ICCAVR 向 WinAVR 过渡指南

相关 ICC/GCC 笔记：http://bbs.armavr.com/

以下内容硬件兰本为 ATmega16，软件版本为：ICCAVR7.14+WinAVR-20090313，并以附录 2 个功能完全一样的 ICCAVR 和 WinAVR 程序为例来对 ICCAVR 向 WinAVR 的过渡技巧进行叙述，以供由 ICCAVR 向 WinAVR 过渡的朋友参考，不足之处，欢迎反馈，一起探讨。

1、头文件需要更改

ICCAVR 中使用的“#include <iom16v.h>”需更换为“#include <avr/io.h>”

其他头文件应以程序而定，可参考 WinAVR 手册。

2、WinAVR 中延时函数可使用“#include <util/delay.h>”头文件中的延时函数：

```
void _delay_us (double __us);     //微秒级
void _delay_ms (double __ms);     //毫秒级
```

两个延时函数的参数均为 double 型，通常我们使用整数延时即可，

如：_delay_ms(100); //延时 100ms

需要注意的时，两个延时函数的最长延时时间都是有限制的，请查阅“util/delay.h”有详细说明。

3、WinAVR 中没有参数的函数需写出“void”，如：void LED_on(void)；否则出现下面警告：

warning: function declaration isn't a prototype

WinAVR 中没有类型的函数需写出“void”，如：void LED_on(void)；否则出现下面警告：

warning: data definition has no type or storage class //提示没有返回类型

warning: type defaults to 'int' in declaration of 'LED_on' //编译器将其类型定为默认的“int”型

4、WinAVR 源程序的最后一行需增加一个空行，否则出现下面警告：

warning: no new line at end of file

```
27  int main(void)
28      {
29      port_init();
30      while (1)
31          {
32            key_scan();
33          }
34          //MCUCR=0x40;
35          //MCUCR=0x50;
36          //MCUCR=0x60;
37          //MCUCR=0x70;
38          //MCUCR=0xE0;
39          MCUCR=0xF0;
40          asm("sleep");
41      }
42
```

5、中断函数写法两者不一样

WinAVR 中使用中断时，需添加“#include <avr/interrupt.h>”头文件：

```
#include <avr/interrupt.h>

ISR(ADC_vect)                    //中断名（中断向量名称）
{
    // user code here            //用户程序
}
```

中断向量名称可在：“avr-libc Manual [WinAVR]”的“<avr/interrupt.h>”中查询

6、WinAVR 位操作与 ICCAVR 不同，在“sfr_defs.h”头文件中定义出：
#define _BV(bit) (1 << (bit))
因“sfr_defs.h”头文件已经被“avr/io.h”头文件所包含，所以使用位操作时只需添加“avr/io.h”头文件即可

7、WinAVR 中，不同型号的单片机的头文件已被“avr/io.h”所包含，因此不需要添加额外对应型号的单片机头文件，这个很方便。

8、WinAVR 中多文件编译解决办法，参考附录 WinAVR 例程：

① 每个.c 文件对应一个.h 文件，.h 文件中写出.c 文件的所有函数声明，其他.c 文件调用该.c 文件中的函数时，只需包含该.c 文件的.h 文件即可。

② Makefile 文件中需列出所有.c 文件的清单，以空格隔开。

③ 满足上面两步即可实现 WinAVR 的多文件编译。

9、WinAVR 中空指令使用没有定义“nop();”，可以使用“asm("nop");”

10、WinAVR 中全局中断操作使用：“cli(); //禁止所有中断”和“sei(); //开全局中断”，它在“<avr/interrupt.h>”中定义，需将其包含；而 ICC 中为大写：“CLI(); //禁止所有中断”和“SEI(); //开全局中断”。

11、WinAVR 和 ICCAVR 最大的不同在于它需要编写单独的 Makefile 文件，Makefile 文件可用“MFile [WinAVR]”工具生成，稍作更改即可，而通常情况下只需更改以下几项：
① 芯片类型，此例为“MCU = atmega16”

