//头文件  
#include<reg52.h>  
#include <intrins.h>  
//宏定义  
#define uchar unsigned char  
#define uint unsigned int    
#define TN  65436  
//进程1，2，3执行时间之比为  T1：T2：T3  （时间单位us）  
#define TN1  62536   //1个进程循环周期内进程1执行的时间T1us  TN1=(65536-T1)  
#define TN2  55536   //1个进程循环周期内进程2执行的时间T2us  TN2=(65536-T1)  
#define TN3  60536   //1个进程循环周期内进程3执行的时间T3us  TN3=(65536-T1)  
//  
#define N1  10    // 进程1的延时参数  邮箱：[niewenli.2007@163.com](mailto:niewenli.2007@163.com)  
#define N2  10    // 进程2的延时参数  联系方式：13572614184  
#define N3  10    // 进程3的延时参数  QQ：723992349

//定义全局变量  
uint address1,address2,address3;  
uchar test1\_1=0,test2\_1=0,test3\_1=0,PID=1;  
//各进程的标志位，是否为第一次执行；PID进程号；  
uint ac1,ac2,ac3; //, PC\_Next; 各进程的初始地址寄存器.  
//test1    的参数  由于进程切换时 没有保存普通变量，  
//所以各进程的普通参数需要定义成全局变量.  
uint m1,i1,j1,k1;  
uchar table1[4];  
//test2    的参数   邮箱：[niewenli.2007@163.com](mailto:niewenli.2007@163.com)  
int m2,i2,j2,k2;  
uchar table2[4];  
//test3    的参数 联系方式：13572614184  
int m3,i3,j3,k3;    // QQ：723992349  
uchar table3[4];

//声明  
//unsigned int Get\_Next\_PC(void);//调用子程序，获取PC  
void chushihua(void);   //初始化函数  
void test1(void);    //进程一  
void test2(void);  
void test3(void);

//main函数  
void main(void)  
{  
// PC\_Next=Get\_Next\_PC();   
 chushihua();  
 ac1=(unsigned int)(test1);  //获取进程1的入口地址  
 ac2=(unsigned int)(test2); //获取进程2的入口地址  
 ac3=(unsigned int)(test3);  //获取进程3的入口地址  
 //常规任务  
 while(1);  
}

//初始化时钟  
void chushihua(void)  
{   
    TMOD=0x01;  // 邮箱：[niewenli.2007@163.com](mailto:niewenli.2007@163.com)  
 EA=1;   // 联系方式：13572614184  
 ET0=1;   // QQ：723992349  
 TH0=TN/256;  
 TL0=TN%6;  
 TR0=1;  
}

//中断处理，进程调度  
void time0() interrupt 1 using 0  
{   
 TR0=0;  
//进程顺序分配   
 PID++;  
 if(PID==4)  
    {PID=1;}  
 //进程调度  
 switch(PID)  
 {  
   case 1:   
  if(test3\_1!=0)  
     { //保存现场，还回地址  
     address3=\*((unsigned char \*)(SP-4));  //PC的高字节  
           address3 <<= 8;  
           address3+=\*((unsigned char \*)(SP-5));   //PC的低字节  
     table3[0]=\*((unsigned char \*)(SP));   //现场保护  
     table3[1]=\*((unsigned char \*)(SP-1));  //现场保护  
     table3[2]=\*((unsigned char \*)(SP-2));  //现场保护  
     table3[3]=\*((unsigned char \*)(SP-3));  //现场保护  
  }  
     if(test1\_1==0)  
  { //执行新进程  
  test1\_1=1;  
     \*((unsigned char \*)(SP-4))=ac1>>8;    //PC的高字节  
        \*((unsigned char \*)(SP-5))=ac1 & 0x00ff;  //PC的低字节  
  }  
  else  
  { //执行新进程，恢复现场  
  \*((unsigned char \*)(SP-4))=address1>>8;     //PC的高字节  
        \*((unsigned char \*)(SP-5))=address1 & 0x00ff;  //PC的低字节  
  \*((unsigned char \*)(SP))=table1[0];     //现场恢复  
  \*((unsigned char \*)(SP-1))=table1[1];    //现场恢复  
  \*((unsigned char \*)(SP-2))=table1[2];    //现场恢复  
  \*((unsigned char \*)(SP-3))=table1[3];    //现场恢复  
  }  
    TH0=TN1/256;  
    TL0=TN1%6;  
       TR0=1;  
    break;

