第 2-19 讲：4×4 矩阵按键识别

1. 学习目的

1. 了解独立按键和矩阵按键的区别，掌握矩阵按键线反转法识别原理。
2. 掌握 STC8A8K64D4 系列单片机使用线反转法识别 4×4 矩阵按键的程序设计。

2. 矩阵按键原理

单片机设计中，按键作为一种常用的人机接口被广泛应用，在家用、娱乐、工控等设备上都可以见到按键的应用。我们最常用的两种按键接入方式是独立按键和矩阵按键。

2.1. 独立按键和矩阵按键

独立按键如下图所示，每个按键占用单片机的一个 GPIO，单片机通过读取 GPIO 的状态即可判断按键是否按下，这种方式编程简单，适合在按键较少的场合下使用。

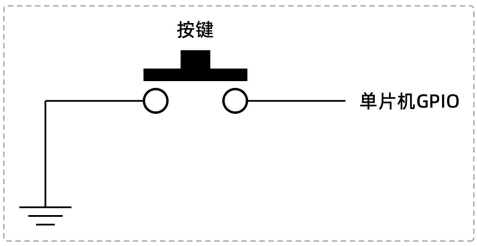


图 1：独立按键

当按键较多时，使用独立按键会占用单片机过多的 IO 资源，这会降低单片机 IO 的利用率，不利于单片机的选型（我们必须选择具有大量 IO 资源的单片机才能完成独立按键的接入），同时，也使得硬件布线变得复杂。

为了解决这个问题，当按键较多时，我们通常会使用矩阵按键方式接入。矩阵按键接入是利用单片机的 GPIO 口组成行与列，在行与列的每一个交点处连接按键，该接入方式最大优势是提高了 IO 的利用率。根据行和列的不同可以有很多种矩阵按键组合，如 3x3 矩阵按键、4×4 矩阵按键等等。其中，典型的是 4×4 矩阵按键，本章我们以 4×4 矩阵按键为例来说明矩阵按键扫描原理和编程，下图是常用的两种 4×4 矩阵按键模块。

图 2：4×4 矩阵按键

2.2. 矩阵按键识别原理

下图是 4×4 矩阵按键的原理示意图，可以看到每个按键都有对应的行号和列号，通过行号和列号的组合就可以确定是哪个按键，如行 1 列 2 就可以确定按键 S2。由此，我们自然想到，如果行和列都用数字信号表示（0 和 1），那么每个按键都可以通过一个数字组合来表示，这个数字组合就是按键对应的编码。

