Software UART

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

代码简述	使用 GPIO 和 TIMER 仿真 UART 半双工通讯传输
BSP 版本	M051 Series BSP CMSIS v3.01.001
开发平台	NuMaker-EVB-M051 v3.0

1 功能介绍

1.1 简介

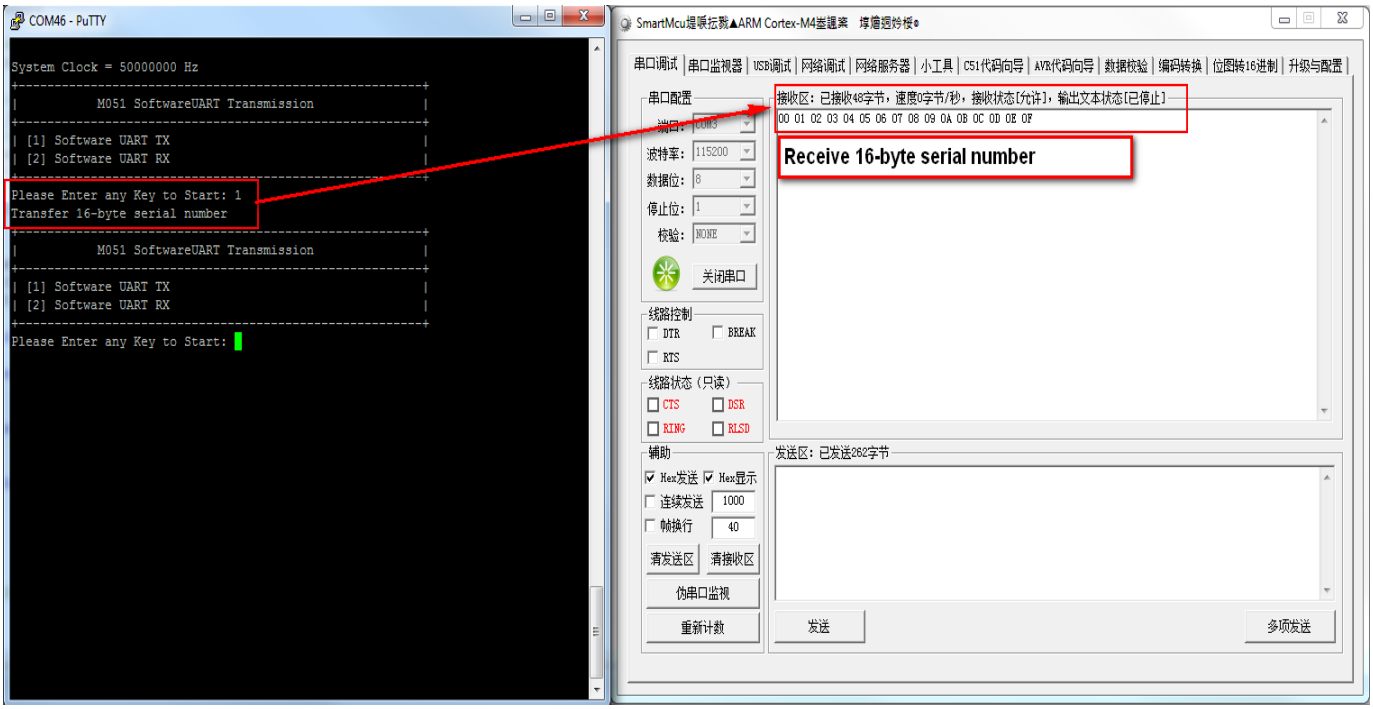
此范例是使用两根GPIO的IO分别设定为输出与输入模式专门负责接收或传送数据，搭配两个TIMER计数功能产生周期，来达成仿真 UART_TX 和 UART_RX 的半双工传输功能。

