile unsigned int *)0x7001
#define P_IOA_Dir           (volatile unsigned int *)0x7002
#define P_IOA_Attrib        (volatile unsigned int *)0x7003
#define P_IOB_Data          (volatile unsigned int *)0x7005
#define P_IOB_Buffer        (volatile unsigned int *)0x7006
#define P_IOB_Dir           (volatile unsigned int *)0x7007
#define P_IOB_Attrib        (volatile unsigned int *)0x7008
#define P_Watchdog_Clear    (volatile unsigned int *)0x7012
extern unsigned GetKey(void);
//=============================================================
// 函数名称:      int main()
// 日期：          20040816
// 功能描述:      按键点亮发光二极管
// 语法格式:      int main()
// 入口参数:      无
// 出口参数:      无
// 注意事项:      仅为用户模型
//=============================================================
```

```
int main()
{
    *P_IOA_Dir = 0x00ff;            //设置 A 口低 8 位为同向低输出，高 8 位为同向上拉输入
    *P_IOA_Attrib = 0x00ff;
    *P_IOA_Data = 0xff00;
     *P_IOB_Dir=0x0040;              //设置 B6 口为高电平输出，保证 LED 共阴极接地
    *P_IOB_Attrib=0x0040;
    *P_IOB_Data=0x0040;

    while(1)
    {
        Key = GetKey();             //取键值
        Key = Key >> 8;             //取 LED 显示初值
        *P_IOA_Data = Key;
     }
}
```

取按键程序代码：

```
#define P_IOA_Data          (volatile unsigned int *)0x7000
#define P_IOA_Buffer        (volatile unsigned int *)0x7001
#define P_IOA_Dir           (volatile unsigned int *)0x7002
#define P_IOA_Attrib        (volatile unsigned int *)0x7003
#define P_Watchdog_Clear    (volatile unsigned int *)0x7012
#define KEY_ALL 0xff00          //使用 IOA8~IOA15 作为键盘输入口
//============================================================
//  函数名称:    GetKey()
//  功能描述:    等待直到有键按下并抬起，返回键值，没有去抖处理
//  输入：       无
//  输出：       返回 16 位键值
//============================================================
unsigned GetKey(void)
{
```

```
    unsigned KeyValue;

    //初始化 IOA 的相应端口为上拉输入
    *P_IOA_Dir&=~KEY_ALL;
    *P_IOA_Attrib&=~KEY_ALL;
    *P_IOA_Buffer|=KEY_ALL;