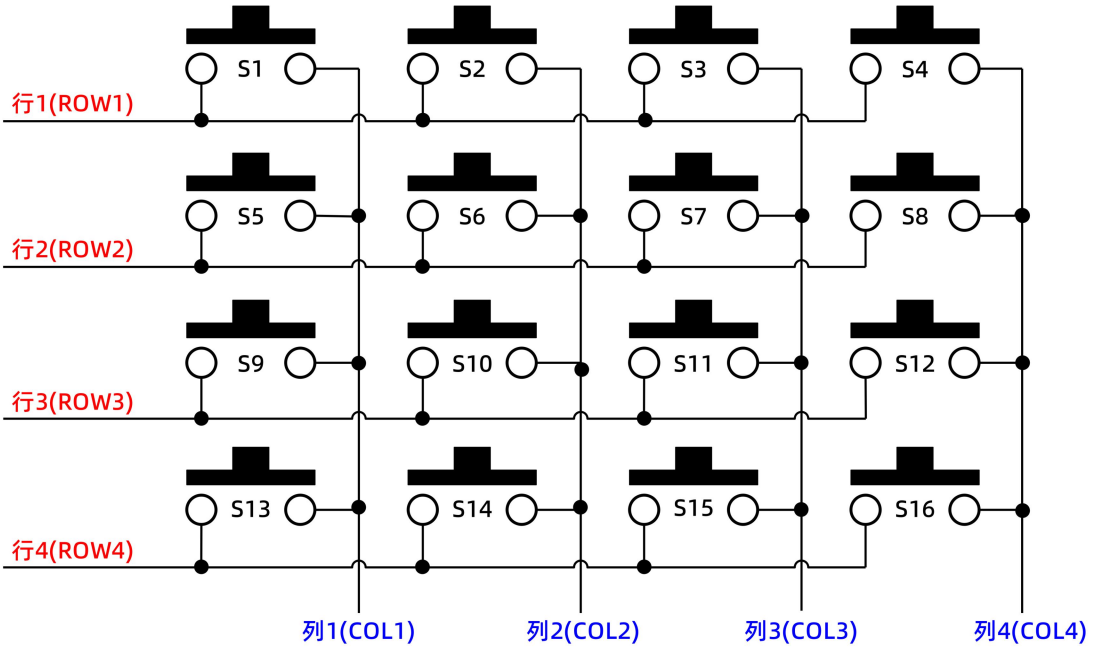


图 3：4×4 矩阵按键原理示意图

矩阵按键的识别通常需要两步操作，第一步是检测键盘上是否有键按下，第二步是识别具体是哪一个按键按下。对于矩阵按键的识别，常用的方法有行列扫描法和线反转法，其中，线反转法只需 2 步即可完成按键识别，速度快于行列扫描法，接下来，我们以线反转法为例，具体看一下矩阵按键的检测过程。

1) 行线（ROW1~ ROW3）配置为输出、列线（COL1~ COL4）配置为输入。行线输出低电平，读取列线数据，如果列线数据全为“1”，则无按键按下，如果有按键按下，则对应列的输入为 0。如 S1 键按下，可以得到列编码为 1110。
2) 线翻转，即行线配置为输入、列线配置为输出。列线输出低电平，读取行线数据，由此获取按下的按键对应的行号。
3) 由行线和列线即可得到具体是哪个按键按下。