② 芯片工具频率，此例为“F_CPU = 3686400”

③ 编译输入烧录文件格式，此例为“FORMAT = ihex”(默认)

④ 目标文件名，此例为“TARGET = main”（默认）

⑤ C 源程序清单，需以空格隔开，此例为“SRC = $(TARGET).c port_init.c led.c key_scan.c”，默认为“SRC = $(TARGET).c”，需依据程序结构进行添加。

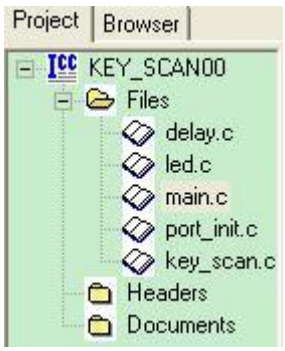
```
# MCU name
MCU = atmega16
```

```
#         F_CPU = 20000000
F_CPU = 3686400


# Output format. (can be srec, ihex, binary)
FORMAT = ihex


# Target file name (without extension).
TARGET = main


# Object files directory
#     To put object files in current directory, use a dot (.), do NOT make
#     this an empty or blank macro!
OBJDIR = .


# List C source files here. (C dependencies are automatically generated.)
SRC = $(TARGET).c port_init.c led.c key_scan.c
```

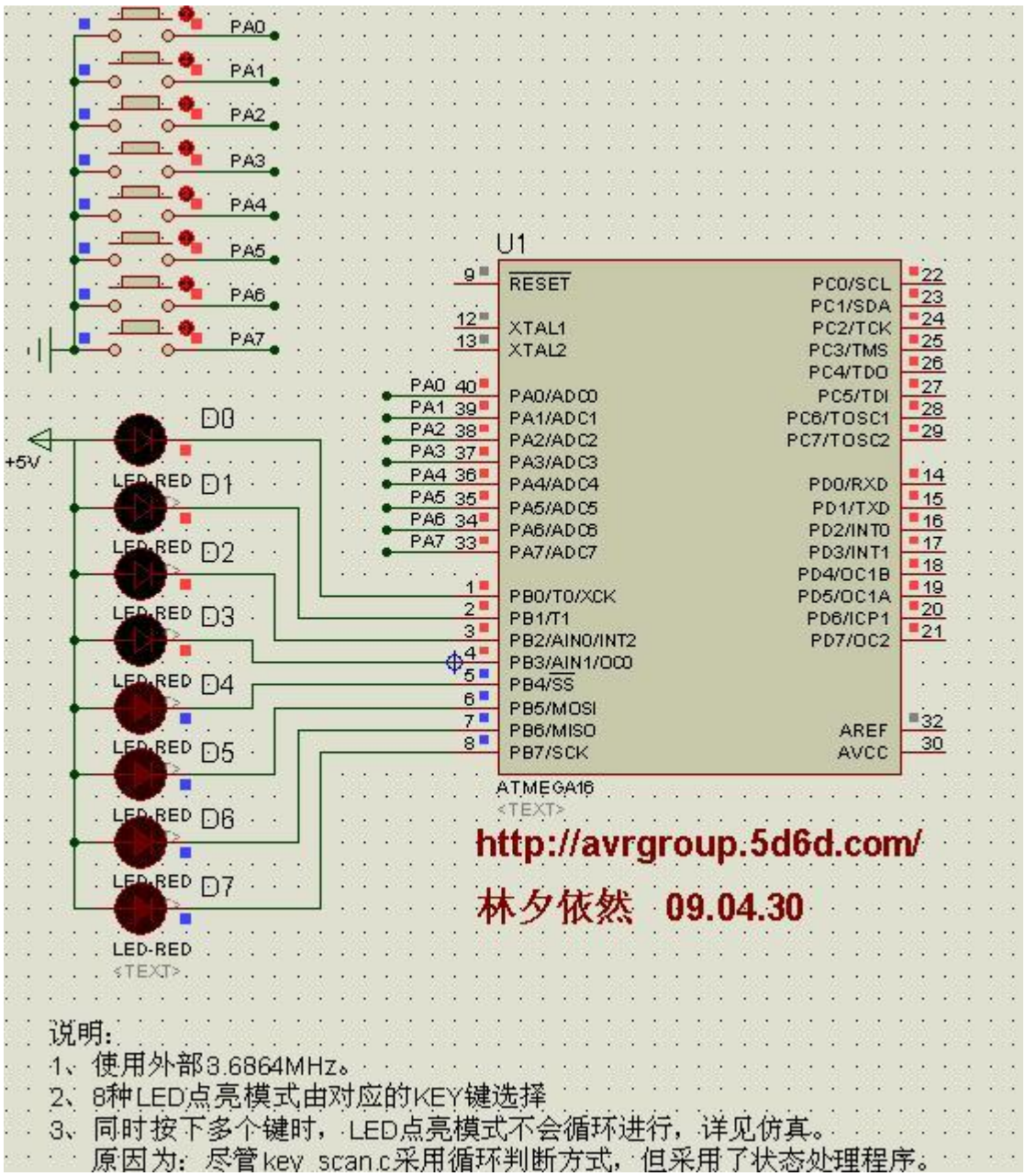
附录：

ICCAVR7.14 程序例程

一、程序结构



二、仿真效果



三、程序源码

1、main.c

```
/*********************************************************************************
Platform : AVR mega16 学习板（www.iccavr.com）
Project  : 实验四：按键扫描+8 种 LED 亮灭模式控制
Clock F  : 3.6864M
