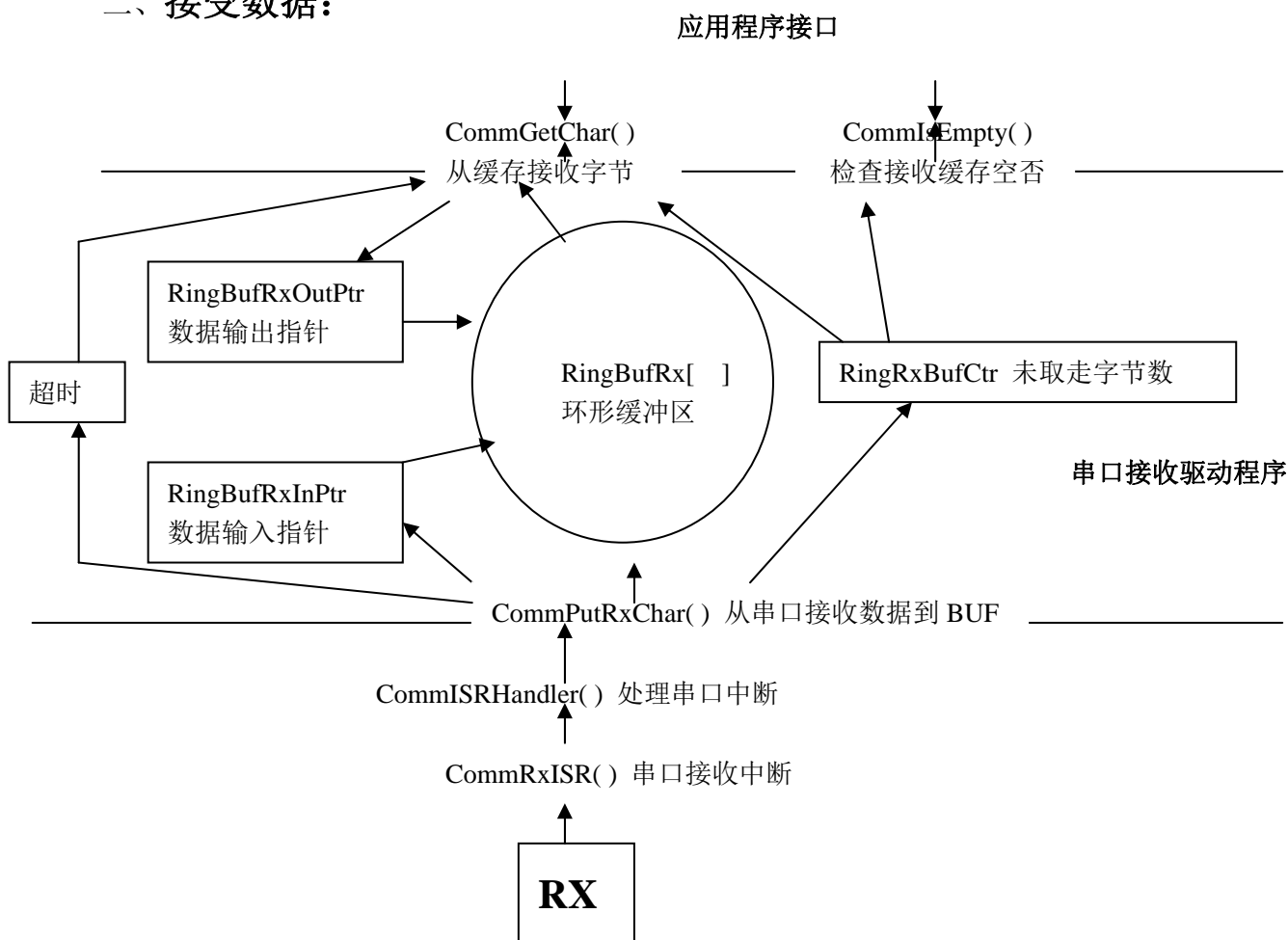


LPC2300 串口驱动程序

一、 UART 串口驱动概述：

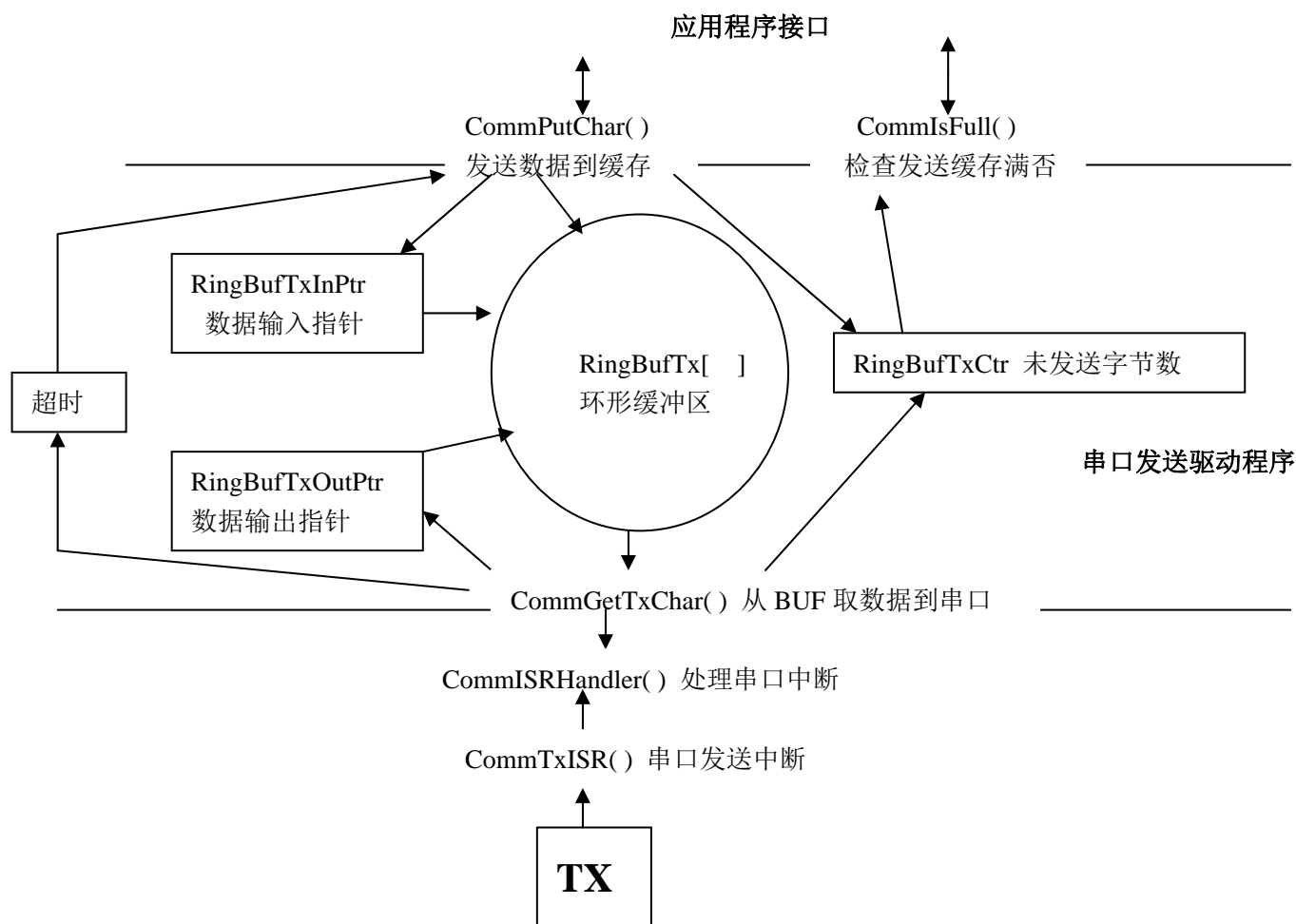
本文介绍用中断驱动模式来接收和传输数据，用中断方式接收数据是比较常见的方法，但发送数据很多人还是用循环发送的方式，这种方式在当 MCU 速度很快，但 UART 发送波特率受限的情况下，是比较浪费时间的，比如 ARM 用 70M 的速度，9600 的波特率，发 100 个字节大约 100MS，如果等发完再做别的工作，会等 100MS,但把 100 字节放入缓冲区让中断自动发送，则只需 10US 就可以完成，然后就可以做其它工作了，所以**中断发送方式，能更有效率。**

二、接受数据：



接收数据容易理解，中断中调用 `CommPutRxChar()`，把收到数据放入接收缓冲区，对应用程序来说，调用 `CommIsEmpty()` 检查接收缓存空否，调用 `CommGetChar()`从缓存接收字节。

三、 发送数据:



应用程序发送数据时,应用程序调用 `CommPutChar()` 把要发送数据放到缓存区, 调用 `CommIsFull()` 检查发送缓存满否。

`TX` 中断调用 `CommGetTxChar()` 从 BUF 取数据发送到串口。

问题的关键是发送第一个字节时, 并没有 `TX` 中断发生, `TX` 中断是发送后才产生。所以第一个字如何发送出去, 如何知道是第一个字节是重点。

看下面的代码, 如果 BUF 中只有一个字节, 则调用 `CommTxIntEn(ch)`; 置 `TX` 中断有效。

这里有两种实现方式:

(1) 发送第一个字节, 发送后会产生 `TX` 中断

(2) 直接置位 `TX` 中断, 如果 MCU 允许这么做。这两种方法前提都必须先判断串口现在不忙时才可以执行这个动作。

```
if (pbuf->RingBufTxCtr == 1) {  
    CommTxIntEn(ch);  
    OS_EXIT_CRITICAL();  
} else {  
    OS_EXIT_CRITICAL();  
}
```

下面是 CommTxIntEn () 的部分代码，用的是第一个字节发送方式：

```
void CommTxIntEn (INT8U ch)  
{  
    INT8U    c,err;  
    c = CommGetTxChar(ch, &err);  
    switch (ch) {  
        case COMM1:  
            OS_EXIT_CRITICAL();  
            while(UART0TxEmpty == 0);  
            OS_ENTER_CRITICAL();  
            U0THR = c;  
            UART0TxEmpty = 0;  
            break;  
    }  
}
```

四、结束语：

用中断方式发送和接收 UART 数据，应用程序只需调用两个函数就可以完成。简单而且提高了效率。