M032 FreeMODBUS 从机 RTU 模式移植

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

应用简述	此范例代码示范如何将 FreeMODBUS 从机 RTU 模式移植到 M032 系列微控制器。
BSP 版本	M031_Series_BSP_CMSIS_V3.06.00
开发平台	NuMaker-M032SE V1.3

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 and microprocessor 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. 概述

本范例代码展示了如何使用 UART 接口并在 RTU 模式下实现 MODBUS 从机。MODBUS RTU 是主 / 从结构的开放式串行通讯协议。MODBUS RTU 消息编码是 16 进制及具有 CRC 校验封包。

本代码移植 FreeMODBUS 作为 MODBUS 从机实作，FreeMODBUS 是一个针对嵌入式应用免费通用 MODBUS 协议，FreeMODBUS 提供了 RTU，ASCII 传输模式及 TCP 协议支持。FreeMODBUS 遵循 BSD 许可证，这意味着用户可以将 FreeMODBUS 应用于商业环境中。

使用者可以从以下官网链接下载 FreeMODBUS 源代码：

FreeMODBUS Downloads - Embedded Experts (embedded-experts.at)

1.1 原理

MODBUS 协议为主(master)/从(slave)通讯协议，每个从机必须有唯一的地址，最多接 247 个设备，MODBUS 通讯接口可以选用 RS232、RS485 或 Ethernet。其通信遵循以下的过程:

1. 主设备向从设备发送请求。
2. 从设备分析并处理主设备的请求，然后向主设备发送结果。
3. 如果出现任何差错，从设备将返回一个异常功能码。

MODBUS 的传输模式，有以下 3 种：

- ASCII (American Standard Code for Information Interchange, 美国信息交换标准代码)。
- RTU (remote terminal unit, 远程终端单元)。
- TCP (Transmission Control Protocol, 传输控制协议)。

本范例 MODBUS 使用 RTU 模式，使用 16 进制编码，2 个字节的 CRC 校验码，消息中的每个字节包含两个 4 位的 16 进位字符，在同样的波特率下，可比 ASCII 方式传送更多的数据，每两个字节之间发送或者接收的时间间隔不能超过 3.5 倍字符传输时间。

MODBUS RTU 数据框架由四部分组成, 设备地址、功能码、数据和校验，长度如表 1-1。

slave 地址	功能码	数据(依功能码决定)	CRC
1 字节	1 字节	0 ~ 252 字节	2 字节

表 1-1 MODBUS RTU 数据帧格式

表 1-2 列出常用功能码如下表。

功能码	名称	说明
0x1	Read coil Status	取得一组数字输出的目前状态
0x2	Read Input Status	取得一组数字输入的目前状态
0x3	Read Holding Registers	取得一组模拟输出的目前状态
0x4	Read Input Registers	取得一组模拟输入的目前状态

0x5	Force Single Coil	强制设定单个数字输出的值
0x6	Preset Single Register	强制设定单个模拟输出的值
0xF	Force Multiple Coils	强制设定多个数字输出的值
0x10	Preset Multiple Registers	强制设定多个模拟输出的值

表 1-2 MODBUS RTU 常用功能码

slave address/slave ID 设备端 ID，当要读写从机时，必须先知道该设备的 address/ID。常用的 MODBUS 功能码– 0x03。0x03 所定义的功能为读取多个寄存器，用来读取连续地址的数据。以下为 MODBUS RTU 通讯范例：

MODBUS RTU Master 读取格式：
设备 ID(slave address/ID) + 0x03 + 读取起始位置(2 字节) + 读取的数量(2 字节) + CRC16。

MODBUS RTU slave 回复格式：
设备 ID(slave address/ID) + 0x03 + 回复数据的字节数 + 数据 1(1 字节) + … + 资料 n(1 字节) + CRC16。

1.2 执行结果

在测试 MODBUS 协议，可以下载 Modpoll 将个人计算机(PC)作为主控端，使用 Numaker - M032SE 开发板的 NU-LINK2-VCOM 与 M032 的 UART 连接测试。

Modpoll 官网及下载连结：

Free Modbus Master Simulator and Test Tool (modbusdriver.com)

先下载并执行 M032 的 FreeMODBUS 移植代码。在 modpoll 的文件夹下，在 PC 命令行上执行：

modpoll.exe -m rtu -a 10 -r 1000 -c 4 -t 3 -b 38400 -d 8 -p none COM[n, 串口編號]