3. 硬件电路设计

STC8A8K64D4 开发板上设计了 4×4 矩阵按键电路，如下图所示。

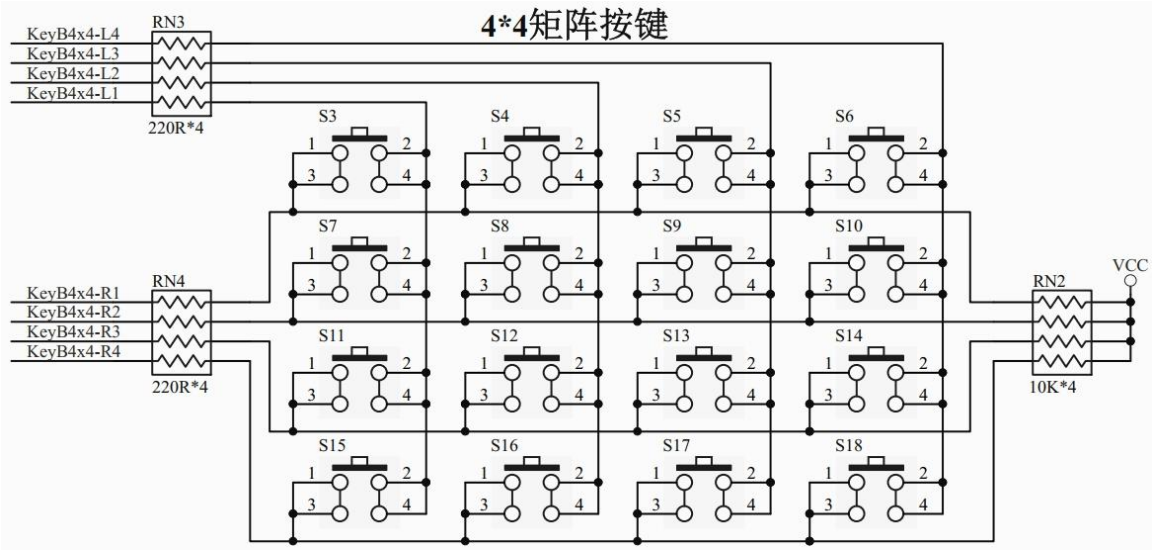


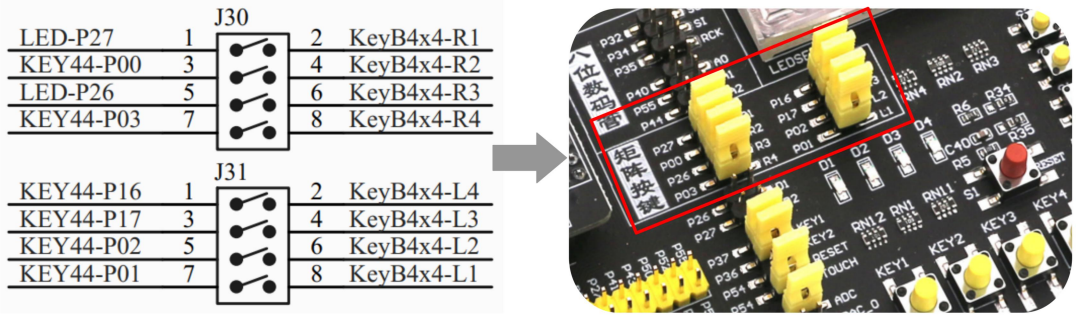
图 4：4×4 矩阵按键电路原理图

上图 4×4 矩阵按键电路中的行线和列线上串联了 220Ω 电阻，其作用为：

1) 对单片机 GPIO 提供保护：我们在调试时，如果误将行线输出高电平，列线输出低电平或者列线输出高电平，行线输出低电平，此时，若按下了某个按键，会让输出高电平的引脚直接连接到了输出低电平的引脚，导致电流过大，增加的电阻可以起到限流的作用，避免损伤 GPIO。

2) 降低按键抖动的峰值电压：按键操作时会产生抖动电压，串接的电阻可有效降低抖动电压峰值。

开发板上，为了复用单片机 IO，单片机和矩阵按键电路之间通过跳线连接，当我们使用矩阵按键时，需要将跳线短接，如下图所示。



矩阵按键电路和指示灯D1、D2以及外部晶振复用GPIO。使用矩阵按键时，需要将矩阵按键的跳线帽短接上，同时断开指示灯D1、D2的跳线帽，并且开发板不能使用外部晶振

图 5：矩阵按键引脚复用

矩阵按键使用的单片机的引脚如下表：

表 1：STC8A8K64D4 单片机矩阵按键引脚分配

名称	编号	引脚	说明

行	行 1	P2.7	和 LED 指示灯 D2 共用 IO
	行 2	P0.0	和 LED 指示灯 D2 共用 IO
	行 3	P2.6	和 LED 指示灯 D1 共用 IO
	行 4	P0.3	独立 IO
列	列 1	P0.1	独立 IO
	列 2	P0.2	独立 IO
	列 3	P1.7	和外部晶振共用 IO
	列 4	P1.6	和外部晶振共用 IO

✧ 注：独立 GPIO 表示开发板没有其他的电路使用这个 GPIO，非独立 GPIO 说明开发板有其他电路用到了该 GPIO。读者在使用非独立 GPIO 使用时需要注意电路的连接，以避免多个电路使用了同一个 GPIO。

4. 软件设计

4.1. 4×4 矩阵按键识别（串口发送键值）实验

✧ 注：本节的实验是在“实验 2-6-1：串口 1 数据收发实验”的基础上修改，本节对应的实验源码是：“实验 2-19-1：4×4 矩阵按键识别（串口发送键值）”。

4.1.1. 实验内容

用线反转方式识别矩阵按键，成功识别后，通过串口输出按键的键值（S3~S18）。

4.1.2. 代码编写

1. 新建一个名称为“matrix_keyboard_4x4.c”的文件及其头文件“matrix_keyboard_4x4.h”并保存到工程的“Source”文件夹，并将“matrix_keyboard_4x4.c”加入到 Keil 工程中的“SOURCE”组。
2. 引用头文件

因为在“matrix_keyboard_4x4.c”文件中使用了“matrix_keyboard_4x4.c”文件中的函数，所以需要引用下面的头文件“matrix_keyboard_4x4.h”。

代码清单：引用头文件

```
1.  //引用矩阵按键的头文件
2.  #include   "matrix_keyboard_4x4.h"
```

3. 4×4 矩阵按键识别

线反转法识别矩阵按键代码清单如下，因为开发板要考虑综合资源分配，使用的 GPIO 不连续（如行线使用的是 P2.7 P0.0 P2.6 P0.3，而不是连续编号的 GPIO，如 P2.0~P2.3），所以，IO 配置相对麻烦一些，读者在自己的设计中使用矩阵按键时，如果条件允许，应尽量使用连续编号的 GPIO。

