UART Rx Time-out Interrupt

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	This code uses UART Rx time-out interrupt to receive data.
BSP Version	NUC100_120BSP_v3.00.004
Hardware	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 Function Description

1.1 Introduction

The FIFO depth for NUC140 UAR0 is 64 bytes. User can configure the data threshold as 1/4/8/14/30/46/62 bytes to trigger Rx interrupt when receiving data. When the received data in Rx FIFO reaches the threshold configuration, the interrupt is triggered to notify CPU to handle these data. It is used to avoid entering interrupt frequently and let CPU have enough time to process other tasks. There is another Rx time-out interrupt which can be used. When the data number in FIFO does not reach threshold value but no more data is incoming for a while, the Rx time-out interrupt is generated to notify CPU to read these data.

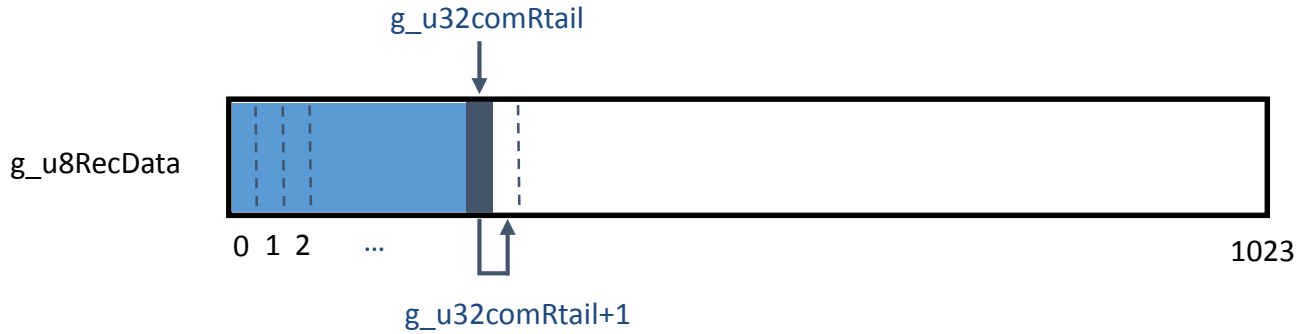
1.2 Principle

UART Rx Time-out Interrupt

If UART Rx time-out counter function is enabled, the counter resets and starts counting whenever the Rx FIFO receives a new data. Once the counter value is equal or over the comparator value and the data in Rx FIFO has not been read, a Rx time-out interrupt is generated. The counting clock is UART baud rate. If an UART data format is 1 start bit, 8 data bits and 1 stop bit, one UART data is 10 bits. When setting comparator value as 100, it means the time-out interval is 100 bits. In other word, it is 10 UART data.

