

# 实验四：中断实验

原创

孙凌星 于 2020-04-18 14:27:14 发布 2433 收藏 4

分类专栏：[51单片机基础学习](#) 文章标签：[单片机](#)

版权声明：本文为博主原创文章，遵循[CC 4.0 BY-SA](#) 版权协议，转载请附上原文出处链接和本声明。

本文链接：[https://blog.csdn.net/qq\\_45284276/article/details/105598131](https://blog.csdn.net/qq_45284276/article/details/105598131)

版权



[51单片机基础学习](#) 专栏收录该内容

4 篇文章 1 订阅

订阅专栏

## 实验三：中断实验

强调：本文章为新手提供学习参考

实验四：中断实验

使用的开发板原理图及本次使用的模块

数码管

备注：

中断

实验四的代码部分

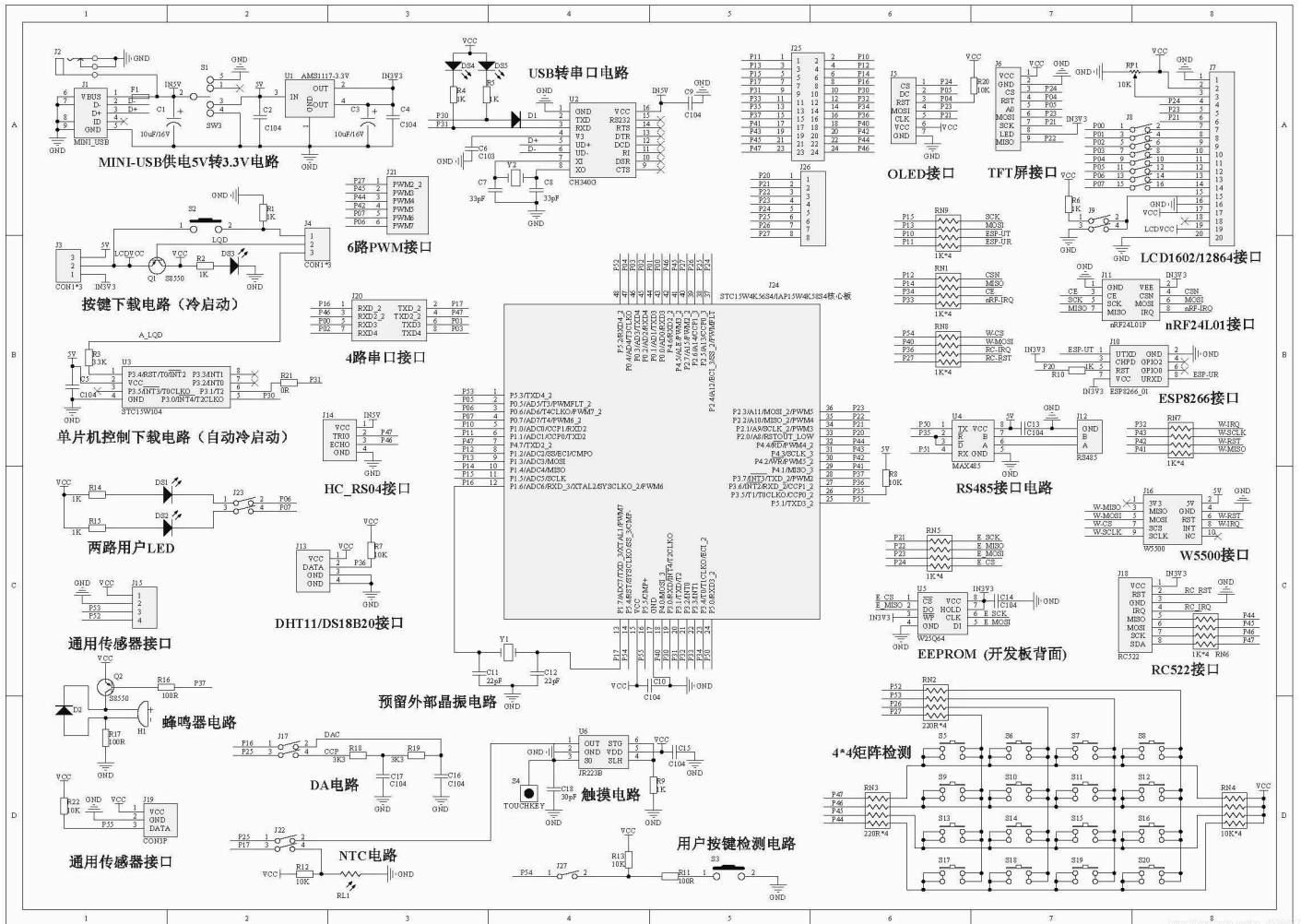
强调：本文章为新手提供学习参考

## 实验四：中断实验

使用定时器来实现延时，编程实现数码管静态显示秒表的倒计时。（此实验与实验三相似，只需要几处代码稍加改动就行）

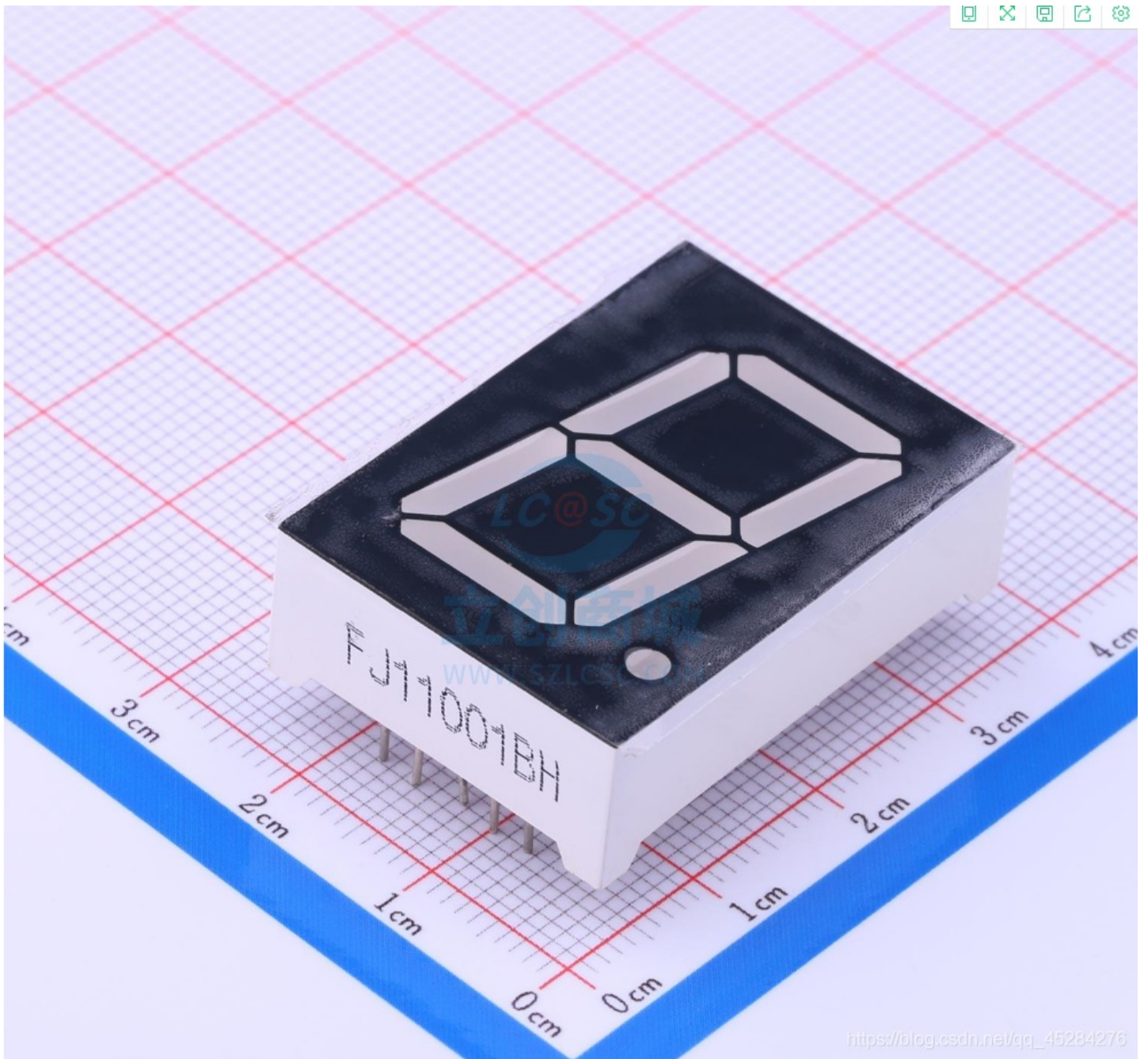