   case 2:  
     if(test1\_1!=0)  
     {  //保存现场，还回地址  
     address1=\*((unsigned char \*)(SP-4));  //PC的高字节  
           address1 <<= 8;  
           address1+=\*((unsigned char \*)(SP-5));  //PC的低字节  
     table1[0]=\*((unsigned char \*)(SP));    //现场保护  
     table1[1]=\*((unsigned char \*)(SP-1)); //现场保护  
     table1[2]=\*((unsigned char \*)(SP-2)); //现场保护  
     table1[3]=\*((unsigned char \*)(SP-3));  //现场保护  
  }  
  if(test2\_1==0)  
  { //执行进程2  
  test2\_1=1;           
     \*((unsigned char \*)(SP-4))=ac2>>8;    //PC的高字节  
        \*((unsigned char \*)(SP-5))=ac2 & 0x00ff;  //PC的低字节  
  }  
  else  
  { //执行进程2，恢复现场  
  \*((unsigned char \*)(SP-4))=address2>>8;  //PC的高字节  
        \*((unsigned char \*)(SP-5))=address2 & 0x00ff; //PC的低字节  
  \*((unsigned char \*)(SP))=table2[0];   //现场恢复  
  \*((unsigned char \*)(SP-1))=table2[1];  //现场恢复  
  \*((unsigned char \*)(SP-2))=table2[2];  //现场恢复  
  \*((unsigned char \*)(SP-3))=table2[3];   //现场恢复  
  }     
        TH0=TN2/256;  
     TL0=TN2%6;  
        TR0=1;  
    break;  
   
    case 3:  
     if(test2\_1!=0)  
     {  //保存现场，还回地址  
     address2=\*((unsigned char \*)(SP-4));   //PC的高字节  
           address2 <<= 8;  
           address2+=\*((unsigned char \*)(SP-5));   //PC的低字节  
     table2[0]=\*((unsigned char \*)(SP));    //现场保护  
     table2[1]=\*((unsigned char \*)(SP-1));   //现场保护  
     table2[2]=\*((unsigned char \*)(SP-2));   //现场保护  
     table2[3]=\*((unsigned char \*)(SP-3));  //现场保护  
  }  
  if(test3\_1==0)  
  { //执行进程3  
  test3\_1=1;  
     \*((unsigned char \*)(SP-4))=ac3>>8;    //PC的高字节  
        \*((unsigned char \*)(SP-5))=ac3 & 0x00ff;    //PC的低字节  
  }  
  else  
  { //执行进程3，恢复现场  
  \*((unsigned char \*)(SP-4))=address3>>8;    //PC的高字节  
        \*((unsigned char \*)(SP-5))=address3 & 0x00ff;  //PC的低字节  
  \*((unsigned char \*)(SP))=table3[0];     //现场恢复  
  \*((unsigned char \*)(SP-1))=table3[1];    //现场恢复  
  \*((unsigned char \*)(SP-2))=table3[2];    //现场恢复  
  \*((unsigned char \*)(SP-3))=table3[3];   //现场恢复  
  }     
        TH0=TN3/256;  
     TL0=TN3%6;  
        TR0=1;  
    break;  
   default:   
    TH0=TN/256;  
     TL0=TN%6;  
        TR0=1;  
   break;  
   }  
}

//进程一  P1演示二进制加法死循环 邮箱：[niewenli.2007@163.com](mailto:niewenli.2007@163.com)  
void test1(void)  
{  
while(1)  
 {  
    
   for(i1=0;i1<256;i1++)  
   {  
   for(k1=0;k1<=N1;k1++)  
   {for(j1=0;j1<=20;j1++)  
   for(m1=0;m1<=113;m1++);}    // 约1ms\*T1/T1+T2+T3  
   P1=i1;  
   //P2=0x0;  
   }  
 }  
}

//进程二 P2演示二进制加法死循环 联系方式：13572614184  
void test2(void)  
{  
while(1)  
 {  
    
   for(i2=0;i2<256;i2++)  
   {  
   for(k2=0;k2<=N2;k2++)  
   {for(j2=0;j2<=20;j2++)  
   for(m2=0;m2<=113;m2++);}  
   P2=i2;  
   }  
 }  
}

//进程三   P0口演示二进制加法死循环    进程3的延时参数  QQ：723992349  
void test3(void)  
{  
while(1)  
 {  
    
   for(i3=0;i3<256;i3++)  
   {  
   for(k3=0;k3<=N3;k3++)  
   {for(j3=0;j3<=20;j3++)  
   for(m3=0;m3<=113;m3++);}  
   P0=i3;  
   }  
 }  
}