其 modpoll 參數，請參考官網說明，COM[n] 請設定為 PC 實際使用串口。

```
\modpoll.exe -m rtu -a 10 -r 1000 -c 4 -t 3 -b 38400 -d 8 -p none  COM31
modpoll 3.4 - FieldTalk(tm) Modbus(R) Master Simulator
Copyright (c) 2002-2013 proconX Pty Ltd
Visit http://www.modbusdriver.com for Modbus libraries and tools.

Protocol configuration: Modbus RTU
Slave configuration...: address = 10, start reference = 1000, count = 4
Communication.........: COM31, 38400, 8, 1, none, t/o 1.00 s, poll rate 1000 ms
Data type.............: 16-bit register, input register table

-- Polling slave... (Ctrl-C to stop)
[1000]: 16513
[1001]: 0
[1002]: 0
[1003]: 0
-- Polling slave... (Ctrl-C to stop)
[1000]: 12967
[1001]: 0
[1002]: 0
[1003]: 0
-- Polling slave... (Ctrl-C to stop)
[1000]: 5544
[1001]: 0
[1002]: 0
[1003]: 0
```

图 1-1 执行 modpoll 通讯结果

2. 代码介绍

M032 的 FreeMODBUS 移植需要 M032 的芯片功能有 UART 及 TIMER。在 portserial.c 代码中使用 UART 必须移植 vMBPortSerialEnable、vMBPortClose、xMBPortSerialInit、xMBPortSerialPutByte、xMBPortSerialGetByte 和 UART02_IRQHandler。范例代码如下：

```
/* ----------------------- Start implementation -----------------------------*/
/**
  * @brief      vMBPortSerialEnable
  * @param[in]  BOOL xRxEnable, BOOL xTxEnable
  * @return     None
  * @details    This function to enable/disabele UART TX and RX interrupt.
  */
void  vMBPortSerialEnable(BOOL xRxEnable, BOOL xTxEnable)
{
    __disable_irq();

    if (xRxEnable)
    {
        UART_ENABLE_INT(UART0, UART_INTEN_RDAIEN_Msk);
    }
    else
    {
        UART_DISABLE_INT(UART0, UART_INTEN_RDAIEN_Msk);
    }
    if (xTxEnable)
    {
        UART_ENABLE_INT(UART0, UART_INTEN_THREIEN_Msk);
    }
    else
    {
        UART_DISABLE_INT(UART0, UART_INTEN_THREIEN_Msk);
    }
    NVIC_EnableIRQ(UART02_IRQn);
    __enable_irq();
}

/**
  * @brief      vMBPortClose
  * @param[in]  None
  * @return     None
  * @details    This function to Close UART port.
  */
void vMBPortClose(void)
{
    /* Disable RDA and THRE Interrupt */
    UART_DISABLE_INT(UART0, UART_INTEN_RDAIEN_Msk | UART_INTEN_THREIEN_Msk);
    NVIC_DisableIRQ(UART02_IRQn);
}

/**
  * @brief      xMBPortSerialInit
  * @param[in]  UCHAR ucPORT, ULONG ulBaudRate, UCHAR ucDataBits, eMBParity eParity
  * @return     TURE or FALSE
  * @details    This function to Open UART port and set ulBaudRate, ucDataBits, eParity.
  */
BOOL xMBPortSerialInit(UCHAR ucPORT, ULONG ulBaudRate, UCHAR ucDataBits, eMBParity
eParity)
{
    if (ucPORT != 0) return FALSE;
```

```
    /* Step 1. GPIO initial */
    /* Set PB multi-function pins for UART0 RXD=PB.12 and TXD=PB.13 */
    SYS->GPB_MFPH = (SYS->GPB_MFPH & ~(SYS_GPB_MFPH_PB12MFP_Msk|
                     SYS_GPB_MFPH_PB13MFP_Msk))|\
                    (SYS_GPB_MFPH_PB12MFP_UART0_RXD | SYS_GPB_MFPH_PB13MFP_UART0_TXD);

    /* Step 2. Enable and Select UART clock source*/
    /* Enable UART0 clock */
    CLK_EnableModuleClock(UART0_MODULE);

    /* UART0 clock source */
    /* Select UART module clock source as HIRC and UART module clock divider as 1 */
    CLK_SetModuleClock(UART0_MODULE, CLK_CLKSEL1_UART0SEL_HIRC, CLK_CLKDIV0_UART0(1));

    /* Step 3. Select Operation BaudRate */
    SYS_ResetModule(UART0_RST);
    UART_Open(UART0, ulBaudRate);

    /* Step 4. Set Configure parity*/
    switch (eParity)
    {
        case MB_PAR_NONE:
            UART0->LINE &= ~UART_PARITY_EVEN;
            break;

        case MB_PAR_EVEN:
            UART0->LINE &= ~UART_PARITY_EVEN;
            UART0->LINE |= UART_PARITY_EVEN;
            break;

        case MB_PAR_ODD:
            UART0->LINE &= ~UART_PARITY_EVEN;
            UART0->LINE |= UART_PARITY_ODD;
            break;
    }

    /* Step 5. Set Configure data length*/
    UART0->LINE &= ~UART_WORD_LEN_8;

    switch (ucDataBits)
    {
        case 5:
            UART0->LINE |= UART_WORD_LEN_5;      //5 bits Data Length
            break;

        case 6:
            UART0->LINE |= UART_WORD_LEN_6;      //6 bits Data Length
            break;

        case 7:
            UART0->LINE |= UART_WORD_LEN_7;      //7 bits Data Length
            break;

        case 8:
            UART0->LINE |= UART_WORD_LEN_8;      //8 bits Data Length
            break;
    }

    return TRUE;
}

/**
  * @brief      xMBPortSerialPutByte
  * @param[in]  CHAR ucByte
  * @return     TURE or FALSE
  * @details    This function to send ucByte to UART TX buffer.
  */
```

```
BOOL xMBPortSerialPutByte(CHAR ucByte)
{

    while (UART_GET_TX_FULL(UART0) != 0x00); //check Tx Fifo full.

    UART0->DAT = (uint8_t) ucByte;

    return TRUE;
}
/**
  * @brief      xMBPortSerialGetByte
  * @param[in]  CHAR *pucByte
  * @return     TURE or FALSE
  * @details    This function to get pucByte from UART RX buffer.
  */
BOOL xMBPortSerialGetByte(CHAR *pucByte)
{

    while (UART_GET_RX_EMPTY(UART0) != 0x00); //check Rx Empty

    *pucByte = UART0->DAT;
    return TRUE;
}

/**
  * @brief      UART02_IRQHandler
  * @param[in]  None
  * @return     None
  * @details    This function is UART TX and RX interrupt handler function.
  */
void UART02_IRQHandler(void)
{

    if (UART_GET_INT_FLAG(UART0, UART_INTSTS_RDAINT_Msk))
    {

        /* Get all the input characters */
        while (UART_IS_RX_READY(UART0))
        {
         pxMBFrameCBByteReceived();
        }

    }
    else if (UART_GET_INT_FLAG(UART0, UART_INTSTS_THREINT_Msk))
    {
    pxMBFrameCBTransmitterEmpty();
    }
}
```

