

USART 中断方式接收无响应问题的一种情况及其处理方法

问题:

此问题由客户工程师提出, 客户在使用 STM32F103 的 USART 做串口通讯时, 发现了一个问题, 当设备正常通信一段时间后, 串口不响应外部的通信请求了.

文中配图引用自 STM32F103 的中文版本用户手册, 下载链接为:

索引	中文文档	英文文档	文档版本	软件包	软件版本	更新时间
RM0008	 第10版		15	无		2014-09-12
文档说明: 先进的基于ARM内核的32位微控制器 STM32F101xx、STM32F102xx、STM32F103xx、STM32F105xx和STM32F107xx参考手册 (STM32F101xx, STM32F102xx, STM32F103xx, STM32F105xx and STM32F107xx advanced ARM-based 32-bit MCUs)						

调研:

一、经过调研:

- 1.1 客户除了使用 USART 做串口通信, 还开启了定时器中断来进行数据采集.
- 1.2 定时器的优先级比串口接收的优先级高.
- 1.3 定时器处理数据操作也比较频繁.
- 1.4 客户使用的 STM32F1 标准库(版本 V3.5.0).

二、经过问题复现和使用 ST-LINK 在线调试和定位发现:

- 2.1 在出现这个问题的时候, 程序不断的进入串口接收中断, 不能够运行到 main 主函数处理其他任务了.
- 2.2 发现 ORE 标志为'1', 也就说明程序是发生了串口溢出错误.
- 2.3 客户在进入串口中断后会调用 USART_GetITStatus(USART2, USART_IT_RXNE) 来获取 RXNE 的值. 如果 RXNE 为 1 则去读取 DR 数据寄存器的数据, 读取后 RXNE 为 0, 但是 ORE 的标志依然为 1, 依然进入了串口中断.

三、经过分析以及我们可以通过我们芯片的用户手册可以得到以下信息:

- 3.1 程序不断进入串口中断是因为 ORE 标志始终为 1, 没有被用户清除掉:

位4	IDLE: 监测到总线空闲 (IDLE line detected) 当检测到总线空闲时, 该位被硬件置位。如果USART_CR1中的IDLEIE为'1', 则产生中断。由软件序列清除该位(先读USART_SR, 然后读USART_DR)。 0: 没有检测到空闲总线; 1: 检测到空闲总线。 注意: IDLE位不会再次被置高直到RXNE位被置起(即又检测到一次空闲总线)
位3	ORE: 过载错误 (Overrun error) 当RXNE仍然是'1'的时候, 当前被接收在移位寄存器中的数据, 需要传送到RDR寄存器时, 硬件将该位置位。如果USART_CR1中的RXNEIE为'1'的话, 则产生中断。由软件序列将其清零(先读USART_SR, 然后读USART_CR)。 R0 0: 没有过载错误; 1: 检测到过载错误。 注意: 该位被置位时, RDR寄存器中的值不会丢失, 但是移位寄存器中的数据会被覆盖。如果设置了EIE位, 在多缓冲器通信模式下, ORE标志置位会产生中断的。

从上图红色部分我们可以看到, 如果串口接收中断开启了, 那么 ORE 为 1 时就会产生中断.

3.2. 从下图我们可以看到, 顺序执行对 USART_SR 和 USART_DR 的操作就会清除 ORE 的标志, 客户在中断中执行了以下操作, 那为什么 ORE 标志还没有被清除呢? (R1 执行了读 USART_SR 操作, R2 执行了读 USART_DR 操作).

```
void USARTx_IRQHandler(void)
{
    if (USART_GetITStatus(EVAL_COM1, USART_IT_RXNE) != RESET)
    {
        /* Read one byte from the receive data register */
        RxBuffer[RxCounter++] = USART_ReceiveData(EVAL_COM1) & 0x7F;
    }
}
```

R1
R2

3.3 我们可以看手册中下表的解释:

3.3.1 ORE 表征的是一个历史事件, 当 ORE 为 1 时, 表明至少有 1 个数据已经丢失.

3.3.2 这有 2 种可能发生的情况, RXNE 为 1 的情况 R3 和 RXNE 为 0 的情况 R4, 如下图中的解释.

