UART 接收超時中斷

NuMicro® 32 位系列微控制器範例代碼介紹

文件資訊

代碼簡述	本範例代碼利用 UART 接收超時中斷接收資料。
BSP 版本	NUC100_120BSP_v3.00.004
開發平台	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 功能介紹

1.1 簡介

NUC140的UART0的FIFO深度有64個位元組，接收資料時，可以設定觸發接收中斷的FIFO深度門檻值為1/4/8/14/30/46/62 個位元組。在Rx FIFO內接收到指定數目的資料後，才觸發中斷通知CPU處理，避免過於頻繁的產生中斷，讓CPU有足夠的時間可以處理其他任務。另外還可以設定接收(time-out)超時中斷，當FIFO內資料數目還沒有達到門檻值，但是經過一段時間沒有再收到新的資料，就產生接收超時中斷，通知CPU將FIFO內的資料讀走。

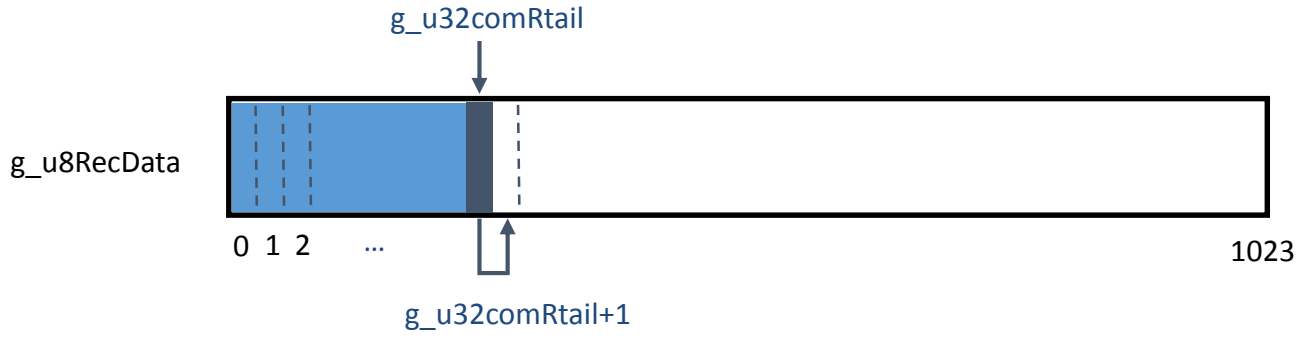
1.2 原理

UART接收超時中斷

啟用UART超時計數器功能後，當Rx FIFO收到資料，計數器便會開始計數，直到計數超過比較值，如果此時Rx FIFO資料還沒有被讀走，就會產生接收超時中斷。計數器的計數時鐘為UART的鮑率(baud rate)。假設一個UART資料格式為：一個起始位元，8個資料位元，一個結束位元，則這個UART的一筆資料總共有10個位元。當計數器比較值設定為100時，表示其逾時的時間為100個位元，換算為10筆UART的資料。