在 porttimer.c 代码中使用 TIMER 必须移植代码，xMBPortTimersInit、vMBPortTimersEnable、vMBPortTimersDisable、TMR0_IRQHandler。

```
/**
  * @brief      xMBPortTimersInit
  * @param[in]  USHORT usTim1Timerout50us
  * @return     TURE or FALSE
  * @details    This function to Open Timer0 and time out count(50us unit).
  */
BOOL xMBPortTimersInit(USHORT usTim1Timerout50us)
{
    float freq_exp = 0;
    freq_exp = 1.0 / ((float)usTim1Timerout50us * 0.00005);//convert time to frequence
```

```
    /* Enable Timer0 clock source */
    CLK_EnableModuleClock(TMR0_MODULE);

    /* Select Timer0 clock source as internal 48M */
    CLK_SetModuleClock(TMR0_MODULE, CLK_CLKSEL1_TMR0SEL_HIRC, 0);

    /* Reset IP TMR0 */
    SYS_ResetModule(TMR0_RST);

    /* Open Timer0 in periodic mode, enable interrupt and interrupt tick per second */
    TIMER_Open(TIMER0, TIMER_PERIODIC_MODE, (uint32_t)freq_exp);

    /* Enable Timer0 interrupt */
    TIMER_EnableInt(TIMER0);

    /* Enable Timer0 NVIC */
    NVIC_EnableIRQ(TMR0_IRQn);
    return TRUE;
}

/**
  * @brief      vMBPortTimersEnable
  * @param[in]  None
  * @return     None
  * @details    This function to enable Timer0 and time count reset.
  */
void vMBPortTimersEnable(void)
{
    /* Reset Timer0 counter */
    TIMER_ResetCounter(TIMER0);
    /* Enable Timer0 */
    TIMER_Start(TIMER0);
}

/**
  * @brief      vMBPortTimersDisable
  * @param[in]  None
  * @return     None
  * @details    This function to disable Timer0 and time count reset.
  */
void vMBPortTimersDisable(void)
{
    /* Disable Timer0 */
    TIMER_Stop(TIMER0);
    /* Reset Timer0 counter */
    TIMER_ResetCounter(TIMER0);
}

/**
  * @brief      TMR0_IRQHandler
  * @param[in]  None
  * @return     None
  * @details    This function is Timer0 interrupt handler
  */
void TMR0_IRQHandler(void)
```

```
{
    /* Clear Timer0 interrupt flag */
    TIMER_ClearIntFlag(TIMER0);
    (void)pxMBPortCBTimerExpired();
}
```