    //等待有键按下，即有端口变为 0
    while(!((*P_IOA_Data&KEY_ALL)^KEY_ALL))
    {
        *P_Watchdog_Clear=1;        //喂狗
    }
    KeyValue=(*P_IOA_Data&KEY_ALL)^KEY_ALL;
    //等待按键抬起
    while((*P_IOA_Data&KEY_ALL)^KEY_ALL)
    {
        *P_Watchdog_Clear=1;
    }
    return KeyValue;
}
```

5 自检操作

用户在使用 LED 键盘模组之前可自行运行自检程序，检查 LED 键盘模组的功能是否正常。

5.1 自检前的准备

软件准备：

下载自检代码。

硬件准备：

1. 61 板及 61 板配套电源；
2. LED 键盘模组；
3. 自检时 61 板和 LED 键盘模组硬件连接情况是：

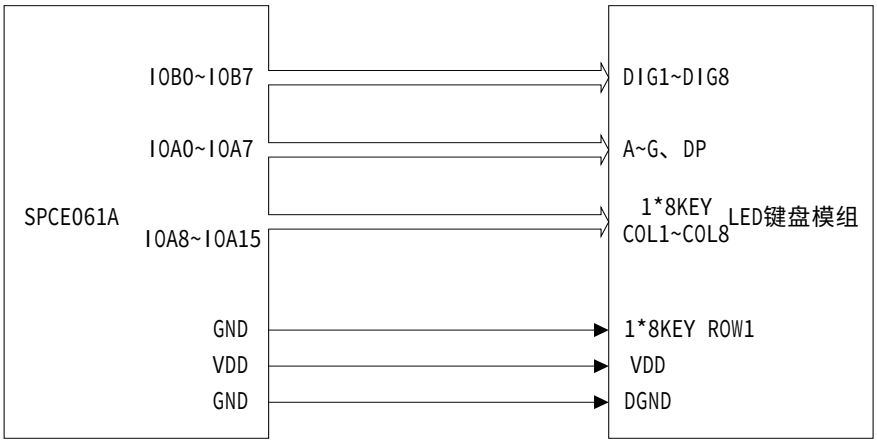
   IOA 低 8 位接至 SEG 接口控制 LED 的导通,IOA 高 8 位接至 1*8 接口获取键盘的状态值，

   IOB 低 8 位连至 DIG 通过 ULN2003A 控制 8 个 LED 的共阴极电平状态和数码管驱动，

   1*8KEY 中 ROW1 直接连至 DGND。

   注意：KEYTYPE 接口选择 1*8KEY，见图 8。

4. 硬件连接框图

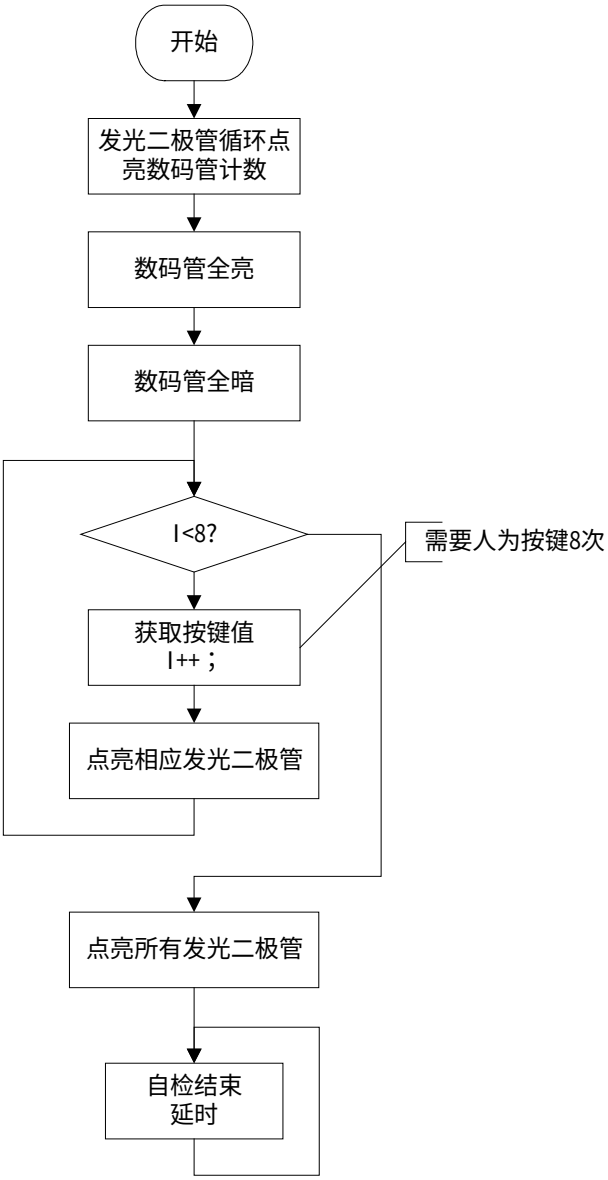


5.2 自检操作流程

确认硬件连接就绪后，使用 IDE 下载自检程序，运行自检程序。自检过程遵循下图所示流程完成。

自检无错后，说明 LED 键盘模组硬件无故障，您就可以利用其进行软硬件开发了。

自检过程描述如下：运行自检程序后，发光二极管循回点亮并数码管计数（LED 巡回闪烁，同时数码管从左至右显示 0~8），全部显示完成后，LED 停止闪烁，数码管全亮，延时一段时间后数码管全灭，然后人为按 8 个按键，顺序自选，按键相对应的数码管将点亮，8 次检测结束后，二极管全亮，自检完成。以上过程完成说明 LED 模组功能完全正常。



自检程序代码附录如下：