Software : ICCAVR7.14C
Author   : 林夕依然
Version  : 08.11.22
Updata   : 09.02.25  模块化
           09.04.30  增加 proteus 仿真模型，修改 key_scan.c 检测程序，仿真通过
comments :
1、以学习板八个 LED 灯和八个按键为硬件电路,JP7 短路块需装上
2、AVR 单片机端口寄存器的使用及理解
3、练习程序模块化，结构化的书写
4、端口电平检测程序的编写方法
5、8 种 LED 点亮模式由对应的 KEY 键选择，同时按下多个键时，LED 点亮模式不会循环进行，
   详见仿真。原因为：尽管 key_scan.c 采用循环判断方式，但采用了状态处理程序
6、增加状态处理，防止运行选定 LED 模式序时按下其他按键
Problem :
1、当前模式运行时切换其他模式时必须先复位，否则不能实现切换，考虑自动切换实现方法
解决方法：更改 key_scan.c 检测程序，使用 for 循环，不用 while(1)无限循环。
*********************************************************************************/
#include <iom16v.h>
#include <macros.h>

void main()
    {
    port_init();
    while (1)
     {
          key_scan();
     }
  //MCUCR=0x40;          //空闲模式，CPU 占用 100%
  //MCUCR=0x50;          //ADC 噪声抑制模式，CPU 占用 100%
  //MCUCR=0x60;          //掉电模式，CPU 占用 80%
  //MCUCR=0x70;          //省电模式，CPU 占用 4%
  //MCUCR=0xE0;          //Standby 模式，CPU 占用 80%
  MCUCR=0xF0;            //扩展 Standby 模式，CPU 占用 4%
  asm("sleep");             //CPU 休眠指令
    }
```

