附件6：如何更好地使用二次开发函数

说明书

说明书版本：V2.02

更新日期：2016.06.01

1、VCI_OpenDevice 函数

此函数用于连接并打开已插入计算机的USB-CAN适配器。

在通过此函数打开USB-CAN适配器时，所在的计算机进程中将自动生成一个针对USB-CAN进行操作的句柄，并同时在内存中建立相关资源。该函数必须与VCI_CloseDevice成对出现，即：**调用VCI_OpenDevice后，在退出进程、关闭软件、重新打开适配器等情况时，必须调用VCI_CloseDevice函数释放资源，否则可能引起进程崩溃、通信错误等未知错误！**

2、VCI_CloseDevice 函数

此函数用来关闭已打开的USB-CAN适配器，关闭后，适配器将不再进行收发活动，直到下一次打开启动。

该函数应与VCI_OpenDevice函数成对出现，关闭适配器的同时，释放系统资源！

3、VCI_InitCAN 函数

此函数在调用VCI_OpenDevice函数成功之后调用，用来初始化适配器上的一个CAN通道，该函数的形参pInitConfig传递了初始化相关的参数，其中包括滤波参数、波特率、工作模式等。

注意：

1.当参数设置界面打开时，USB-CAN Tool 的接收线程和发送线程将挂起，并停止正在进行的发送操作。因此，参数设置期间可能造成数据接收缓冲区的溢出或被迫停止发送操作而丢失数据！

2.CAN 总线在正常收发数据的时候，尽量不要通过 USBCAN 适配器修改 CAN 总线参数或关闭 CAN 总线，应等数据收发停止或将 USBCAN 适配器脱离 CAN 总线再进行相应操作。

4、VCI_GetReceiveNum 函数

该函数必须在CAN打开的状态下调用，用来获取在CAN适配器某个通道缓冲区中已经接收到的但未被VCI_Recive函数读取的帧的数量。

在多线程的二次开发应用中，可单独开一个线程，用此函数轮询CAN通道，在返回值大于0的时候，向接收线程发送一个“接收消息”，可实现类似的接收中断操作。

此函数的作用在于，通过调用此函数，可以知道CAN适配器某个通道缓冲区中已经接收到的但未被VCI_Recive函数读取的帧的数量，这个参数可以供VCI_Recive使用，即可以开辟一个动态空间，来接收数据，以节约内存。但是在实际编程中，内存一般不会出现紧张，所

以一般不用调VCI_GetReceiveNum，以节约CPU资源，而直接调用VCI_Recive，地址长度直接设为大于缓冲区长度2000帧，可以设为2500帧。

5、VCI_StartCAN 函数

该函数用在VCI_InitCAN函数调用成功后，作用是启动一个CAN通道，所启动的CAN通道必须经过VCI_InitCAN函数初始化后才能成功启动，通道启动后，即可通过调用VCI_Receive和VCI_Transmit函数接收和发送CAN消息。

6、VCI_Receive 函数

该函数必须在CAN通道启动后调用，用来读取CAN通道接收缓存区（设备驱动提供每个通道2000帧CAN消息长度的接收缓冲区，这为应用程序提供了充足的反应处理时间。）中未被读取的CAN消息，它的形参pReceive为接收数据帧数组的首指针；另一个形参len为接收数据帧数组的长度，如果接收缓冲区帧数小于或等于Len时，所有数据均会被读出；如果接收缓冲区帧数大于Len时，Len个数据被读出。所以对应用程序来讲，如果2000帧的数据存储空间不占用太多资源，那么Len可以设为2000及以上，这样能保证每次调用VCI_Receive函数都能把所有数据一次性读回。

在接收线程中，一般通过循环调用VCI_Receive函数来查询接收缓冲区并接收数据。关键参数有：调用频率、接收长度Len。一般情况Len设置成存储空间长度，或者根据实际接收频率及总线消息出现频率设置适当值，前提是这个接收数组要足够大，否则可能出现丢帧现象。