使用的开发板原理图及本次使用的模块

进取者STC15开发板原理图

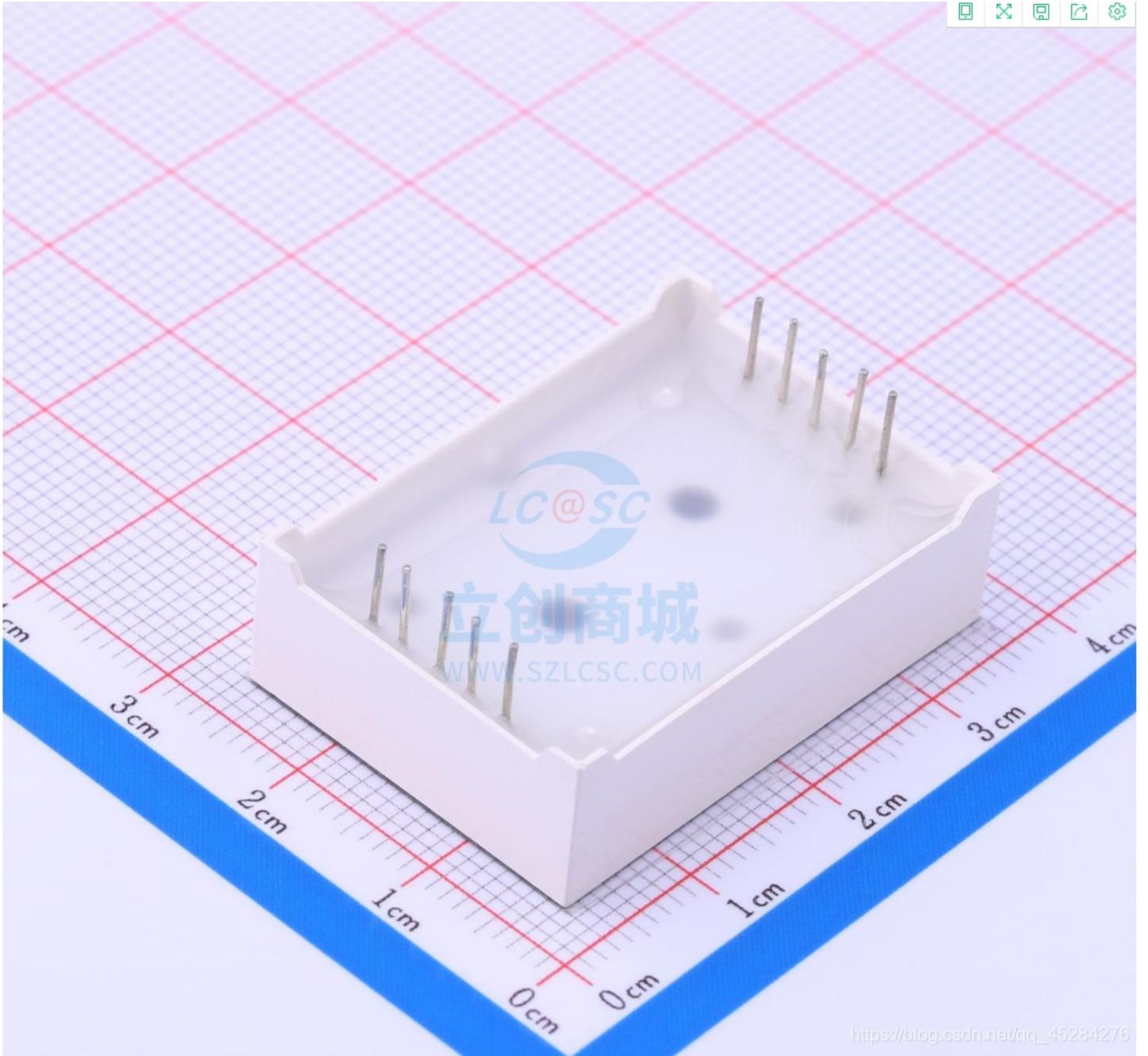


本实验采用拓展口连接数码管的八个引脚实现功能

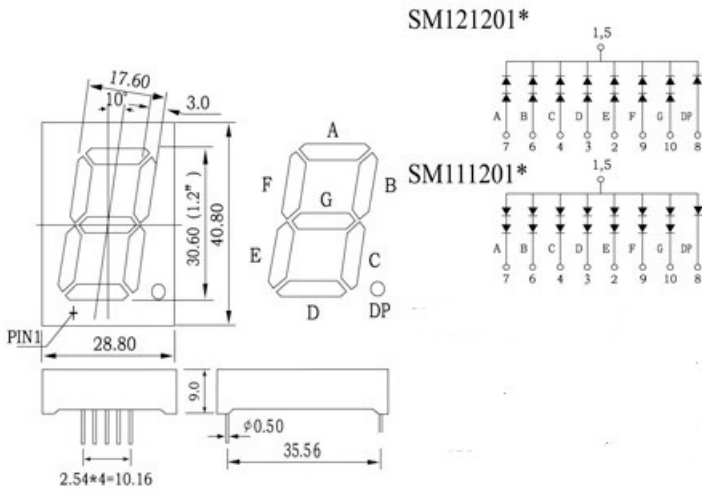
## 数码管



数码管正面



数码管背面



[https://blog.csdn.net/qq\\_45284276](https://blog.csdn.net/qq_45284276)

### 数码管内部原理

以共阳极数码管为例，从图中可以看出，数码管由8个LED组成，共阳极他们的阳极接在了一起，只要外部给一个VCC就可以通过控制他们的阴极来控制数码管显示

共阳极代码：0xc0,0xf9,0xa4,0xb0,0x99,0x92,0x82,0xf8,0x80,0x90,0x88,0x83,0xc6,0xa1,0x86,0x8e

对应0 1 2 3 4 5 6 7 8 9 a b c d e f十六个符号

共阴极代码：0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f,0x77,0x7c,0x39,0x5e,0x79,0x71