2、key_scan.c

```
#include <iom16v.h>
#include <macros.h>
/***键盘扫描函数***/
void key_scan()
{
```

```
    int l,m,n,o,p,q,r,s,i;

if((PINA&(1<<PA0))==0)          //模式 1：顺序点亮
    {
      PORTB=0XFF;
      for(l=0;l<5;l++)
     {
        for (i = 0; i < 8; i++)      //顺序单个点亮 LED
        LED_01(i);
        for (i = 6; i > 0; i--)      //逆序单个点亮 LED
        LED_01(i);
     }
  LED_off();//关闭 LED
  while(PINA!=0XFF);            //状态处理，防止运行此程序时按下其他按键
    }

    if((PINA&(1<<PA1))==0)          //模式 2：顺序单个间隔点亮
    {
      PORTB=0XFF;
      for(m=0;m<5;m++)
      {
       for (i = 0; i < 8; i += 2)  //顺序间隔点亮 LED
       LED_01(i);
       for (i = 7; i > 0; i -= 2)  //逆序间隔点亮 LED
       LED_01(i);
      }
   LED_off();//关闭 LED
   while(PINA!=0XFF);            //状态处理，防止运行此程序时按下其他按键
    }

    if((PINA&(1<<PA2))==0)          //模式 3：间隔点亮
    {
      PORTB=0XFF;
      for(n=0;n<5;n++)
      {
        for (i = 2; i < 8; i++)      //间隔顺序同时点亮
        LED_02(i);
        for (i = 6; i > 2; i--)  //间隔逆序同时点亮
        LED_02(i);
      }
   LED_off();//关闭 LED
   while(PINA!=0XFF);            //状态处理，防止运行此程序时按下其他按键
    }

    if((PINA&(1<<PA3))==0)          //模式 4：相临点亮
    {
      PORTB=0XFF;
      for(o=0;o<5;o++)
      {
```

```
      for (i = 1; i < 8; i++)      //相临顺序同时点亮
      LED_03(i);
      for (i = 6; i > 1; i--)  //相临逆序同时点亮
      LED_03(i);
    }
LED_off();//关闭 LED
while(PINA!=0XFF);              //状态处理，防止运行此程序时按下其他按键
 }

 if((PINA&(1<<PA4))==0)          //模式 5：发散聚集点亮
 {
   PORTB=0XFF;
   for(p=0;p<5;p++)
   {
     for(i=0;i<4;i++)             //发散点亮
     LED_04(i);
     for(i=2;i>0;i--)             //聚集点亮
     LED_04(i);
   }
LED_off();//关闭 LED
while(PINA!=0XFF);              //状态处理，防止运行此程序时按下其他按键
 }

 if((PINA&(1<<PA5))==0)          //模式 6：四四点亮
 {
   PORTB=0XFF;
   for(q=0;q<5;q++)
   {
     for(i=0;i<4;i++)             //四四顺序点亮
     LED_05(i);
     for(i=2;i>0;i--)             //四四逆序点亮
     LED_05(i);
   }
LED_off();//关闭 LED
while(PINA!=0XFF);              //状态处理，防止运行此程序时按下其他按键
 }

 if((PINA&(1<<PA6))==0)          //模式 7：四四点亮
 {
   PORTB=0XFF;
   for(r=0;r<5;r++)
   {
     for(i=0;i<2;i++)             //四四顺序点亮
     LED_06(i);
   }
LED_off();//关闭 LED
while(PINA!=0XFF);              //状态处理，防止运行此程序时按下其他按键
 }
```

```
    if((PINA&(1<<PA7))==0)          //模式 8：全部点亮熄灭
    {
      PORTB=0XFF;
      for(s=0;s<5;s++)
     {
        LED_on();
        LED_off();
     }
  LED_off();//关闭 LED
  while(PINA!=0XFF);              //状态处理，防止运行此程序时按下其他按键
    }
}
```