代码清单：4×4 矩阵按键识别

```
1.  /*********************************************************************************
```

```
2.  功能描述：4x4 矩阵按键识别
3.  参    数：无
4.  返 回 值：按键编号 1~16
5.  ***************************************************************************/
6.  u8 KeyScan(void)
7.  {
8.      u8 row,col,key_val = 0xFF;
9.      u8 tmp;
10.
11.     //配置行为准双向
12.     P0M1 &= ~0x09; P0M0 &= ~0x09;     //设置 P0.0 和 P0.3 为准双向口
13.     P2M1 &= ~0xC0; P2M0 &= ~0xC0;     //设置 P2.6 和 P2.7 为准双向口
14.     //配置列为输入并开启上拉
15.     P0M1 |= 0x06;  P0M0 &= ~0x06;     //设置 P0.1 和 P0.2 为高阻输入
16.     P1M1 |= 0xC0;  P1M0 &= ~0xC0;     //设置 P1.6~P1.7 为高阻输入
17.     P_SW2 |= 0x80;                    //将 EAXFR 位置 1，允许访问扩展 RAM 区特殊功能寄存器(XFR)
18.     P0PU |= 0x06;                     //开启 P0.1、P0.2 的上拉电阻
19.     P1PU |= 0xC0;                     //开启 P1.6、P1.7 的上拉电阻
20.     P_SW2 &= 0x7F;                    //将 EAXFR 位置 0，禁止访问 XFR
21.     ROW1=0;ROW2=0;ROW3=0;ROW4=0;      //行输出低电平
22.     delay_ms(5);
23.
24.     tmp = COL_DATA;
25.     if(tmp != 0x0F)      //有键按下
26.     {
27.        delay_ms(20);   //软件消抖
28.        if(tmp != 0x0F) //确实有键按下
29.        {
30.           switch(tmp)
31.           {
32.              case 0x0E:col = 1;break;  //第 1 列有按键按下
33.              case 0x0D:col = 2;break;  //第 2 列有按键按下
34.              case 0x0B:col = 3;break;  //第 3 列有按键按下
35.              case 0x07:col = 4;break;  //第 4 列有按键按下
36.              default:col = 0xFF;
37.           }
38.        }
39.     }
40.     //线反转：行输入，列输出低电平
41.     //配置行为高阻输入并开启上拉电阻
42.     P0M1 |= 0x09;   P0M0 &= ~0x09;     //设置 P0.0 和 P0.3 为高阻输入
43.     P2M1 |= 0xC0;   P2M0 &= ~0xC0;     //设置 P2.6 和 P2.7 为高阻输入
44.     P_SW2 |= 0x80;                     //将 EAXFR 位置 1，允许访问扩展 RAM 区特殊功能寄存器(XFR)
```

```
    P0PU |= 0x09;                       //开启 P0.1、P0.2 的上拉电阻
    P2PU |= 0xC0;                       //开启 P1.6、P1.7 的上拉电阻
    P_SW2 &= 0x7F;                      //将 EAXFR 位置 0，禁止访问 XFR
    //配置列为准双向
    P0M1 &= ~0x06;  P0M0 &= ~0x06;      //设置 P0.1 和 P0.2 为准双向口
    P1M1 &= ~0xC0;    P1M0 &= ~0xC0;    //设置 P1.6~P1.7 为准双向口

    COL1=0;COL2=0;COL3=0;COL4=0;        //列输出低电平
    tmp = ROW_DATA;
    if(tmp != 0x0F)//有键按下
    {
       switch(tmp)
       {
          case 0x0E:row = 0;break;  //第 1 列有按键按下（列编号从 0 开始是为了后面的计算方便）
          case 0x0D:row = 1;break;  //第 2 列有按键按下
          case 0x0B:row = 2;break;  //第 3 列有按键按下
          case 0x07:row = 3;break;  //第 4 列有按键按下
          default:row = 0xFF;
       }
       key_val = row*4 + col;
    }
    while(ROW_DATA != 0x0F);//等待按键释放
    return key_val;
}
```

4. 主函数

主函数中，调用 4×4 矩阵按键识别函数实时检测是否有按键按下，如果有按键按下，通过串口输出按键的编号（S3~S18）代码清单如下。

代码清单：主函数

```
/*********************************************************************************
功能描述：主函数
参    数：无
返 回 值：int 类型
*********************************************************************************/
int main(void)
{
    u8 temp;

    P3M1 &= 0xFE;   P3M0 &= 0xFE;     //设置 P3.0 为准双向口（串口 1 的 RxD）
    P3M1 &= 0xFD;   P3M0 |= 0x02;     //设置 P3.1 为推挽输出（串口 1 的 TxD）

    uart1_init();                     //串口 1 初始化
```

```
    EA = 1;                                 //使能总中断
    while(1)
    {
        temp=KeyScan();                     //按键识别
        if(temp!=0xFF)                      //有键按下
        {
            //串口打印上传的采集的原始值为了和开发板上的按键丝印一致，这里需要加 2
            printf("按键 %bd 按下\r\n",temp+2);
        }
    }
}
```