对应0 1 2 3 4 5 6 7 8 9 a b c d e f十六个符号

### 备注：

本次实验使用的是P0\_0到P0\_78个管脚，实际开发板中用杜邦线接出来（图片来源网络，侵权）

模块1个脚提供高电平，其余脚与单片机P0口连接

### 中断

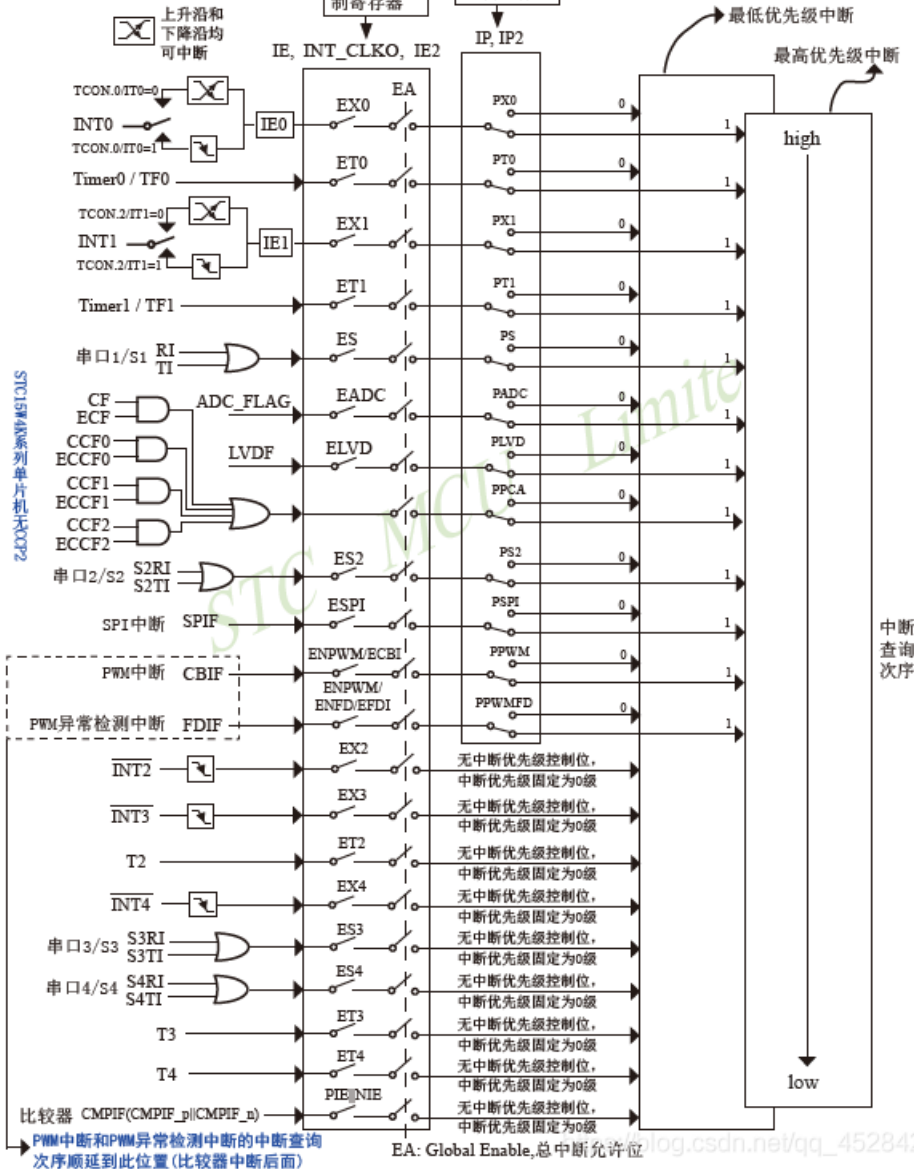
STC15W4K32S4系列单片机提供了21个中断请求源，它们分别是：外部中断0(INT0)、定时器0中断、外部中断1(INT1)、定时器1中断、串口1中断、A/D转换中断、低压检测(LVD)中断、CCP/PWM/PCA中断、串口2中断、SPI中断、外部中断2( $\overline{\text{INT2}}$ )、外部中断3( $\overline{\text{INT3}}$ )、定时器2中断、外部中断4( $\overline{\text{INT4}}$ )、串口3中断、串口4中断、定时器3中断、定时器4中断、比较器中断、PWM中断及PWM异常检测中断。除外部中断2( $\overline{\text{INT2}}$ )、外部中断3( $\overline{\text{INT3}}$ )、定时器2中断、串口3中断、串口4中断、定时器3中断、定时器4中断及比较器中断固定是最低优先级中断外，其它的中断都具有2个中断优先级。

[https://blog.csdn.net/qq\\_45284276](https://blog.csdn.net/qq_45284276)

STC15w4k58s4与STC15W4k32s4为一个系列，只不过io口有点区别，K32s4的有62个io口，k58s4有48个io口，其余没有太多的差别，单片机所带的中断请求源如图示，有很多，所以说STC15系列芯片是很强大的，

## 6.2 中断结构图

注意：当定时器/计数器0工作在不可屏蔽中断的16位自动重载模式时，其优先级是所有中断中最高的，而且不受总中断允许位EA控制。



这是中断执行的内部结构图，通过开关选择不同的模式，在此就不多说了，因为中断这些代码可以直接用相关软件生成。

## 5、定时器T0和T1的中断控制寄存器:IE和IP

IE：中断允许寄存器（可位寻址）

SFR name	Address	bit	B7	B6	B5	B4	B3	B2	B1	B0
IE	A8H	name	EA	ELVD	EADC	ES	ET1	EX1	ET0	EX0

EA：CPU的总中断允许控制位，EA=1，CPU开放中断，EA=0，CPU屏蔽所有的中断申请。

EA的作用是使中断允许形成多级控制。即各中断源首先受EA控制；其次还受各中断源自己的中断允许控制位控制。

ET1：定时/计数器T1的溢出中断允许位，ET1=1，允许T1中断，ET1=0，禁止T1中断。

ET0：T0的溢出中断允许位，ET0=1允许T0中断，ET0=0禁止T0中断。

IP：中断优先级控制寄存器（可位寻址）

SFR name	Address	bit	B7	B6	B5	B4	B3	B2	B1	B0
IP	B8H	name	PPCA	PLVD	PADC	PS	PT1	PX1	PT0	PX0