3、led.c

```
/*******************************
Platform : AVR mega16 学习板（www.iccavr.com）
function ：功能函数集
Clock F  : 3.6864M
Software : ICCAVR7.14C
Author   : 林夕依然
Version  : 09.02.25
comments :
*******************************/
#include <iom16v.h>
#include <macros.h>
void LED_on()                           //打开所有 LED
    {
PORTB =0X00;
DelayMs(100);
}

void LED_off()                          //关闭所有 LED
    {
PORTB = 0xFF;
DelayMs(100);
}
void LED_01(int i)                      //LED 亮灭控制
    {
    PORTB = ~BIT(i);                    //输出低电平
    DelayMs(100);                       //调用延时程序
    }

void LED_02(int i)                      //间隔点亮
    {
PORTB=~(BIT(i)|BIT(i-2));
DelayMs(100);
}

void LED_03(int i)                      //相临点亮
```

```
    {
PORTB=~(BIT(i)|BIT(i-1));       //~后内容需用括号括起来
DelayMs(100);
}

void LED_04(int i)                  //发散聚集点亮
    {
switch(i)
  {
  case 0:PORTB=0xE7;DelayMs(100);break;     //延时 100ms
  case 1:PORTB=0xDB;DelayMs(100);break;
  case 2:PORTB=0xBD;DelayMs(100);break;
  case 3:PORTB=0x7E;DelayMs(100);break;
  default:break;
  }
}

void LED_05(int i)                  //00,0F,F0,FF 方式显示
    {
switch(i)
  {
  case 0:PORTB=0x00;DelayMs(100);break;     //延时 100ms
  case 1:PORTB=0x0F;DelayMs(100);break;
  case 2:PORTB=0xF0;DelayMs(100);break;
  case 3:PORTB=0xFF;DelayMs(100);break;
  default:break;
  }
}

void LED_06(int i)
   {
   switch(i)
    {
case 0:PORTB=0XAA;DelayMs(100);break;
case 1:PORTB=0X55;DelayMs(100);break;
}
   }
```

4、port_init.c

```
#include <iom16v.h>
#include <macros.h>
/***端口初始化函数***/
void port_init()
{
DDRA =0X00;                         //端口上拉输入
PORTA=0XFF;
    DDRB =0xFF;                         //端口输出
    PORTB=0xFF;                         //输出高电平，LED 熄灭
DDRC =0X00;
```

```
PORTC=0XFF;
DDRD =0X00;
PORTD=0XFF;
}
```

5、delay.c

```
/*******************************
Platform : AVR mega16 学习板（www.iccavr.com）
function : 延时函数
Clock F  : 3.6864M
Software : ICCAVR7.14C
Author   : 林夕依然
Version  : 09.02.25
comments :
1、两种方式实现延时
*******************************/
/*--------------------------------------------------------------------------------
     延时程序计算方法
     计数个数 j = 延时时间/6*晶振频率 - 1
--------------------------------------------------------------------------------*/
#define uchar unsigned char
#define uint unsigned int

//方式一：
void Delay()
    {
    uchar a, b, c;
    for (a = 1; a; a++)
        for (b = 1; b; b++)
            for (c = 0; c<10; c++)  //循环次数=255*255*10
          ;
    }

//方式二：1ms 延时,准确性较 Delay();高
void DelayMs(uint i)
   {
    while(i--)
    {
     uint j;
        for(j=1;j<=613;j++)
      ;
    }
   }
```

