//本程序采用IIC通信方式，4兆PIC16F877  
//是ADS1100,ADS1110的通用程序。  
刚通过。  
  
  
# include<pic.h>  
# define uch unsigned char  
# define unint unsigned int  
# define SDA  RD1  
# define SCL  RD0  
uch ACK;  
unint j;  
void delay()            //延时  
{  
for(j=5535;--j;) continue;  
}  
void start (void)       //开始  
{  
   TRISD0=0;  
   TRISD1=0;  
   SDA=1;  
   asm("nop");  
   SCL=1;  
   asm("nop");  
   asm("nop");  
   asm("nop");  
   SDA=0;  
   asm("nop");  
   asm("nop");  
   asm("nop");  
   SCL=0;  
   asm("nop");  
   asm("nop");  
}  
  
void stop(void)             //结束  
   {   
    TRISD0=0;  
    TRISD1=0;  
    SDA=0;  
    asm("nop");  
    SCL=1;  
    asm("nop");  
    asm("nop");  
    asm("nop");  
    SDA=1;  
    asm("nop");  
    asm("nop");  
    asm("nop");  
    SCL=0;  
    asm("nop");  
    asm("nop");  
    }  
    
    uch  check(void)    //  检查应答信号  
     {    
        TRISD1=1;  
        SCL=0;  
        SCL=1;  
     
        if (SDA )  
        {  
            ACK = 1;  
        }  
       else  
        {  
            ACK = 0;  
         }  
       SCL=0;  
      return(ACK);  
     }  
       
      void sendBYTE(uch data)   //  发送一个字节  
{  
     uch bitcount=8;               //发送8位  
  
          do  
          {   
            TRISD0=0;  
            TRISD1=0;            //置输出  
             
            if((data&0x80)==0x80)     
                {  
                 SDA=1;         //写 1  
                 }  
            else    
                {  
                 SDA=0;        //写 0  
                }  
            SCL=0;              //在时钟大于4u秒期间写数据  
            SCL=1;  
           asm("nop");  
           asm("nop");  
           asm("nop");  
           asm("nop");  
           asm("nop");  
            SCL=0;   
          data=data<<1;  
          bitcount--;  
          } while(bitcount);  
           TRISD1=1 ;                  //释放总线等待应答  
           asm("nop");  
           asm("nop");  
           asm("nop");  
           asm("nop");  
           asm("nop");           
}     
  
uch reciveBYTE(void)             //接受一个字节  
  {  
    uch temp1=0;  
    uch bitcount1=8;  
    TRISD0=0;  
    TRISD1=1;               //置输入  
   do  
   { SCL=0;              //在时钟大于4u秒期间读数据  
     SCL=1;  
     asm("nop");  
     asm("nop");  
    
    if(SDA)                //读 1  
       {   
        temp1=temp1|0x01;      
       }  
    else                     //读 0  
      {  
        temp1=temp1&0xfe;  
      }  
     SCL=0;  
     if(bitcount1-1)  
      {  
         temp1=temp1<<1;  
           
      }  
       bitcount1--;    
      }while(bitcount1);  
       return(temp1);  
  }    
    
   void ack(void)          //  发送连续读信号  
{   
    TRISD0=0;  
    TRISD1=0;  
    SDA=0;  
    SCL=0;  
    SCL=1;  
   asm("nop");  
   asm("nop");  
   asm("nop");  
   asm("nop");  
   asm("nop");  
    SCL=0;  
}  
      
  void nack(void)        //  发送不连续读信号  
  {   
    TRISD0=0;  
    TRISD1=0;  
    SDA=1;  
    SCL=0;  
    SCL=1;  
    asm("nop");  
    asm("nop");  
    asm("nop");  
    asm("nop");  
    asm("nop");  
    SCL=0;  
  }    
    
   void wrtoIIC(uch data)     //写器件地址和配置字  
  
  {   
  
      
     start();               //开始写  
     sendBYTE(0x90);         //写器件地址（写）  
           
     do{  
          ;  
        }while(check());       //等待应答  
         sendBYTE(data);      //写配置字  
    do{;  
        }while(check());  
          
     stop();            //结束写  
      
      
       
}       
  
   void rdfromIIC(void)       //从ADS1100中读出数据  
{    
  
    
     start();                  //开始读  
     sendBYTE(0x91);           //写器件地址（读）  
      do{  
         ;  
         }while(check());      //等待应答  
        
       
      TMR1H=reciveBYTE();     // 读高8位  
       ack();                  //连续读  
      TMR1L=reciveBYTE();     //读低8位  
      ack();                  //连续读  
      TMR2=reciveBYTE();      //读配置字  
      nack();  
      stop();                  //结束读  
  //    delay();  
     
}   
       
  
main()  
{  
     
  
     
   while(1)  
    {  
     di();  
     wrtoIIC( 0x8c);  
     rdfromIIC();  
    
    }  
}