```
#define P_IOA_Data          (volatile unsigned int *)0x7000
#define P_IOA_Buffer        (volatile unsigned int *)0x7001
#define P_IOA_Dir           (volatile unsigned int *)0x7002
#define P_IOA_Attrib        (volatile unsigned int *)0x7003
#define P_IOB_Data          (volatile unsigned int *)0x7005
#define P_IOB_Buffer        (volatile unsigned int *)0x7006
#define P_IOB_Dir           (volatile unsigned int *)0x7007
#define P_IOB_Attrib        (volatile unsigned int *)0x7008
#define P_Watchdog_Clear    (volatile unsigned int *)0x7012
#define KEY_ALL 0xff00
```

```
int Led_Disp(int Led, int Loop);
unsigned GetKey(void);
                                                          //显示段码存放区
const    int DispTbl[10] = { 0x003F,0x0006,0x005B,0x004F,0x0066,       //0，1，2，3，4
                        0x006D,0x007D,0x0007,0x007F,0x00FF};//5，6，7，8，全亮

unsigned LedControl = 0x0001;
//=============================================================
// 函数名称:      int main()
// 功能描述:      LED 键盘模组自检程序
// 注意事项:      仅为用户模型
//=============================================================
int main()
{
        int Loop_Numb = 0x0000;          //循环次数
        int Led_Numb = 0x0000;           //显示数字个数
        unsigned Key = 0x0000;
        int i = 0;

       *P_IOA_Dir = 0x00ff;              //设置 A 口低 8 位为同向低输出，控制 LED 和数码管的显示状态
        *P_IOA_Attrib = 0x00ff;          //设置 A 口高 8 位为同向高输入，获取按键的状态值
        *P_IOA_Data = 0xff00;
       *P_IOB_Dir = 0x00ff;              //设置 B0~B7 口为同相低电平输出，LED 和数码管的片选
        *P_IOB_Attrib = 0x00ff;
        *P_IOB_Data = 0x0000;
        for (Loop_Numb = 0; Loop_Numb < 6; Loop_Numb++)
        {
                *P_IOB_Data = LedControl << Loop_Numb; //对应数码管显示 0
                *P_IOA_Data = DispTbl[0];
                Delay();

                for (Led_Numb = 0; Led_Numb<8; Led_Numb++)
```

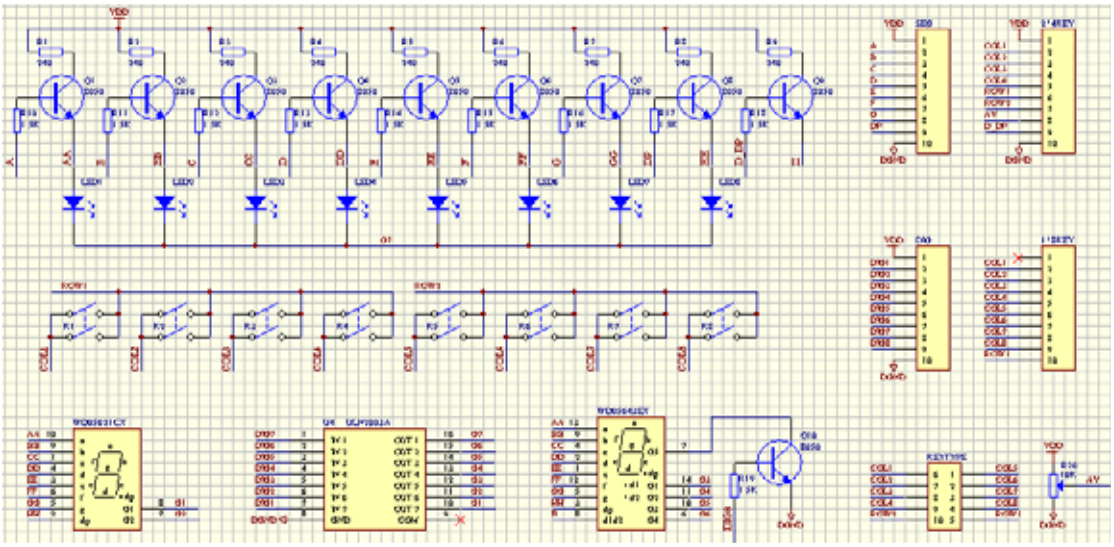
```
                {
                    Led_Disp(Led_Numb, Loop_Numb);        //点亮发光二极管并显示数字 1~8
                }
            }
            *P_IOB_Data = 0x00bf;                          //数码管全亮
            *P_IOA_Data = DispTbl[9];
             Delay();
             *P_IOB_Data = 0x00bf;                         //数码管全灭
             *P_IOA_Data = 0x0000;
            Delay();
           *P_IOB_Data = 0x0040;
            for (i = 0; i < 8; i++)                        //按键点亮发光二极管
            {
                 Key = GetKey();                           //取键值
                 Key = Key >> 8;                           //取发光二极管显示值
                *P_IOA_Data = Key;                         //发光二极管点亮并延时
                Delay();
             }
            Delay();
            *P_IOA_Data = 0x00ff;                          //发光二极管全亮
            while(1)
                 Delay();
}
//============================================================
// 函数名称:      void Delay()
// 功能描述:      延时并清看门狗
//============================================================
int Delay()
{
    int DelayValue = 0;
    for (DelayValue = 0; DelayValue < 0x2000; DelayValue++)
         *P_Watchdog_Clear = 1;
```