溢出错误

如果 RXNE 还没有被复位, 又接收到一个字符, 则发生溢出错误。数据只有当 RXNE 位被清零后才能从移位寄存器转移到 RDR 寄存器。RXNE 标记是接收到每个字节后被置位的。如果下一个数据已被收到或先前 DMA 请求还没被服务时, RXNE 标志仍是置起的, 溢出错误产生。

当溢出错误产生时:

- ORE 位被置位。
- RDR 内容将不会丢失。读 USART_DR 寄存器仍能得到先前的数据。
- 移位寄存器中以前的内容将被覆盖。随后接收到的数据都将丢失。
- 如果 RXNEIE 位被设置或 EIE 和 DMAR 位都被设置, 中断产生。
- 顺序执行对 USART_SR 和 USART_DR 寄存器的读操作, 可复位 ORE 位。

注意: 当 ORE 位置位时, 表明至少有 1 个数据已经丢失。有两种可能性:

- 如果 RXNE=1, 上一个有效数据还在接收寄存器 RDR 上, 可以被读出。 R3
- 如果 RXNE=0, 这意味着上一个有效数据已经被读走, RDR 已经没有任何可读。当上一个有效数据在 RDR 中被读取的同时又接收到新的(也就是丢失的)数据时, 此种情况可能发生。在读序列期间(在 USART_SR 寄存器读访问和 USART_DR 读访问之间)接收到新的数据, 此种情况也可能发生。 R4

3.3.3 我们从上图可以看出, 3.2 图中的 R1+R2 操作只能处理 RXNE 为 1 的情况;

当 RXNE 为 0 的特殊情况, 就是在读序列器件(在 USART_SR 寄存器读访问和 USART_DR 读访问之间)接收到新的数据, 数据 SR 虽然被读过了, 但是 overrun 事件依然发生了, 以上操作是不能够处理的:

注意: 当 ORE 位置位时, 表明至少有 1 个数据已经丢失。有两种可能性:

- 如果 RXNE=1, 上一个有效数据还在接收寄存器 RDR 上, 可以被读出。
- 如果 RXNE=0, 这意味着上一个有效数据已经被读走, RDR 已经没有任何可读。当上一个有效数据在 RDR 中被读取的同时又接收到新的(也就是丢失的)数据时, 此种情况可能发生。在读序列期间(在 USART_SR 寄存器读访问和 USART_DR 读访问之间)接收到新的数据, 此种情况也可能发生。 R4

3.4 因此我们要在应用中对 ORE 标志进行处理, 即当判断发生 ORE 中断的时候, 我们再读一次 USART_DR 的值, 这样如果没有新的 Overrun 溢出事件发生的时候, ORE 会被清除, 然后程序就不会因为 ORE 未被清除一直不断的进入串口中断了, 代码如下:

```
void USARTx_IRQHandler(void)
{
    if (USART_GetFlagStatus(EVAL_COM1, USART_FLAG_ORE) != RESET)
    {
        USART_ReceiveData(EVAL_COM1);
    }

    if (USART_GetITStatus(EVAL_COM1, USART_IT_RXNE) != RESET)
    {
```

结论:

对于这种情况，我们可以看到 USART 串口接收中断使能了，那 ORE 中断也就开启了；为了消除在通过读序列(USART_SR/USART_DR)的过程中产生的溢出错误，我们可以尝试在应用程序中需要针对 ORE 标志做一个处理来清除 ORE 标志，使得串口通信可以正常的继续工作。

建议客户在做串口通信时需要增加帧检验功能，如 CRC 校验等...,当发生 ORE 时，帧校验肯定是通不过的，不会造成从机误响应。

处理:

重要通知 - 请仔细阅读

意法半导体公司及其子公司（“ST”）保留随时对ST 产品和/ 或本文档进行变更、更正、增强、修改和改进的权利，恕不另行通知。买方在订货之前应获取关于ST 产品的最新信息。ST 产品的销售依照订单确认时的相关ST 销售条款。

买方自行负责对ST 产品的选择和使用， ST 概不承担与应用协助或买方产品设计相关的任何责任。

ST 不对任何知识产权进行任何明示或默示的授权或许可。

转售的ST 产品如有不同于此处提供的信息的规定，将导致ST 针对该产品授予的任何保证失效。

ST 和ST 徽标是ST 的商标。所有其他产品或服务名称均为其各自所有者的财产。

本文档中的信息取代本文档所有早期版本中提供的信息。