UART接收超时中断

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

代码简述	本范例代码利用UART接收超时中断接收数据。
BSP 版本	NUC100_120BSP_v3.00.004
开发平台	NuTiny-EVB-NUC140 LQFP100 V1.0

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1 功能介绍

1.1 简介

NUC140的UART0的FIFO深度有64个字节，接收数据时，可以设定触发接收中断的FIFO深度门坎值为1/4/8/14/30/46/62 个字节。在Rx FIFO内接收到指定数目的数据后，才触发中断通知CPU处理，避免过于频繁的产生中断，让CPU有足够的时间可以处理其他任务。另外还可以设定接收(time-out)超时中断，当FIFO内数据数目还没有达到门坎值，但是经过一段时间没有再收到新的数据，就产生接收超时中断，通知CPU将FIFO内的资料读走。

1.2 原理

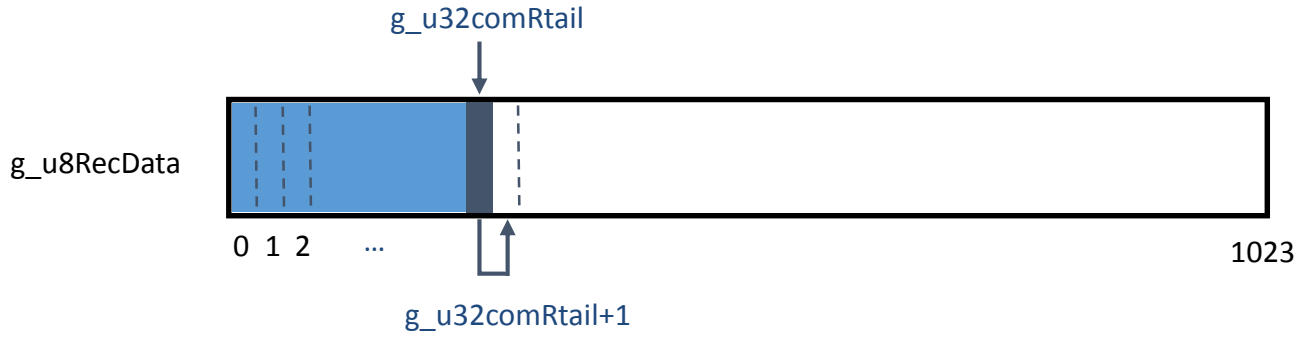
UART接收超时中断

启用UART超时计数器功能后，当Rx FIFO收到数据，计数器便会开始计数，直到计数超过比较值，如果此时Rx FIFO数据还没有被读走，就会产生接收超时中断。计数器的计数时钟为UART的鲍率(baud rate)。假设一个UART数据格式为：一个起始位，8个数据位，一个结束位，则这个UART的一笔数据总共有10个位。当计数器比较值设定为100时，表示其逾时的时间为100个位，换算为10笔UART的数据。

接收中断处理

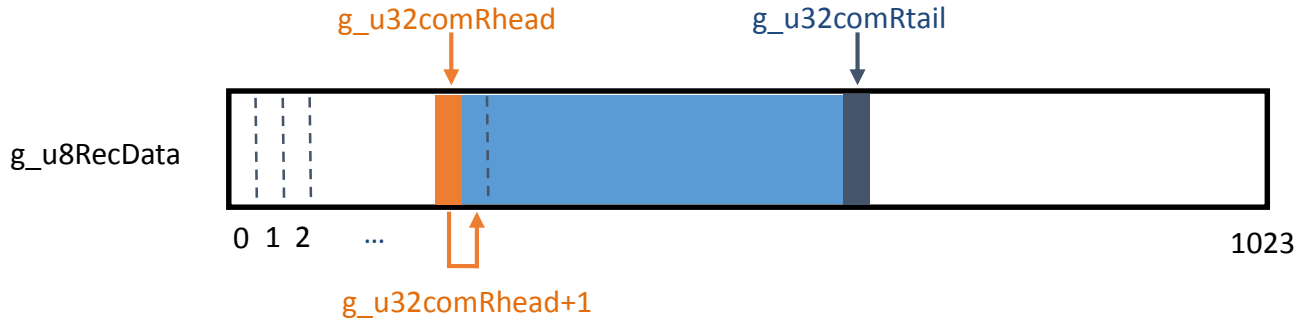
本范例代码宣告一个1024字节的数据数组来储存接收到的UART数据(g_u8RecData)，一个表示数据起始的索引值(g_u32comRhead)和另一个表示数据尾端的索引值(g_u32comRtail)。

当Rx FIFO接收到数据，产生接收中断或超时中断，将该笔数据存到数据数组的最尾端(g_u32comRtail)，并将该索引值加1，直到数据储存了1024笔，再绕回数组最开始的位置储存。



传送中断处理

当Tx FIFO的数据都传完是空的，产生传送中断，从数据数组的最前面读取一笔数据，写到Tx FIFO，并将表示资料起始(g_u32comRhead)的索引值加1，此时终端机打印出从微控制器传送过来的数据。



1.3 执行结果

使用NUC140开发板，PB.0(UART RX)及PB.1(UART TX)连接到NuBridge，计算机端打开终端机接收从UART0打印的讯息。从终端机按键盘传送数据，UART接收到从终端机传送的数据，当该数据是透过接收超时中断接收的，讯息显示”Time-out:”。

```
UART Sample Demo. (Press '0' to exit)

Time-out:1
Transmission Test:1
Time-out:2
Transmission Test:2
Time-out:3
Transmission Test:3
Time-out:0
Transmission Test:0

UART Sample Demo End.
```