4.1.3. 硬件连接

矩阵按键电路和指示灯 D1、D2 以及外部晶振复用 GPIO。使用矩阵按键时，需要将矩阵按键的跳线帽短接上，同时断开指示灯 D1、D2 的跳线帽，并且开发板不能使用外部晶振。

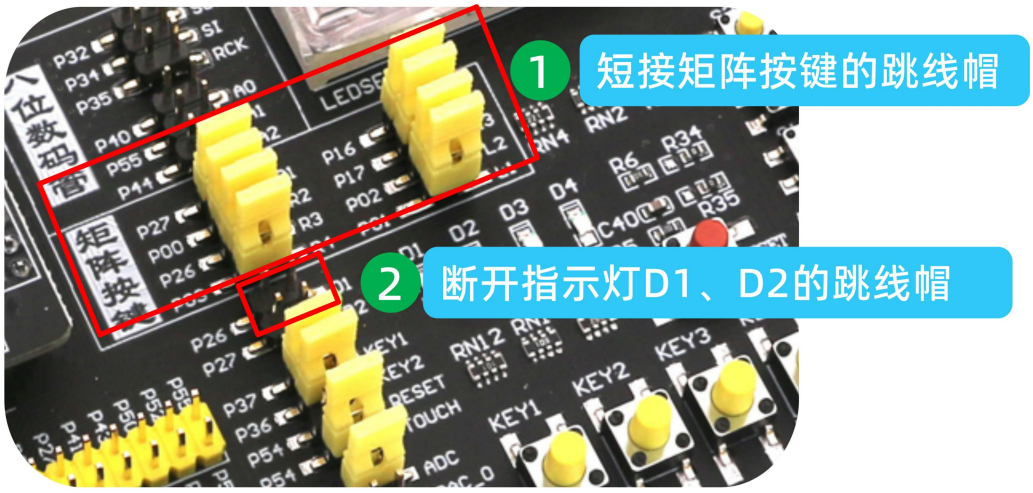


图 6：硬件连接

4.1.4. 实验步骤

1) 解压“…\第 3 部分：配套例程源码”目录下的压缩文件“实验 2-19-1：4×4 矩阵按键识别（串口发送键值）”，将解压后得到的文件夹拷贝到合适的目录，如“D\STC8”（**这样做的目的是为了防止中文路径或者工程存放的路径过深导致打开工程出现问题**）。
2) 双击“…\matrix_keyboard_4x4\project”目录下的工程文件“matrix_keyboard_4x4.uvproj”。
3) 点击编译按钮编译工程，编译成功后生成的 HEX 文件“matrix_keyboard_4x4.hex”位于工程的“…\matrix_keyboard_4x4\Project\Object”目录下。
4) 打开 STC-ISP 软件下载程序，下载使用内部 IRC 时钟，IRC 频率选择：24MHz。

