Software UART

NuMicro® 32 位系列微控制器範例代碼介紹

文件資訊

代碼簡述	使用 GPIO 和 TIMER 模擬 UART 半雙工通訊傳輸
BSP 版本	M051 Series BSP CMSIS v3.01.001
開發平台	NuMaker-EVB-M051 v3.0

1 功能介紹

1.1 簡介

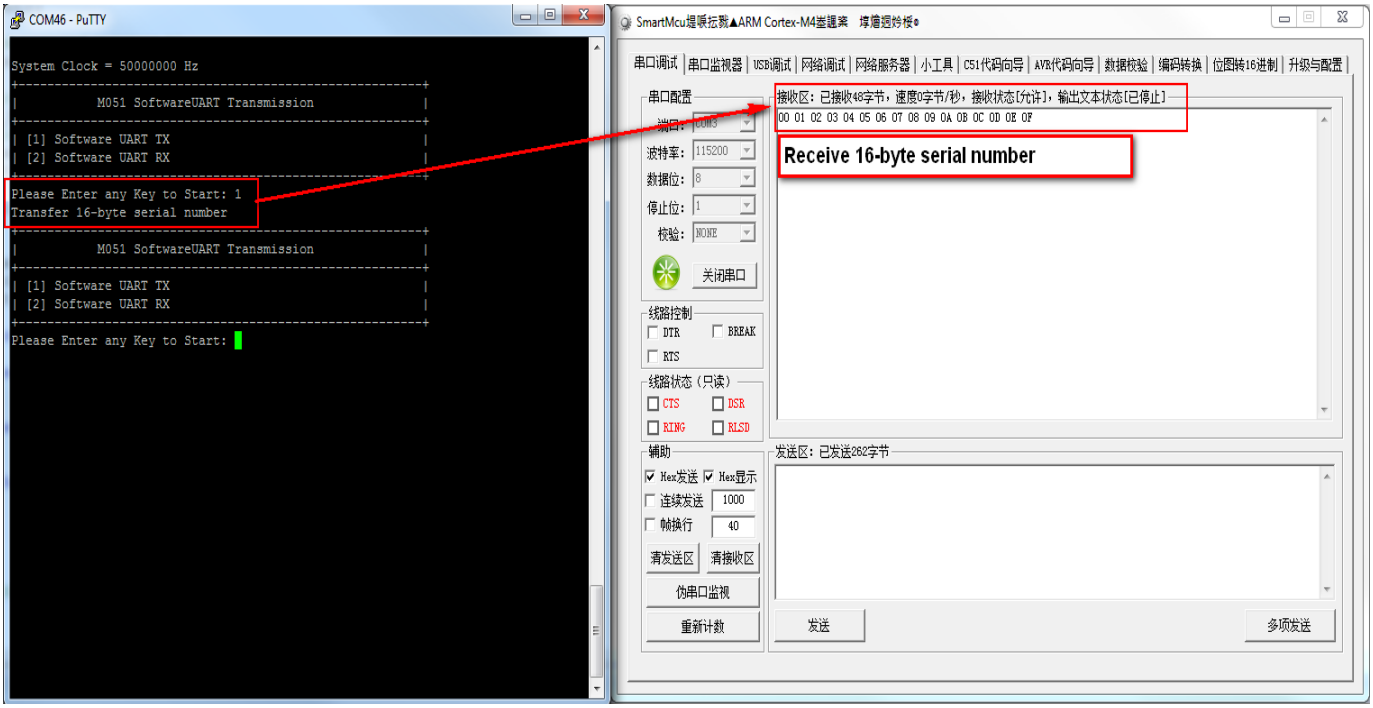
此範例是使用兩根GPIO的IO分別設定為輸出與輸入模式專門負責接收或傳送資料，搭配兩個TIMER計數功能產生週期，來達成模擬 UART_TX 和 UART_RX 的半雙工傳輸功能。