2 代码介绍

UART中断设定

设定接收中断的触发门坎值，为4笔数据。

开启超时计数器并设定计数比较值为100。

开启接收、传送以及接收超时中断。

```
    /* Set RX Trigger Level as 4 bytes */
    UART0->FCR = (UART0->FCR & (~UART_FCR_RFITL_Msk)) | UART_FCR_RFITL_4BYTES;

    /* Set Rx time-out count and enable Rx time-out function */
    UART_SetTimeoutCnt(UART0, 100);

    /* Enable RDA, THRE and Time-out Interrupt */
    UART_EnableInt(UART0, (UART_IER_RDA_IEN_Msk | UART_IER_THRE_IEN_Msk |
UART_IER_TOUT_IEN_Msk));
```

UART接收超时中断处理

变数g_u32comRtail指的是存放数据的数组，数据的最尾端。当UART超时中段发生时，从Rx FIFO读取数据并放到储存数据数组中数据的最尾端，并将该索引值加 1。如果数据已经储存到第1024笔(数组的深度)，就回到数组最前面从头开始储存。

```
    /* RX Time-out Interrupt Handle */
    if(u32IntSts & UART_ISR_TOUT_INT_Msk)
    {
         printf("\nTime-out:");

        /* Get all the input characters */
        while( UART_GET_RX_EMPTY(UART0) == 0 )
        {
            /* Get the character from UART Buffer */
            u8InChar = UART_READ(UART0);
            printf("%c ", u8InChar);

            if(u8InChar == '0')
            {
                g_bWait = FALSE;
```

```
            }

            /* Check if buffer full */
            if(g_u32comRbytes < RXBUFSIZE)
            {
                /* Enqueue the character */
                g_u8RecData[g_u32comRtail] = u8InChar;
                g_u32comRtail = (g_u32comRtail == (RXBUFSIZE - 1)) ? 0 : (g_u32comRtail +
1);
                g_u32comRbytes++;
            }
        }
         printf("\nTransmission Test:");
    }
```

UART传送中断处理

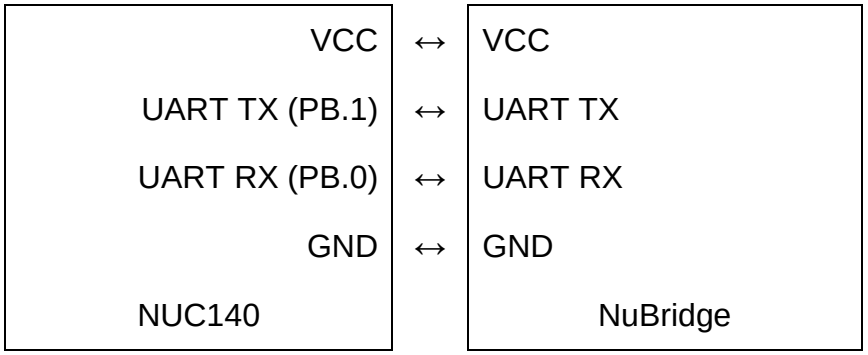
变数g_u32comRhead指的是存放数据的数组，数据的最前端。当发生传送中断时，表示Tx FIFO是空的可以处理数据，则读取数据数组最前端的一笔数据写入Tx FIFO中，处理完数据后同样的将该索引值加1。

```
    /* Transmit Holding Register Empty Interrupt Handle */
    if(u32IntSts & UART_ISR_THRE_INT_Msk)
    {
        uint16_t tmp;
        tmp = g_u32comRtail;
        if(g_u32comRhead != tmp)
        {
            u8InChar = g_u8RecData[g_u32comRhead];
            UART_WRITE(UART0, u8InChar);
            g_u32comRhead = (g_u32comRhead == (RXBUFSIZE - 1)) ? 0 : (g_u32comRhead + 1);
            g_u32comRbytes--;
        }
    }
```

3 软件与硬件环境

- **软件环境**
  - BSP 版本
    - NUC100_120BSP_v3.00.004
  - IDE 版本
    - Keil uVersion 5.28
- **硬件环境**
  - 电路组件
    - NuTiny-EVB-NUC140 LQP100 V1.0
    - NuBridge
  - 示意图

使用 NUC140 开发板，PB.0(UART RX)及 PB.1(UART TX)连接到连接至 NuBridge。



4 目录信息

- EC_NUC140_UART_Rx_Time_out _V1.00
  - Library — Sample code header and source files
    - CMSIS — Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
    - Device — CMSIS compliant device header file
    - StdDriver — All peripheral driver header and source files
  - SampleCode
    - ExampleCode — Source file of example code

5 如何执行范例程序

1. 根据目录信息章节进入 ExampleCode 路径中的 KEIL 文件夹，双击 NUC140_UART_Rx_Time_out_PDWK_INT.uvproj。
2. 进入编译模式接口
   a. 编译
   b. 下载代码至内存
3. 进入除错模式接口
   a. 执行代码

6 修订纪录

Date	Revision	Description
Aug. 31, 2019	1.00	1. 初始发布.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.