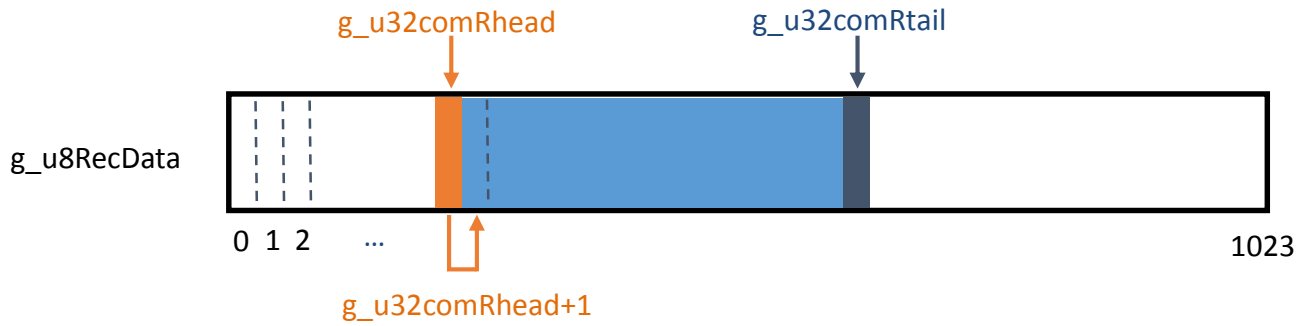
接收中斷處理

本範例代碼宣告一個1024位元組的資料陣列來儲存接收到的UART資料(g_u8RecData)，一個表示資料起始的索引值(g_u32comRhead)和另一個表示資料尾端的索引值(g_u32comRtail)。當Rx FIFO接收到資料，產生接收中斷或超時中斷，將該筆資料存到資料陣列的最尾端(g_u32comRtail)，並將該索引值加1，直到資料儲存了1024筆，再繞回陣列最開始的位置儲存。



傳送中斷處理

當Tx FIFO的資料都傳完是空的，產生傳送中斷，從資料陣列的最前面讀取一筆資料，寫到Tx FIFO，並將表示資料起始(g_u32comRhead)的索引值加1，此時終端機打印出從微控制器傳送過來的資料。



1.3 執行結果

使用NUC140開發板，PB.0(UART RX)及PB.1(UART TX)連接到NuBridge，電腦端打開終端機接收從UART0打印的訊息。從終端機按鍵盤傳送資料，UART接收到從終端機傳送的資料，當該資料是透過接收超時中斷接收的，訊息顯示”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 代碼介紹

UART中斷設定

設定接收中斷的觸發門檻值，為4筆資料。

開啟超時計數器並設定計數比較值為100。

開啟接收、傳送以及接收超時中斷。

```
    /* 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接收超時中斷處理

變數g_u32comRtail指的是存放資料的陣列，資料的最尾端。當UART超時中段發生時，從Rx FIFO讀取資料並放到儲存資料陣列中資料的最尾端，並將該索引值加 1。如果資料已經儲存到第1024筆(陣列的深度)，就回到陣列最前面從頭開始儲存。

```
    /* 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傳送中斷處理

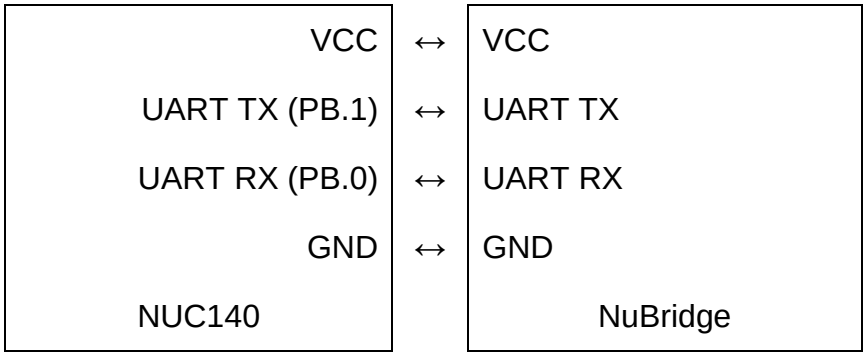
變數g_u32comRhead指的是存放資料的陣列，資料的最前端。當發生傳送中斷時，表示Tx FIFO是空的可以處理資料，則讀取資料陣列最前端的一筆資料寫入Tx FIFO中，處理完資料後同樣的將該索引值加1。

```
    /* 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 軟體與硬體環境

- **軟體環境**
  - BSP 版本
    - NUC100_120BSP_v3.00.004
  - IDE 版本
    - Keil uVersion 5.28
- **硬體環境**
  - 電路元件
    - NuTiny-EVB-NUC140 LQP100 V1.0
    - NuBridge
  - 示意圖

使用 NUC140 開發板，PB.0(UART RX)及 PB.1(UART TX)連接到連接至 NuBridge。



4 目錄資訊

- 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 如何執行範例程式

1. 根據目錄資訊章節進入 ExampleCode 路徑中的 KEIL 資料夾，雙擊 NUC140_UART_Rx_Time_out.uvproj。
2. 進入編譯模式介面
   a. 編譯
   b. 下載代碼至記憶體
3. 進入除錯模式介面
   a. 執行代碼

6 修訂紀錄

Date	Revision	Description
Aug. 31, 2019	1.00	1. 初始發布.

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.