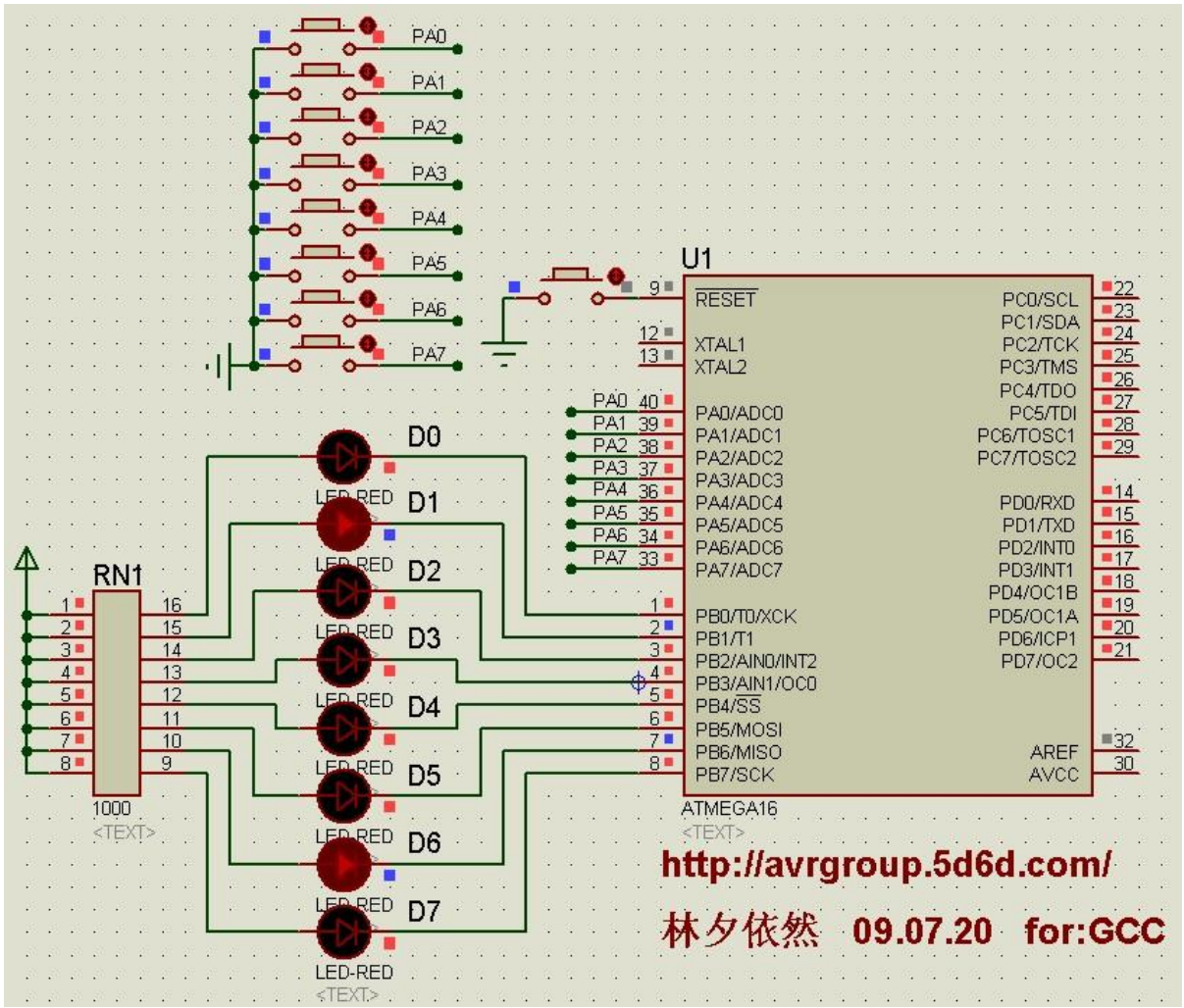
四、完整项目文件下载

http://avrgroup.5d6d.com/thread-70-1-1.html

WinAVR-20090313 程序例程

一、程序结构

二、仿真效果



三、程序源码

1、main.c

```
/*********************************************************************************
Platform : AVR mega16 学习板（www.iccavr.com）
Project  : 实验四：按键扫描+8 种 LED 亮灭模式控制 GCC
Clock F  : 3.6864M
Software : WinAVR-20071221+Proteus7.4
Author   : 林夕依然
```

Version : 08.11.22

Updata : 09.02.25 模块化

09.04.30 增加 proteus 仿真模型，修改 key_scan.c 检测程序，仿真通过

09.07.21 WinAVR-20071221 环境下编译通过

comments :

1、以学习板八个 LED 灯和八个按键为硬件电路,JP7 短路块需装上

2、AVR 单片机端口寄存器的使用及理解

3、练习程序模块化，结构化的书写

4、端口电平检测程序的编写方法

5、8 种 LED 点亮模式由对应的 KEY 键选择，同时按下多个键时，LED 点亮模式不会循环进行，详见仿真。原因为：尽管 key_scan.c 采用循环判断方式，但采用了状态处理程序

