USB 发送数据时出现迟滞现象

关键字：USB，迟滞

1 问题描述

客户反馈，使用 STM32F446 的高速 USB 外设，即 USB_OTG_HS 外设，且使用内置全速 PHY。客户的产品 USB 用做 device，自定义 HID 类，当连接带 UOS 操作系统的 HOST 时，会发现当前数据并没有成功发送，但是会发送上一次的数据，即发送数据出现"迟滞"现象。但在 Windows 下却没有出现此类问题。另外，客户同时还使用了 STM32F446 上的 USB_OTG_FS 外设，且此外设做同样的事一切正常，目前此问题只出现在 USB_OTG_HS 外设上。

2 问题分析及解决方法

刚开始猜测是长度问题，即发送最大包长需要再发送一次空包。但客户反馈他们的发送长度为 62 个字节。于是去客户现场使用 USB 协议分析仪采数分析，发现一切通信正常。

通过查看客户演示重现问题的过程，发现在正常时是一切 OK 的，只在进行 USB 拔插时才发送问题。应用程序不断发送数据的过程中拔掉 USB 线，然后再次插上，在此过程中应用程序一直尝试发送数据。当 USB 线重新连接上且重新枚举成功后，“迟滞”现象则重现了，即每次应用程序调用发送接口实现发送的是上一次尝试发送的内容。

调试客户的程序，发现当 USB 线拔掉后，应用程序还会往 USB IP 对应的发送 FIFO 内写入数据，这其实是不对的。按理 USB 线拔掉后 USB 的状态应该恢复到默认状态，即 pdev->dev_state=USBD_STATE_DEFAULT. 但实际上，通过调试发现此状态在 USB 线拔掉后是 suspend 状态。那么为什么会是这样的呢？于是立即想到 Vbus sensing 功能。马上与客户硬件工程师核对，原来客户产品的 USB_OTG_HS 的 Vbus_sensing 脚是悬空的，并没有连接 Vbus，但是客户的 USB_OTG_FS 外设却又是连接了。于是客户的产品两个 USB 口，同样的工作，一个 USB 口 正常，另一个 USB 口却会出现问题。

差异找到了，接下来就是分析由此如何造成问题的。

由于 USB_OTG_HS 并没有真正实现 Vbus sensing 功能（因为没有硬件连接），于是当 USB 线断开时，应用程序并不能准确地检测到断开事件（Disconnected），只会出现 suspend，应用程序是无法直接的区分真正的 suspend 和 USB 线断开连接的。当应用程序有数据需要通过 USB 口发送时，如果当前是 suspend 状态，那么它会首先唤醒 USB 总线然后再发送数据：

```
if(hUsbDeviceHS.dev_state ==USBD_STATE_SUSPENDED)
{
      __HAL_PCD_UNGATE_PHYCLOCK((PCD_HandleTypeDef*) hUsbDeviceHS.pData);
      HAL_PCD_ActivateRemoteWakeup((PCD_HandleTypeDef*) hUsbDeviceHS.pData);
```

```
      HAL_Delay(20);
      HAL_PCD_DeActivateRemoteWakeup((PCD_HandleTypeDef*) hUsbDeviceHS.pData);
      (void)USBD_LL_Transmit(&hUsbDeviceHS, HID_EPIN_ADDR, gData, 4);
}
```

而这样发送远程唤醒信号时，device 本身会产生一个 resume 中断，于是在 resume 中断回调函数内:

```
USBD_StatusTypeDef USBD_LL_Resume(USBD_HandleTypeDef *pdev)
{
  if (pdev->dev_state == USBD_STATE_SUSPENDED)
  {
    pdev->dev_state = pdev->dev_old_state;
  }

  return USBD_OK;
}
```

如上所示，程序会将 dev_state 错误地恢复到上一次状态，即正常状态 USBD_STATE_CONFIGURED, 如此一来，程序就错误地往 USB IP 的内的发送 FIFO 写入数据了，即使此时由于 USB 线已经断开而导致无法真正发送成功，但 USB IP 的内置发送 FIFO 此时是有了数据的。

通过调试，查看 OTG_DTXFSTS1 寄存器相应端点 1 对应的发送 FIFO 的剩余空间可知，这个时候的发送 FIFO 的确实有数据的。接下来是 USB 线插上重新枚举，那么为什么 USB 重新枚举后还会再现问题呢？通过设置断点发现，在 USB 成功重新枚举过后，通过 OTG_DTXFSTS1 寄存器指示，发送 FIFO 内容并没有清空，于是在接下来发送数据时，永远都是实际上发送的是上一次写入到 FIFO 中的数据。

问题已经找到，于是解决方法就很容易找到了。在 USB 重新枚举过后在合适的地方将端点 1 对应的发送 FIFO 清空一下即可。

```
USBD_LL_FlushEP(&hUsbDeviceHS,0x81);
```

3 总结

- 在客户的这个案子中，由于 USB_OTG_FS 连接了 VBUS SENSING 脚，当 USB 线拔掉后，会产生正确的 disconnect 中断，USB device 的状态也会正确地切换到 default 状态，从而过滤掉应用程序想要发送的数据，因此并不会出现类似问题，因此，在客户的产品设计中，建议硬件千万不要忘了连接 vbus 引脚，即使在想省 IO 引脚的情况下，这样容易造成对软件的开发诸多不便.
- 在 USB 的状态处于非 configured 状态时，最好不要往发送 FIFO 写入数据，应用程序应该想办法将这些数据过滤掉。

版本历史

日期	版本	变更
2021 年 10 月 27 日	1.0	首版发布