调用频率：调用VCI_Receive函数，实质是读写USB，读写USB有一个最小的耗时，一般为3～5ms，故，VCI_Receive函数调用间隔至少应设置在5ms以上。

调用频率、接收长度Len两个参数直接影响适配器的性能。假设总线速率为8000帧/s，接收长度Len为2000帧，则每秒至少要调用4次，即调用间隔小于250ms。如果接收长度Len设为1000帧，则每秒至少调用8次，即调用间隔小于125ms。否则接收缓冲区会溢出。如此即可匹配调用频率、接收长度Len，得到最优值。

7、VCI_Transmit 函数

该函数必须在CAN通道启动后调用，用来向CAN通道发送缓冲区（USB_CAN适配器提供约每个通道1000帧CAN消息长度的发送缓冲区，每次调用VCI_Transmit最多发送约1000帧数据。）发送数据，然后通过CAN总线发送出去，它的形参pSend为发送的数据帧数组的首指针；另一

个形参len为要发送的数据帧数量，可设置为1-1000之间的任意整数。

发送设备的发送速度由当前计算机软硬件性能决定，一般连续发送速度在8000 fps以上(1Mbps)，若发送速度过快将有可能使总线利用率达到100%，使发送方出现发送超时。这样用户可在应用程序中适当添加延时以降低发送速度。

发送过程中每次调用VCI_Transmit函数都有超时限制，发送时超时时间约10ms。发送超时一般由于CAN总线繁忙且当前节点优先级较低时发生，并不是函数调用或通讯错误，用户可以编程实现重发。因此，在系统设计时注意保证CAN总线占用不应该超过总线容量的60-70%。

由于VCI_Transmit函数与VCI_Receive函数类似，每次调用都有个3～5ms的固定耗时，故，要使USB_CAN适配器达到最高的发送速度，需要一次发送多帧，而不是一次发送一帧。实际测试中，线程中循环发送，一次发送48帧，可达到最佳发送速度8000帧/s以上。

调用频率：调用VCI_Transmit函数，实质是读写USB，读写USB有一个最小的耗时，一般为3～5ms，故，VCI_Transmit函数调用间隔至少应设置在5ms以上。

调用频率、单次发送帧数两个参数直接影响适配器的性能。假设需要达到发送速率为8000帧/s，单次发送帧数80帧，则每秒调用100次，即调用间隔为10ms。如果单次发送帧数设为160帧，则每秒调用50次，即调用间隔为20ms。如此即可匹配调用频率、单次发送帧数，得到最优值。发送间隔不能太小，否则总线占用率达100%时，发送缓冲区满，由于有堵塞机制，调用VCI_Transmit函数会超时返回，这时候每次调用VCI_Transmit函数都不会立即返回，而是等待10ms直到超时返回，反映到应用程序就是占用资源增加，界面响应迟钝。所以参数设置很重要。

重要说明：

在调用VCI_Receive和VCI_Transmit函数之间需要进行软件延时，一般根据接收或发送的帧数多少其间隔时间也略有不同，一般最短要间隔3~5ms为宜，尤其在高速发送时，如通过VCI_Transmit函数一次发送多帧数据时，需要适当增加间隔时间！

8、VCI_ConnectDevice 函数

该函数可在任意时刻调用，查询USB-CAN设备是否能和计算机正常通讯。

USB-CAN在运行过程中，可能受到外界干扰或者异常操作导致USB-CAN设备从操作系统中掉线。此函数即是用来检测通讯连接状况的。如发现异常，可进行提示和相应处理。

9、VCI_FindUsbDevice 函数

此函数可以在任意时刻调用，用来查询计算机中所有插入的USB-CAN适配器的信息，并

返回前4个适配器的信息，若计算机中插入多余4个适配器，则使用VCI_FindUsbDevice2函数。