6、增加状态处理，防止运行选定 LED 模式序时按下其他按键

7、使用 GCC delay.h 头文件中延时函数，更换 io 头文件

Problem :

1、ICC 中 DelayMs()函数在 GCC 中不可用，不知何因。

**********/

```
#include <avr/io.h>
#include "port_init.h"
#include "key_scan.h"

int main(void)
    {
    port_init();
    while (1)
     {
          key_scan();
     }
  //MCUCR=0x40;          //空闲模式，CPU 占用 100%
  //MCUCR=0x50;          //ADC 噪声抑制模式，CPU 占用 100%
  //MCUCR=0x60;          //掉电模式，CPU 占用 80%
  //MCUCR=0x70;          //省电模式，CPU 占用 4%
  //MCUCR=0xE0;          //Standby 模式，CPU 占用 80%
  MCUCR=0xF0;           //扩展 Standby 模式，CPU 占用 4%
  asm("sleep");         //CPU 休眠指令
    }
```

2、key_scan.c

```
#include <avr/io.h>
#include "key_scan.h"
#include "led.h"
/***键盘扫描函数***/
```

```
void key_scan(void)
{
    int l,m,n,o,p,q,r,s,i;

if((PINA&(1<<PA0))==0)          //模式 1：顺序点亮
    {
      PORTB=0XFF;
      for(l=0;l<5;l++)
     {
        for (i = 0; i < 8; i++)      //顺序单个点亮 LED
        LED_01(i);
        for (i = 6; i > 0; i--)      //逆序单个点亮 LED
        LED_01(i);
     }
  LED_off();//关闭 LED
  while(PINA!=0XFF);            //状态处理，防止运行此程序时按下其他按键
    }

    if((PINA&(1<<PA1))==0)          //模式 2：顺序单个间隔点亮
    {
      PORTB=0XFF;
      for(m=0;m<5;m++)
      {
       for (i = 0; i < 8; i += 2)  //顺序间隔点亮 LED
       LED_01(i);
       for (i = 7; i > 0; i -= 2)  //逆序间隔点亮 LED
       LED_01(i);
      }
   LED_off();//关闭 LED
   while(PINA!=0XFF);            //状态处理，防止运行此程序时按下其他按键
    }

    if((PINA&(1<<PA2))==0)          //模式 3：间隔点亮
    {
      PORTB=0XFF;
      for(n=0;n<5;n++)
      {
        for (i = 2; i < 8; i++)      //间隔顺序同时点亮
        LED_02(i);
        for (i = 6; i > 2; i--)  //间隔逆序同时点亮
```

```
      LED_02(i);
    }
LED_off();//关闭 LED
while(PINA!=0XFF);              //状态处理，防止运行此程序时按下其他按键
 }

 if((PINA&(1<<PA3))==0)          //模式 4：相临点亮
 {
   PORTB=0XFF;
   for(o=0;o<5;o++)
   {
     for (i = 1; i < 8; i++)     //相临顺序同时点亮
     LED_03(i);
     for (i = 6; i > 1; i--)  //相临逆序同时点亮
     LED_03(i);
   }
LED_off();//关闭 LED
while(PINA!=0XFF);              //状态处理，防止运行此程序时按下其他按键
 }

 if((PINA&(1<<PA4))==0)          //模式 5：发散聚集点亮
 {
   PORTB=0XFF;
   for(p=0;p<5;p++)
   {
     for(i=0;i<4;i++)            //发散点亮
     LED_04(i);
     for(i=2;i>0;i--)            //聚集点亮
     LED_04(i);
   }
LED_off();//关闭 LED
while(PINA!=0XFF);              //状态处理，防止运行此程序时按下其他按键
 }

 if((PINA&(1<<PA5))==0)          //模式 6：四四点亮
 {
   PORTB=0XFF;
   for(q=0;q<5;q++)
   {
     for(i=0;i<4;i++)            //四四顺序点亮
```

```
        LED_05(i);
        for(i=2;i>0;i--)              //四四逆序点亮
        LED_05(i);
      }
   LED_off();//关闭 LED
   while(PINA!=0XFF);              //状态处理，防止运行此程序时按下其他按键
    }

    if((PINA&(1<<PA6))==0)          //模式 7：四四点亮
    {
      PORTB=0XFF;
      for(r=0;r<5;r++)
      {
        for(i=0;i<2;i++)             //四四顺序点亮
        LED_06(i);
      }
   LED_off();//关闭 LED
   while(PINA!=0XFF);              //状态处理，防止运行此程序时按下其他按键
    }

    if((PINA&(1<<PA7))==0)          //模式 8：全部点亮熄灭
    {
      PORTB=0XFF;
      for(s=0;s<5;s++)
     {
        LED_on();
        LED_off();
     }
  LED_off();//关闭 LED
  while(PINA!=0XFF);              //状态处理，防止运行此程序时按下其他按键
    }
}
```