1.2 原理

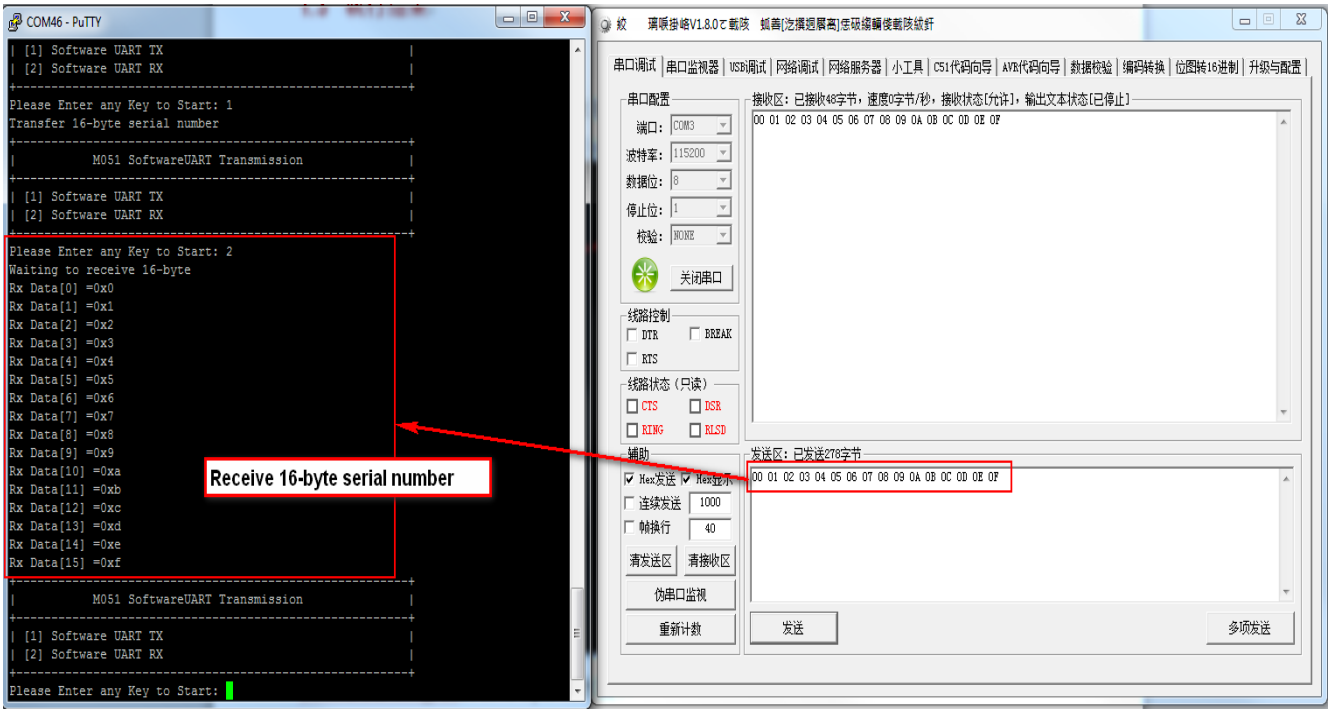
此范例有两个功能分别是仿真UART 传送数据 (TX) 和接收数据功能 (RX)，UART TX 部分首先将所要传送的数据格式组合成所要传送的位数(start bit (1) + data bit (5~8) + parity bit (0~1) + stop bit (1~2))，搭配计数器(TIMER1)的周期计数中断模式，产生所要传送数据的波特率，再将组合字节透过GPIO输出High或Low讯号变化，达成仿真UART传送数据的通讯传输模式。
UART RX 部分首先会从通讯格式算出接收的字符数(start bit (1) + data bit (5~8) + parity bit (0~1) + stop bit (1~2))，搭配计数器(TIMER0)的周期计数中断模式，产生要接收数据的波特率的两倍速度，透过接收GPIO输入讯号High或Low变化转换数据字符组，并对接收完成的字符组会做错误验证和数据转换，达成仿真UART接收数据的通讯传输模式，其中两倍波特率速度 是为了让输入端讯号的撷取点尽量接近中心点位置。

1.3 执行结果

1.3.1 Software UART 传送数据执行结果



1.3.2 Software UART 接收数据执行结果



2 代码介绍

2.1.1 Software UART 初始化设定

```
/* 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 波特率设定

```
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 数据格式设定

```
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 接收数据

```
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 传送数据

```
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;

                                                .
                                                .
}
```

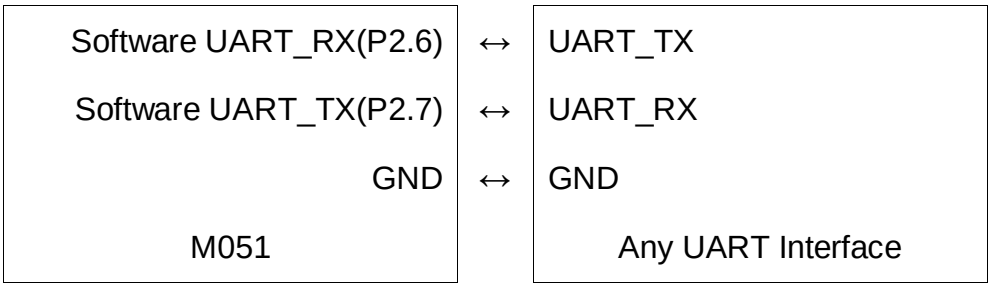
```
void TMR1_IRQHandler(void)
{
    /* Clear Timer1 time-out interrupt flag */
     TIMER_ClearIntFlag(TIMER1);

     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 软件与硬件环境

- 软件环境
  - BSP 版本
    - M051 Series BSP CMSIS v3.01.001
  - IDE 版本
    - Keil uVersion 4.70
- 硬件环境
  - 电路组件
    - NuMaker-EVB-M051 v3.0
    - Any UART Interface
    - RS232 to TTL Module
  - 示意图



4 目录信息

- 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 SoftwareUART.uvproj。
2. 进入编译模式接口
   a. 编译
   b. 下载代码至内存
   c. 进入 / 离开除错模式
3. 进入除错模式接口
   a. 执行代码

6 修订纪录

Date	Revision	Description
Jul. 5, 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.