```
}
//=============================================================
// 函数名称:      void Led_Disp()
// 功能描述:      发光二极管循回点亮并数码管计数
// 入口参数:      Led 点亮的发光二极管号码，Loop 显示对应数字的数码管号码
//=============================================================
int Led_Disp(int Led, int Loop)
{
        *P_IOB_Data = 0x0040;                   //LED 点亮
        *P_IOA_Data = LedControl << Led;
        Delay();
        *P_IOB_Data = LedControl << Loop;     //对应数字显示在数码管上
        *P_IOA_Data = DispTbl[Led + 1];
        Delay();
}
//=============================================================
//        函数名称:     GetKey()
//        功能描述:     等待直到有键按下并抬起，返回键值，没有去抖处理
//        输入：        无
//        输出：        返回 16 位键值
//=============================================================
unsigned GetKey(void)               //使用 IOA8~IOA15 作为键盘输入口
{
        unsigned KeyValue;

        //初始化 IOA 的相应端口为上拉输入
        *P_IOA_Dir&=~KEY_ALL;
        *P_IOA_Attrib&=~KEY_ALL;
        *P_IOA_Buffer|=KEY_ALL;

        //等待有键按下，即有端口变为 0
        while(KeyValue==0)
```

```
        {
         KeyValue=(*P_IOA_Data & KEY_ALL)^KEY_ALL;
         *P_Watchdog_Clear=1;
        }
        KeyValue=(*P_IOA_Data&KEY_ALL)^KEY_ALL;
        //等待按键抬起
        while((*P_IOA_Data&KEY_ALL)^KEY_ALL)
        {
         *P_Watchdog_Clear=1;
        }
        return KeyValue;
}
```

6 附录

6.1 电路原理图



6.2 实物图

6.3 配件清单

LED 键盘模组配件包括：LED 键盘模组板，10 针排线 2 个。

6.4 元件清单

器件分类	器件标号	元件内容	数量
电阻	R1~R9	240Ω	9 个
	R10~R19	1.5KΩ	10 个
电位器	R20	10KΩ	1 个
三极管	Q1~Q10	8050	10 个
发光二极管	LED1~LED8	LED（绿）	8 个
按键	K1~K8	按键	8 个

数码管	LG5621AH	2 位数码管	1 个
	LG5641AH	4 位数码管	1 个
三极管阵列	U4	ULN2003A	1 个
插座	SEG 、 2*4KEY 、 DIG 、 1*8KEY	10pin 单排针	4 个
	KEYTYPE	10pin 双排针	1 个