1.2 原理

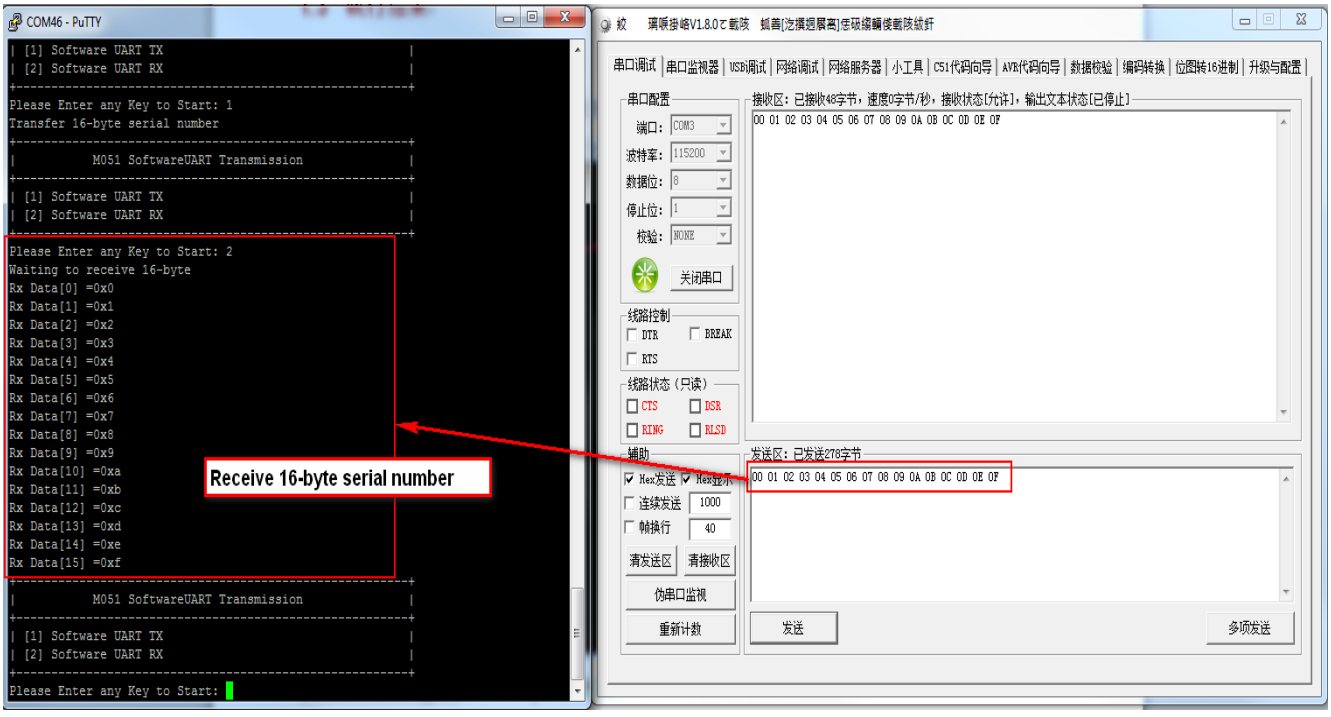
此範例有兩個功能分別是仿真UART 傳送資料 (TX) 和接收資料功能 (RX)，UART TX 部分首先將所要傳送的資料格式組合成所要傳送的位元數(start bit (1) + data bit (5~8) + parity bit (0~1) + stop bit (1~2))，搭配計數器(TIMER1)的週期計數中斷模式，產生所要傳送資料的波特率，再將組合位元組透過GPIO輸出High或Low訊號變化，達成模擬UART傳送資料的通訊傳輸模式。UART RX 部分首先會從通訊格式算出接收的字元數(start bit (1) + data bit (5~8) + parity bit (0~1) + stop bit (1~2))，搭配計數器(TIMER0)的週期計數中斷模式，產生要接收資料的波特率的兩倍速度，透過接收GPIO輸入訊號High或Low變化轉換資料字元組，並對接收完成的字元組會做錯誤驗證和資料轉換，達成模擬UART接收資料的通訊傳輸模式，其中兩倍波特率速度 是為了讓輸入端訊號的擷取點盡量接近中心點位置。

