在WIN32  API编程中,除了DCB结构之外,还需要了解COMMTIMEOUTS结构.这个结构是为了读写串口的超时而设置的.   
  
COMMTIMEOUTS结构如下：     
      
  typedef   struct   \_COMMTIMEOUTS   {       
  DWORD   ReadIntervalTimeout;                 //任意相邻连个字符之间的超时设置  
  DWORD   ReadTotalTimeoutMultiplier;        //读操作总的超时时间的系数   
  DWORD   ReadTotalTimeoutConstant;       //读操作总的超时时间的修正常量   
  DWORD   WriteTotalTimeoutMultiplier;       //写操作总的超时时间的系数  
  DWORD   WriteTotalTimeoutConstant;       //写操作总的超时时间的修正常量  
  }   COMMTIMEOUTS,\*LPCOMMTIMEOUTS;       
        
      
  　　ReadIntervalTimeout：两相邻字符之间最大的延时。当读串口数据时，一旦两个字符传输的时间间隔超过该时间，读函数将返回现有的数据。设置为0表示该参数不起作用。       
      
  　　ReadTotalTimeoutMultiplier：读操作总的超时事件的系数。 这个变量是不能单独使用的。 必须和ReadTotalTimeoutConstant 一起使用才有效果。      
      
  　　ReadTotalTimeoutConstant：读操作总的超时时间的修正常量。 这个变量也是不能单独使用的。必须和ReadTotalTimeoutMultiplier一起使用才有效果。   
      
  　　WriteTotalTimeoutMultiplier：写操作总的超时事件的系数。 这个变量是不能单独使用的。 必须和WriteTotalTimeoutConstant 一起使用才有效果。     
      
  　　WriteTotalTimeoutConstant：写操作总的超时时间的修正常量。 这个变量也是不能单独使用的。必须和WriteTotalTimeoutMultiplier一起使用才有效果。  
  
       在整个串口的读写操作中， 存在着两种超时设置。一种是间隔超时， 一种是总超时。 这两种超时是独立存在，互不影响的。   
  
      间隔超时， 只在读操作中存在。就是ReadIntervalTimeout。 当读操作中，前后两个字符之间的时间间隔超过时，读操作就结束了。举例来说，你一次读取8个字符，但是在你读取了第一个字符之后，在读取第二个字符时，间隔超时了，那么读操作就结束了， 这样整个操作就只读取了1个字节。 即使， 你的总时间没有超时。   
  
       另一种超时，就是总超时。 这里有一个公式。   
       总的读/写超时时间 = Read(Write)TotalTimeoutMultiplier x 要读/写的字节数 + Read(Write)TotalTimeoutConstant.   
       这里要说明的一点，要读/写的字节数是从哪里来的:这个是从ReadFile 或者WriteFile 函数中定义的,是第三个参数所给定的。  
       在读操作时， 若当前所花读取时间已经超过了总的超时设置， 则读操作就结束了。即使， 每两个字符之间的间隔没有超时。 举例来说， 若总共读取8个字节。 间隔设置为8ms， 总超时系数为3ms，总超时常数为3ms。 则总的超时时间为3\*8+3=27ms。若每个字符读取的间隔为7ms， 则这次操作总共能读取4个字符。 就结束了。因为读取第5个字符时， 已经需要35ms， 超过总超时时间了。  
  
下面来讨论一下这几个参数的设定：  
将ReadIntervalTimeout设置为MAXDWORD，将ReadTotalTimeoutMultiplier   和ReadTotalTimeoutConstant设置为0，表示读操作将立即返回存放在输入缓冲区的字符。     
  
将ReadIntervalTimeout设置为MAXDWORD，将ReadTotalTimeoutMultiplier   和ReadTotalTimeoutConstant设置为MAXDWORD， 表示读操作会一直等待直到所需要读取的字节数全部接收到为止。 （大家可以把MAXDWORD 认为是永远）  
  
将ReadIntervalTimeout设置为0， 则不使用间隔超时， 只考虑总超时设置。