在 main 函数中，调用 eMBInit，处理设定 RTU 模式，设定 slave address，UART 开启及设定 baudrate，parity 设定，开启 MODBUS 协议栈，最后在主循环中处理 eMBPoll 判断 MODBUS 封包事件及处理。

```
int32_t main(void)
{
    …
    /* Enable the Modbus RTU mode, MODBUS slave Address, UART port, BaudRate, Parity*/
    eMBInit(MB_RTU, 0x0A, 0, 38400, MB_PAR_NONE);

    /* Enable the Modbus Protocol Stack. */
    eMBEnable();

    for (;;)
    {
        /*eMBPoll for MODBUS event process*/
        (void)eMBPoll();

        /* Here we simply count the number of poll cycles. */
        /* The usRegInputBuf[0] will be accumulated in the main loop, allowing the master
        to read the data changes.*/
        usRegInputBuf[0]++;
    }
}
```

3. 软件与硬件需求

3.1 软件需求

- BSP 版本
  - M032_Series_BSP_CMSIS_V3.06.000
- IDE 版本
  - Keil uVersion 5.28

3.2 硬件需求

- 电路组件
  - NuMaker-M032SE V1.3
- 线路示意图
  - 将NuMaker-M032SE开发板上的Nu-link2-Me的SW1、SW2拨码开关拨到“ON”位置，使能 VCOM，允许主机与 M032 做 MODBUS 通讯。

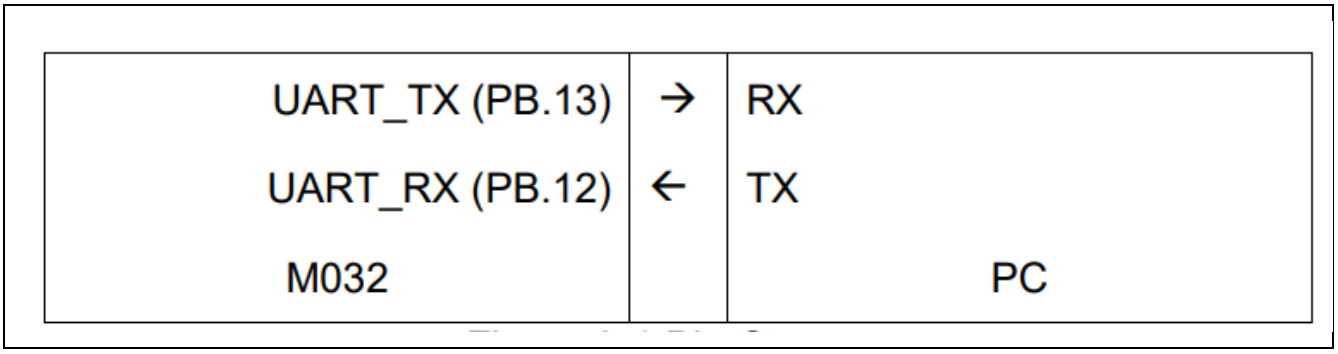

图 3-1 线路示意图

4. 目录信息

EC_M032_FreeMODBUS_RTU_Slave_Porting_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	
Project	Source file of example code

图 4-1 目录信息

5. 范例程式执行

1. 根据目录信息章节进入 ExampleCode 路径中的 KEIL 文件夹，双击 *FreeMODBUS_RTU_Slave_Porting.uvprojx*。
2. 进入编译模式接口
   - 编译
   - 下载代码至内存
   - 进入 / 离开除错模式
3. 进入除错模式接口
   - 执行代码

6. 修订纪录

Date	Revision	Description
2024.05.20	1.00	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.