1.3 執行結果

1.3.1 Software UART 傳送資料執行結果



1.3.2 Software UART 接收資料執行結果



2 代碼介紹

2.1.1 Software UART 初始化設定

```
/* Set to use GPIO emulation UART transfer */
#define GPIO_UART_TX       P27
#define GPIO_UART_RX       P26
```

```
void SoftwareUART_Init(sUART_Config *psConfig)
{
        /* Configure P2.6 as Input mode(Rx)*/
        GPIO_SetMode(P2, BIT6, GPIO_PMD_INPUT);
        /* Configure P2.7 as Output mode(Tx)*/
        GPIO_SetMode(P2, BIT7, GPIO_PMD_OUTPUT);
        /*Initial the SoftwareUART configuration setting*/
        SoftwareUART_ConfigInit(psConfig);
        /* Configure SoftwareUART and set SoftwareUART baudrate */
        SoftwareUART_Open(psConfig,115200);
}
```

2.1.2 Software UART 波特率設定

```
void SoftwareUART_Open(sUART_Config *psConfig,uint32_t u32BaudRate)
{
       uint32_t u32TimePeriodic;
       psConfig->u32BaudRate = u32BaudRate;
       /*Setting the UART Rx bit rate*/
       u32TimePeriodic = psConfig->u32BaudRate*2;
       /* Open Timer0 frequency in periodic mode, and enable interrupt */
       TIMER_Open(TIMER0,TIMER_PERIODIC_MODE,u32TimePeriodic);
      /* Enable Timer0 INT */
       TIMER_EnableInt(TIMER0);
      /* Setting the UART Tx bit rate */
       u32TimePeriodic  =  psConfig->u32BaudRate;
       /* Open Timer0 frequency in periodic mode, and enable interrupt */
       TIMER_Open(TIMER1,TIMER_PERIODIC_MODE,u32TimePeriodic);
       /* Enable Timer0 INT */
       TIMER_EnableInt(TIMER1);
                                              .
                                              .
}
```

2.1.3 Software UART 資料格式設定

```
void SoftwareUART_DataFormartConfig(sUART_Config *psConfig)
{

     psConfig->u8DataBits = 8;                /* Setting data length is 8 bits */
     psConfig->u8Parity = PARITY_NONE;        /* No Parity */
     psConfig->u8StopBit = STOP_BIT_1;        /* 1 STOP Bit */
     psConfig->u8TxLatency = 2;               /* Transmission interval is 2 bits rate */
     psConfig->u32SendLen = 16;               /* Send 16 bytes (Tx) */
     psConfig->u32ReceiveLen = 16;            /* Receive 16 bytes (Rx) */
                                              .
                                              .
    /* Configure total communication bits length */
     psConfig->u8CommunBitsLen = psConfig->u8DataBits;

     if(psConfig->u8Pairty != PARITY_NONE)
      psConfig->u8CommunBitsLen += 1;     /* Parity bit */

     if(psConfig->u8StopBit != STOP_BIT_1)
      psConfig->u8CommunBitsLen += 3;     /* Start Bit + 2 Stop Bits */

     else
      psConfig->u8CommunBitsLen += 2;     /* Start Bit + 1 Stop Bit */

}
```

2.1.4 Software UART RX 接收資料

```
void SoftwareUART_ReadData(sUART_Config *psConfig)
{
                                              .
                                              .
       u8RxCount = 0;
      /* Use TIMER0 to emulate the bit rate of UART reception*/
      psConfig->u8RxTiming =0;
      /* Wait to Receive the start bit */
      while(GPIO_UART_RX == 1){};
      psConfig->u8RxStatus = eRxRcvData;

      do{
         while(psConfig->u8RxTiming == 0){};
         /* Receive Data bit */
        if(GPIO_UART_RX == 0)
           psConfig->pu16RxBuff[u32Index] |= (0 << u8RxCount);
        else
           psConfig->pu16RxBuff[u32Index] |= (1 << u8RxCount);
         psConfig->u8RxTiming =0;
         while(psConfig->u8RxTiming == 0){};
         psConfig->u8RxTiming =0;
         u8RxCount++;
     /* Total bits Length (start bit(1)+data bits(5~8)+ parity bit(0-1)+stop bit(1~2))*/
     }while(u8RxCount < psConfig->u8CommunBitsLen);

      u8RxCount =0;
     /* End of reception */
     psConfig->u8RxStatus = eRxIdel;
     psConfig->u8RxTiming =0;

     }
                                              .
                                              .
}
```

```
void TMR0_IRQHandler(void)
{
    /* Clear Timer0 time-out interrupt flag */
    TIMER_ClearIntFlag(TIMER0);
        /* Receiving data time */
    if( g_psSoftwareUART_Config->u8RxStatus == eRxRcvData)
        g_psSoftwareUART_Config->u8RxTiming++;

}
```

