【单片机实验】按键实验（一，二，三）

原创

Larya_csdn 于 2017-11-09 18:24:30 发布 13202 收藏 34

分类专栏： 单片机 文章标签： 单片机

本文链接：https://blog.csdn.net/qq_35824338/article/details/78492586

单片机 专栏收录该内容

5 篇文章 1 订阅

订阅专栏

按键实验：

目的：通过按键控制其他元器件

实验一

K1~K4控制LED移位

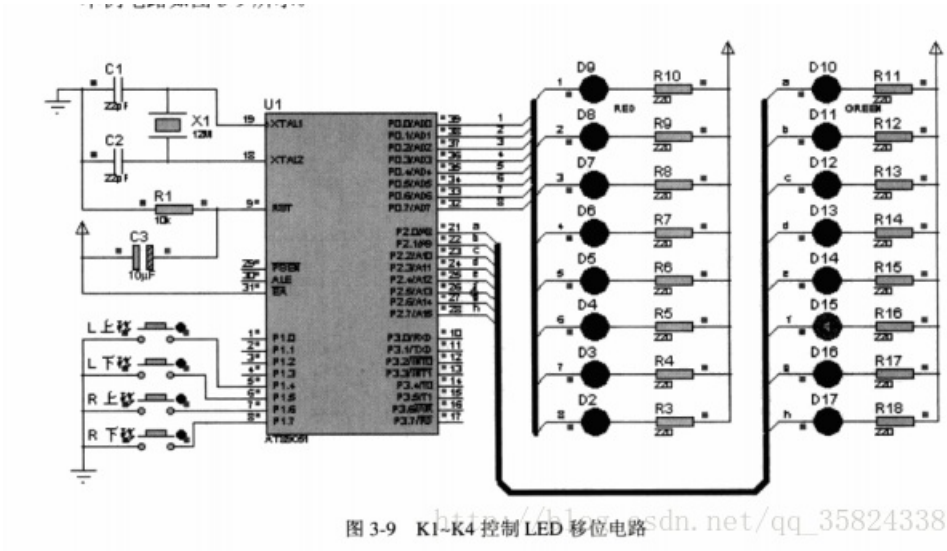

图 3-9 K1~K4 控制 LED 移位电路

目的：按下独立按键K1~K4,可分别上下控制连接在P0，P2端口的LED移位显示。

```c
#include<reg51.h>//使用芯片的库函数
#include<intrins.h>//*intrins.h，函数，C51单片机编程中，使用到空指令_nop_();字符循环移位指令_crol_等时使用。*
#define uint unsigned int//define宏定义
#define uchar unsigned char

void DelayMS(uint x)//延时函数，不同的芯片由于频率不同而不同
{
    uchar i;
    while (x--)
        for (i = 0;i < 120;i++);
}

void Move_LED()//LED灯移动功能
//P1，按键引脚；P0：左流水灯；P2：右流水灯；
{
    if ((P1 & 0x10) == 0) P0 = _crol_(P0, 1);
    //左边流水灯向上移动
    else if ((P1 & 0x20) == 0) P0 = _cror_(P0, 1);
    //左边流水灯向下移动
    else if ((P1 & 0x40) == 0) P2 = _crol_(P2, 1);
    //右边流水灯向上移动
    else if ((P1 & 0x80) == 0) P2 = _cror_(P2, 1);
    //右边流水灯向下移动
}
```

