STM32串口通信的三种方式查询、中断、DMA

在STM32处理器中，将发送数据写入USART_DR寄存器，此动作清除TXE（发送允许位）。软件读RXNE位完成对RXNE（接收寄存器非空位）清零。RXNE必须在下一个字符接收结束前清零。

USART的所有中断事件被连接到一个中断向量中，也就是说需要在中断例程中判别各种可能出现的情况。

数据寄存器实际上由两个寄存器组成，一个给发送用（TDR只写），一个给接收用（RDR只读）。和AVR的类似，两个寄存器合并成一个UDR寄存器。

采用中断方式进行串口通信

通过对CodeVersion AVR上的串口通信程序的移植，在STM32上实现了串口数据收发的中断通信。收发各自使用两个循环队列实现文件缓冲，从而提高了执行效率。

队列：一种先进先出（FIFO：First In First Out）的策略。

在向USART写数据时，先检测接收数据寄存器是否“满”，如有数据则写入队列中。当每发送完一帧数据后进入中断程序，检测队列中是否有数据，如有数据则发送，否则退出。USART数据时的情况类似。
需要注意的是在USART_putchar() 和USART_getchar() 函数对缓冲区队列指针操作时需要禁止中断。
www.ec66.com帝国提示，这种为查询方式通信。

Tips：在串口通讯中调用函数USART_GetITStatus(USART1, USART_IT_TC)检测接收是否完成，
函数USART_ClearFlag(USART1,USART_FLAG_TC)清除完成中断标志位，它们操作的都是同一寄存器位USART_CR->TC（状态寄存器中的完成标志位）。
在其它中断中的USRAT_FLAG_xx（标志位）和USART_TI_xx（中断标志位）都表示同一个位。只是为了强调其在特定函数中的作用，而采用不同表述方式。
但USART_ITConfig(USART1, USART_IT_TC, DISABLE)函数中的USART_IT_TC位则是相应的中断允许位，实际是对寄存器USART_CR1->TCIE位操作。

采用DMA方式进行串口通信

STM32串口通信模块支持使用DMA方式进行数据传输，以下代码实现了数据DMA发送方式。当发送完毕产生中断。

程序首先在SRAM中开辟大小为BufferSize的缓冲区SRC_Char_Buffer[]，在main()函数中对其进行初始化。DMA初始化后SRC_Char_Buffer为源地址，USART1_DR_Base（USART1数据寄存器）为目的地址。通过USART_DMACmd()函数设置USART_CR3中的DMAT（允许DMA发送位）。执行DMA_Cmd()函数后使能DMA通道4传输，开始向串口数据寄存器发送数据。每发送一个字节源地址自动加1，总共发送BufferSize个字节。这一过程由DMA控制器完成，无须CPU参与。发送完成后进入中断，中断服务程序CurrDataCounter的值并通过软件设置清除通道全局标志（同时发送完成标志TC自动得到清除）。主程序通过判断CurrDataCounter的值是否为零，决定DMA传输是否结束。为“0”则表示成功，打印相应信息。在实际使用中，CPU可以在数据发送同时执行其它操作。