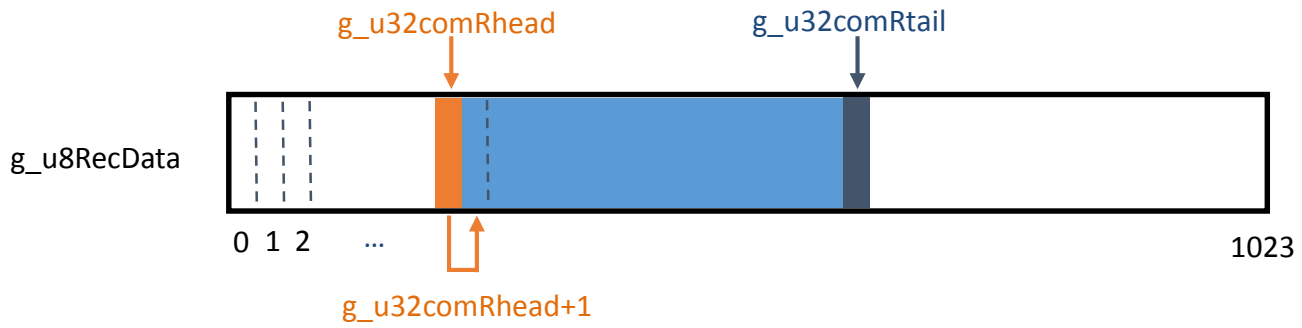
UART Rx Interrupt Handle

This code declares a 1024 bytes data array to store received data (g_u8RecData), an index for the start of data (g_u32comRhead) and an index for the end of data (g_u32comRtail). When data is received in Rx FIFO, the Rx or Rx time-out interrupt is generated. Read these data and store them in the end of data array (g_u32comRtail) and increase the index by 1 until the last position (the 1024th) is used, then store the data back to the beginning of data array.



UART Transmit Interrupt Handle

When Tx FIFO is empty, Tx interrupt is generated, then read one data from the beginning of data array, write to Tx FIFO and increase the start of data index (g_u32comRhead) by 1. Terminal will receive the data and print on the screen.



1.3 Demo Result

Use NUC140 board and connect PB.0(UART RX) and PB.1(UART TX) to NuBridge. Open the terminal in the computer to receive the print message from UART0. Press keyboard from terminal to send data to UART0, too. If UART receives data by Rx time-out interrupt, the message shows “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 Code Description

UART Interrupt Configuration

Configure Rx trigger threshold as 4 bytes data.
Enable Rx time-out function and set the counter comparator value as 100.
Enable Rx, Tx and Rx time-out interrupt.

```
    /* 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 Rx Time-out Interrupt Handle

The variable g_u32comRtail indicates the last data position in the data array to store data. When UART Rx time-out interrupt is generated, read data from Rx FIFO, store in the end of array data and increase the index by 1. If the last position of data array is used, then store the data back to the beginning of array.

```
    /* 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 Transmit Interrupt Handle

The variable g_u32comRhead indicates the start position of stored data in the data array. When UART Tx interrupt is generated, the Tx FIFO is empty and ready for data transmission. Then read one data from the start of data array and write into Tx FIFO. Increase the index by 1 after handling data.

```
    /* 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 Software and Hardware Environment

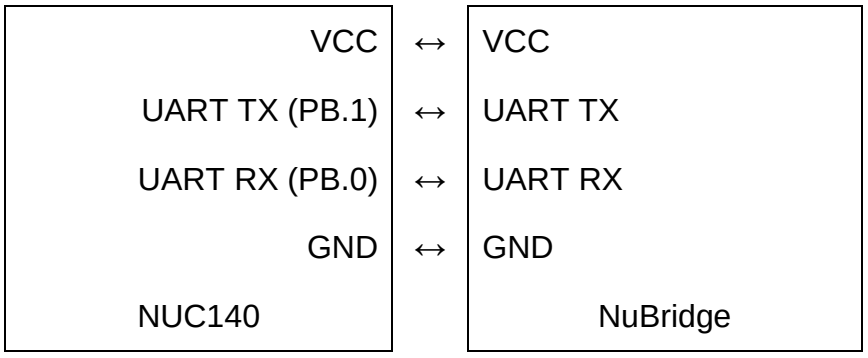
Software Environment

- BSP version
  - NUC100_120BSP_v3.00.004
- IDE version
  - Keil uVersion 5.28

Hardware Environment

- Circuit components
  - NuTiny-EVB-NUC140 LQP100 V1.0
  - NuBridge
- Diagram

NUC140 board PB.0(UART RX) and PB.1(UART TX) is connected to NuBridge.



4 Directory Information

- 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 How to Execute Example Code

1. Browsing into sample code folder by Directory Information (section 4) and double click NUC140_UART_Rx_Time_out.uvproj.
2. Enter Keil compile mode
   a. Build
   b. Download
3. Enter debug mode
   a. Run

6 Revision History

Date	Revision	Description
Aug. 31, 2019	1.00	1. Initially issued.

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.