实验二

K1~K4按键状态显示：

LED1，2和3，4，控制灯的亮灭不同。

按下K1或K2时，LED1或2亮，松开熄灭。按下K3或K4后释放时，LED3或4 亮，再次按下并释放时熄灭。

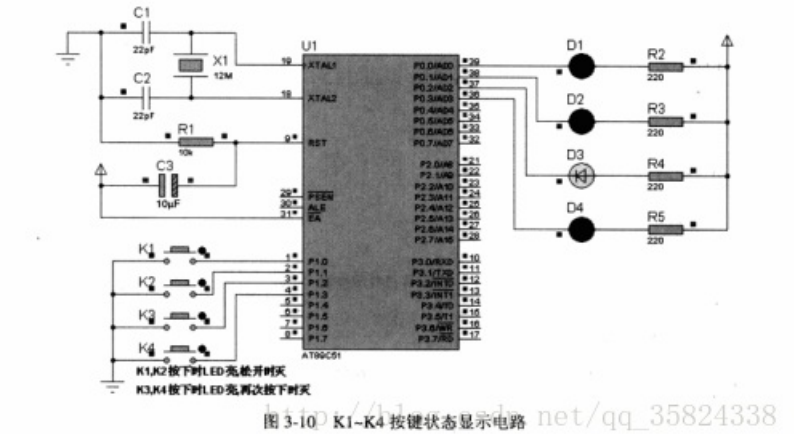

图 3-10 K1~K4 按键状态显示电路


```c
#include<reg51.h>
#define uchar unsigned char
#define uint unsigned int

sbit LED1 = P0 ^ 0;
sbit LED2 = P0 ^ 1;
sbit LED3 = P0 ^ 2;
sbit LED4 = P0 ^ 3;

sbit K1 = P1 ^ 0;
sbit K2 = P1 ^ 1;
sbit K3 = P1 ^ 2;
```

```
sbit K4 = P1 ^ 3;

void DelayMS(uint x)
{
    uchar i;
    while (x--)
        for (i = 0;i < 120;i++);
}

void main()
{

    P1 = 0xFF;
    P0 = 0xFF;
    while (1)
    {
        LED1 = K1;
        LED2 = K2;
        if (K3 == 0)
        {
            while (K3 == 0);
            LED3 = ~LED3;

        }
        if (K4 == 0)
        {
            while (K4 == 0);
            LED4 = ~ LED4;
        }
        DelayMS(10);
    }
}

void main()
{
    P1 = 0xFF;
    P0 = 0xFF;
    while (1)
    {
        LED 1 = K1;
//K1= 1，灯灭；K1 = 0，灯亮
//LED1，2，由按键直接控制，按键状态即是灯的状态
//LED3，4，由按键间接控制，按键第一次按下

        if (K3 == 0)//按下K3，K3=0
        {
            while (K3 == 0);
//保持当时灯的状态，等待释放按键，释放按键后，执行下一条指令，即是相反状态

            LED3 = ~LED3;
//实现灯的相反状态，两次按键状态切换
        }
        if (K4 == 0)
        {
            while (K4 == 0)
                LED4 = ~LED4;
        }
        DelayMS(10);
    }
}
```

重点

—while（1）和while（1）；（有分号）—

```
    while(1) {……} :是让单片机一直执行 {……} 中内容，**防止程序跑飞**，通常用于主程序主体，确保程序持续执行
    while(1);      ：是一条指令，它让单片机停在这个位置，一般用来检测中断,只有cpu收到中断指令，才会跳出while(1)，进入中断服务子程序；
    whlie(1)本质是死循环，while(1)中的指令会不断重复执行，除非有中断，while(1);可以看作while(1){//空指令}，它执行的是空指令，
    *于是单片机就停在这行代码处*
```