PT1：定时器1中断优先级控制位。

当PT1=0时，定时器1中断为最低优先级中断(优先级0)

当PT1=1时，定时器1中断为最高优先级中断(优先级1)

PT0：定时器0中断优先级控制位。

当PT0=0时，定时器0中断为最低优先级中断(优先级0)

当PT0=1时，定时器0中断为最高优先级中断(优先级1)

**注意：**当定时器/计数器0工作在模式3(不可屏蔽中断的16位自动重载模式)时，不需要允许EA/IE. 7(总中断使能位)只需允许ET0/IE. 1(定时器/计数器0中断允许位)就能打开定时器/计数器0的中断，此模式下的定时器/计数器0中断与总中断使能位EA无关。一旦此模式下的定时器/计数器0中断被打开后，该定时器/计数器0中断优先级就是最高的，它不能被其它任何中断所打断(不管是比定时器/计数器0中断优先级低的中断还是比其优先级高的中断，都不能打断此时的定时器/计数器0中断)，而且该中断打开后既不受EA/IE. 7控制也不再受ET0控制了，清零EA或ET0都不能关闭此中断。

[https://blog.csdn.net/qq\\_45284276](https://blog.csdn.net/qq_45284276)

本次使用到的就是T0的定时器中断，我们按照datasheet所示编写好代码就可以使用定时器中断功能了。

(摘自STC15datasheet)

## 实验四的代码部分

```

#include <reg51.h> //调用reg51头文件
#define uchar unsigned char //定义uchar类型为unsigned char 储存大小1字节, 值范围: 0 到 255
#define uint unsigned int //定义uint类型为unsigned int 储存大小2或4个字节, 值范围: 0~65536
uchar code tab[]={0xc0,0xf9,0xa4,0xb0,0x99,0x92,0x82,0xf8,0x80,0x90,0x88,0x83,0xc6,0xa1,0x86,0x8e}; //一维数组表示
数码管显示“0~F”共阳极
uchar i,j,k=9; //i, j开始为0.k开始为9

void display() //显示函数
{
    P0=tab[k]; //P0等于对应的一维数组中值, 从而数码管显示相应的数值, 例如: k=1, P0=tab[k]=0xc0;
}
void InitTimer0(void) //使用定时器0
{
    TMOD = 0x01; //选择为定时器0模式, 工作方式1, 仅用TR0打开启动
    TH0 = 0xFC; //给定时器赋初值, 定时1ms
    TL0 = 0x18;
    EA = 1; //打开定时器0中断允许
    ET0 = 1; //打开总中断
    TR0 = 1; //打开定时器
}

void main(void)
{
    InitTimer0(); //定时器初始化
    while(1)
    {
        display();
    }
}

void Timer0Interrupt(void) interrupt 1 //定时器0中断函数
{
    TH0 = 0xFC; //给定时器赋初值, 定时1ms
    TL0 = 0x18;
    i++; //每过1ms, i+1一次
    if(i==100) //当i达到100时, 及经历了0.1s, 进入判断
    {
        i=0; j++; //让i恢复初值, 每次i=100时, j+1一次
        if(j==10)
        {
            j=0; //让j恢复初值
            k--; //k-1
            if(k==0)
            {
                k=10; //让k恢复初值
            }
        }
    }
}

```

打字不易, 各位少侠点个赞吧, 感谢!!!

- 。
- 。



- 
- 
- 

与实验三相比，

```
k--;//k-1
if(k==0)
{
    k=10;//让k恢复初值
}
```

在这里，只是把之前实验三默认0改为十，自加改为自减就行了。

还有一种写法就是把 `uchar code tab[]=`

`{0xc0,0xf9,0xa4,0xb0,0x99,0x92,0x82,0xf8,0x80,0x90,0x88,0x83,0xc6,0xa1,0x86,0x8e};` 里面代码的顺序改一下，其他的和实验三一样也可以。