5) 电脑上打开串口调试助手，选择开发板对应的串口号，将波特率设置为 9600bps。

6) 程序运行后按动 4×4 矩阵按键中的按键，可以在串口调试助手接收框中观察到开发板发送的按键信息，如下图所示。

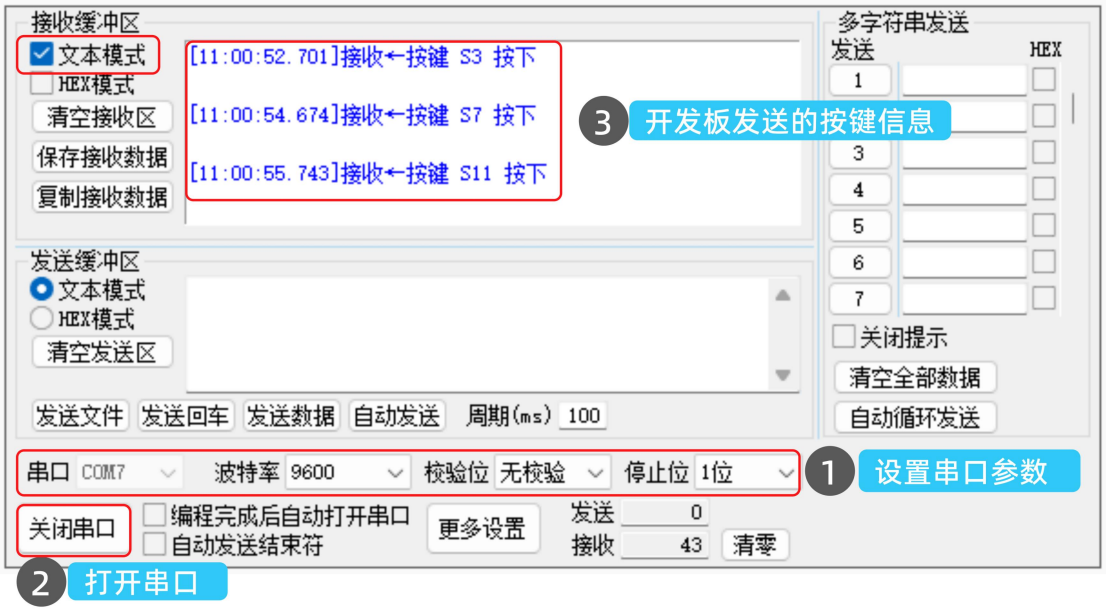


图 7：串口接收的按键信息

4.2. 4×4 矩阵按键识别（数码管显示键值）实验

✧ 注：本节的实验是在“实验 2-19-1：4×4 矩阵按键识别（串口发送键值）”的基础上修改，本节对应的实验源码是：“实验 2-19-2：4×4 矩阵按键识别（数码管显示键值）”。

4.2.1. 实验内容

用线反转方式识别矩阵按键，成功识别后，使用数码管的第 1 位和第 2 位显示出键值。

✧ 关于数码管显示的内容，读者可以参阅《第 2-12 讲：数码管显示》。

4.2.2. 代码编写

串口输出键值的例子中，识别到按键按下后，通过串口将按键的键值发送给计算机。我们做如下修改即可：

1) 识别到按键按下后，将获取的键值按照 10 进制提取出十位和个位（因为 4×4 矩阵按键的按键数量是 16 个，所以只有十位和个位）。

2) 调用数码管显示更新函数“LEDseg_DispUpdata()”更新个位显示。

3) 调用数码管显示更新函数“LEDseg_DispUpdata()”更新十位显示。这里需要注意：如果十位为 0，即键值小于 10，关闭十位数码管的显示，如果十位不为 0，即键值在 10~16 的范围内，显示十位。

数码管显示键值的代码清单如下（省略了数码管等的初始化代码）。

代码清单：数码管显示键值

```
while(1)
{
    temp=KeyScan();                                    //得到键值
    if(temp != 0xFF)
    {
        Temp += 2;              //为了和开发板上的按键丝印一致，这里需要加 2
        node1 = (temp/10)%10; //十位
        node2 = temp%10;        //个位
        if(node1 != 0)//十位不等于 0，显示十位
        {
            LEDseg_DispUpdata(LEDSEG_1,node1,LEDSEG_DP_OFF);//更新第 1 个数码管显示内容
        }
        else           //十位等于 0，关闭十位显示
        {
            LEDseg_DispUpdata(LEDSEG_1,LEDSEG_NODE_OFF,LEDSEG_DP_OFF);//更新第 8 个数码管显示内容
        }
        LEDseg_DispUpdata(LEDSEG_2,node2,LEDSEG_DP_OFF);//更新显示个位
    }
}
```

4.2.3. 硬件连接

按照下图所示短接跳线帽。

图 8：硬件连接

4.2.4. 实验步骤

1) 解压“…\第 3 部分：配套例程源码”目录下的压缩文件“实验 2-19-2：4×4 矩阵按键

识别（数码管键值）”，将解压后得到的文件夹拷贝到合适的目录，如“D\STC8”（**这样做的目的是为了防止中文路径或者工程存放的路径过深导致打开工程出现问题**）。

2) 双击“…\matrix_keyboard_4x4_ledseg\project”目录下的工程文件“matrix_keyboard_4x4_ledseg.uvproj”。

3) 点击编译按钮编译工程，编译成功后生成的 HEX 文件“matrix_keyboard_4x4_ledseg.hex”位于工程的“…\matrix_keyboard_4x4\Project\Object”目录下。

4) 打开 STC-ISP 软件下载程序，下载使用内部 IRC 时钟，IRC 频率选择：24MHz。

5) 程序运行后，按动 4×4 矩阵按键中的按键，可以观察到数码管上会显示对应的键值。