2.1.5 Software UART TX 傳送資料

```
void SoftwareUART_SendData(sUART_Config *psConfig)
{
                                                .
                                                .
     psConfig->u8TxStatus = eTxIdel;
     psConfig->u8TxTiming =0;
                                                .
                                                .
    GPIO_UART_TX = 1;
     /* Wait the latency timing */
    u16SendData = psConfig->pu16TxBuff[u32Index];
    while( psConfig->u8TxStatus != eTxActive){};
    psConfig->u8TxStatus = eTxSendData;

   /* Use TIMER1 to emulate the bit rate of UART transmission */
   do{
              psConfig->u8TxTiming =0;
              if((u16SendData & 0x01) == 0x01)
                 GPIO_UART_TX =1;
              else
                 GPIO_UART_TX =0;
              u16SendData = u16SendData >> 1;
              while(psConfig->u8TxTiming == 0){};
     /* Total bits Length(start bit(1)+data bits(5~8)+parity bit(0~1)+stop bit(1~2)) */
   }while(psConfig->u8TxSendCount < (psConfig->u8CommunBitsLen+psConfig->u8TxLatency ));

    psConfig->u8TxStatus = eTxIdel;
    psConfig->u8TxSendCount =0;


                                                .
                                                .
}
```

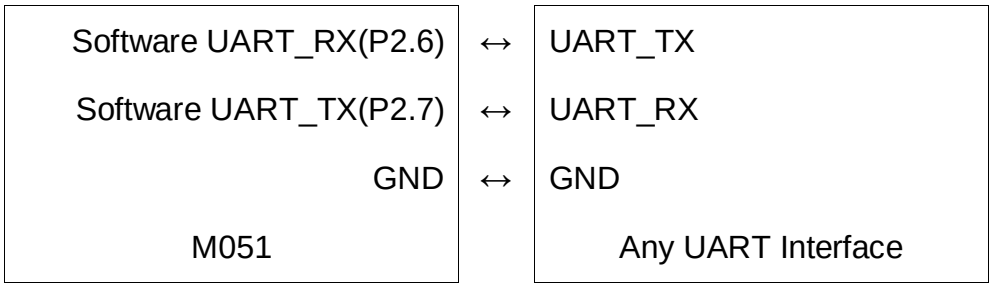
```
void TMR1_IRQHandler(void)
{
    /* Clear Timer1 time-out interrupt flag */
     TIMER_ClearIntFlag(TIMER1);

     g_psSoftwareUART_Config->u8TxSendCount++;
     /*Wait for the Tx transfer start bit to complete*/
     if(g_psSoftwareUART_Config->u8TxStatus == eTxIdel)
     {
        if(g_psSoftwareUART_Config->u8TxSendCount == g_psSoftwareUART_Config->u8TxLatency)
           g_psSoftwareUART_Config->u8TxStatus = eTxActive;
     }/* Tx transmission time */
     else if(g_psSoftwareUART_Config->u8TxStatus == eTxSendData)
              g_psSoftwareUART_Config->u8TxTiming = 1;

}
```

3 軟體與硬體環境

- **軟體環境**
  - BSP 版本
    - M051 Series BSP CMSIS v3.01.001
  - IDE 版本
    - Keil uVersion 4.70
- **硬體環境**
  - 電路元件
    - NuMaker-EVB-M051 v3.0
    - Any UART Interface
    - RS232 to TTL Module
  - 示意圖



4 目錄資訊

- EC_M051_SoftwareUART_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 SoftwareUART.uvproj。
2. 進入編譯模式介面
   a. 編譯
   b. 下載代碼至記憶體
   c. 進入 / 離開除錯模式
3. 進入除錯模式介面
   a. 執行代碼

6 修訂紀錄

Date	Revision	Description
Jul. 5, 2019	1.00	1. 初始發佈.

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.