3、led.c

```
/*******************************
Platform : AVR mega16 学习板（www.iccavr.com）
function : 功能函数集
Clock F  : 3.6864M
Software : WinAVR-20071221
Author   : 林夕依然
Version  : 09.07.20
```

```
comments :
*******************************/
#include <avr/io.h>
#include <util/delay.h>
#include "led.h"
void LED_on(void)                          //打开所有 LED
    {
PORTB =0X00;
_delay_ms(100);
}

void LED_off(void)                         //关闭所有 LED
    {
PORTB = 0xFF;
_delay_ms(100);
}
void LED_01(int i)                  //LED 亮灭控制
    {
    PORTB = ~_BV(i);                //输出低电平
    _delay_ms(100);                   //调用延时程序
    }

void LED_02(int i)                  //间隔点亮
    {
PORTB=~(_BV(i)|_BV(i-2));
_delay_ms(100);
}

void LED_03(int i)                 //相临点亮
    {
PORTB=~(_BV(i)|_BV(i-1));       //~后内容需用括号括起来
_delay_ms(100);
}

void LED_04(int i)                 //发散聚集点亮
    {
switch(i)
  {
  case 0:PORTB=0xE7;_delay_ms(100);break;     //延时 100ms
  case 1:PORTB=0xDB;_delay_ms(100);break;
```

```
  case 2:PORTB=0xBD;_delay_ms(100);break;
  case 3:PORTB=0x7E;_delay_ms(100);break;
  default:break;
  }
}

void LED_05(int i)                    //00,0F,F0,FF 方式显示
    {
switch(i)
  {
  case 0:PORTB=0x00;_delay_ms(100);break;     //延时 100ms
  case 1:PORTB=0x0F;_delay_ms(100);break;
  case 2:PORTB=0xF0;_delay_ms(100);break;
  case 3:PORTB=0xFF;_delay_ms(100);break;
  default:break;
  }
}

void LED_06(int i)
   {
   switch(i)
    {
case 0:PORTB=0XAA;_delay_ms(100);break;
case 1:PORTB=0X55;_delay_ms(100);break;
}
   }
```

4、port_init.c

```
#include <avr/io.h>
#include "port_init.h"
/***端口初始化函数***/
void port_init(void)
{
DDRA =0X00;                           //端口上拉输入
PORTA=0XFF;
    DDRB =0xFF;                             //端口输出
    PORTB=0xFF;                             //输出高电平，LED 熄灭
DDRC =0X00;
PORTC=0XFF;
DDRD =0X00;
```

```
PORTD=0XFF;
}
```

5、key_scan.h

```
void key_scan(void);
```

6、led.h

```
void LED_on(void);
void LED_off(void);
void LED_01(int i);
void LED_02(int i);
void LED_03(int i);
void LED_04(int i);
void LED_05(int i);
void LED_06(int i);
```

7、port_init.h

```
void port_init(void);
```

四、完整项目文件下载

http://avrgroup.5d6d.com/thread-796-1-1.html