—按键的高低电平—

按键的高低电平

```
    很明显：我们是高电平有效，未按下时，是高电平，为1；当K1=1时，灯是灭的；从P0，P1的引脚初始化也可以看出。
```

实验三

K1~K4分组控制LED

实验内容：在得出键号后分别对LED执行4种不同的操作。

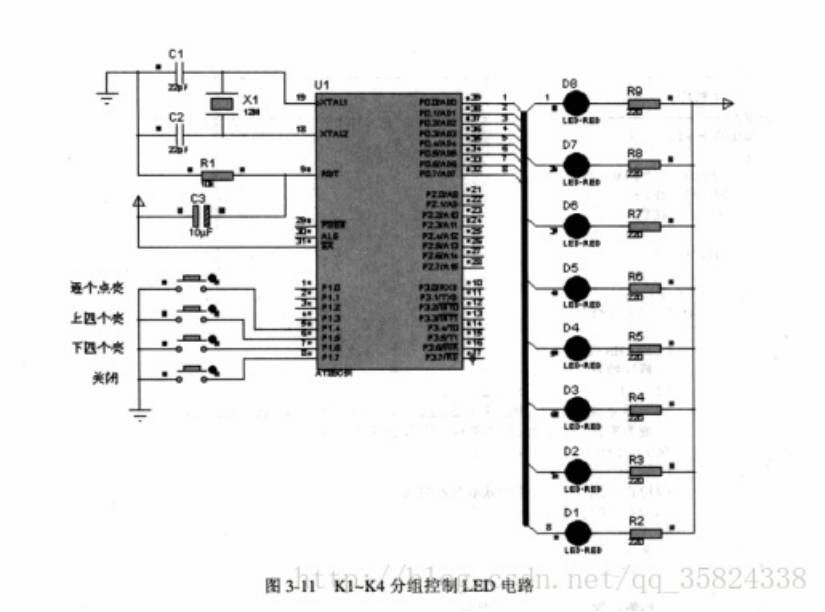

图 3-11 K1~K4 分组控制 LED 电路

```c
#include<reg51.h>
#define uchar unsigned char
#define uint unsigned char

void DelayMS(uint x)
{
    uchar t;
    while (--x)
        for (t = 0;t < 120;t++);
}

void main()
{
    uchar k, t, Key_State;//定义变量，以及键盘
    P0 = 0xFF;//初始化P0,P1引脚
    P1 = 0xFF;

    while (1)
    {
        t = P1;      //记录P1 初值
        if (t!= 0xFF)//
        {
            DelayMS(10);
            if (t != P1) continue;//再次检查按键
            Key_State = -t >> 4;//取得4位按键值，
            //由模式XXXX1111（X中有一位为0，其他均为1）
            //变为模式0000XXXX（X中有一位为1，其他均为0）
            k = 0;
            //检查1所在的位置，累加获取按键号K
            while (Key_State != 0)
            {
                k++;
                Key_State >>= 1;

            }
            //根据按键号K进行4种处理

//（当前情况是在循环体内，switch中的指令会多次判断并执行）
switch (k)
        {
        case 1:if (P0 == 0x00) P0 = 0xFF;//1111 1111
                P0 <<= 1;//P0左移一位 1111 1110 左移
                DelayMS(200);
                break;
        case 2:P0 = 0xF0;break;//点亮【上面】4只
        case 3:P0 = 0x0F;break;//点亮【下面】4只
        //涉及知识点：共阴极还是阳极，以及引脚对应问题
        case 4:P0 = 0xFF;//关闭所有
        }

        }
    }
}
```

重点

—引脚和值的对应关系（反序）—

```
P0口的8个引脚被赋值成1111,1110，就是说只有P0^0口是低电平，其余都是高电平！8个引脚与0xfe的2进制分别对应！
    即是1111,1110的值对应P0的7，6，5，4，3，2，1，0号引脚，不同于我们数数时01234567从左到右的逻辑，这种现象在汇编语言和单片机里很常见
```

—循环移位和移位（CF的作用有无）—

```
循环移位：单片机中的移位运算不同于C语言中的移位之后补零，而是分情况进行大循环和小循环（即是微机原理中的不带进位CF的CRL和带进位CF的ROR 循环移位）常使用_crol_()和_cror_()函数完成。
```

“`

移位：>> 右移；<<左移

```
    本实验则是【未使用】循环移位。故在switch语句中进行

case 1:if (P0 == 0x00) P0 = 0xFF;
//若灯亮全灭
P0 <<= 1;
//（原来是ff 1111 1111 变 1111 1110，亮一个-》亮两个，3个，四个……八个，循环，全灭，也是逐个，从下到上移动。
```

—共阳极和共阴极（灯该怎么亮？）—

```
    灯亮取决于高电平还是低电平，由共阳极和共阴极决定。
共阳极:低电平0亮，（和电源形成电平差，点亮LED灯）
共阴极：高电平1亮。
```

后语

认真做了按键实验，感觉单片机并不简单，需要自己好好努力，认真对待才行，各位大朋友，小朋友加油！（PS：markdown用的还有点无力，下次加油吧）