使用ML51實現LoRa模組資料對傳

NuMicro® 8 位系列微控制器範例代碼介紹

文件資訊

代碼簡述	本範例代碼基於NuMicro ML51晶片中的SPI及GPIO功能，實現LoRa模組資料對傳
BSP 版本	ML51_BSP_Keil_C51_V1.0.0
開發平台	NT-ML51PC V1.1

1 功能介紹

1.1 簡介

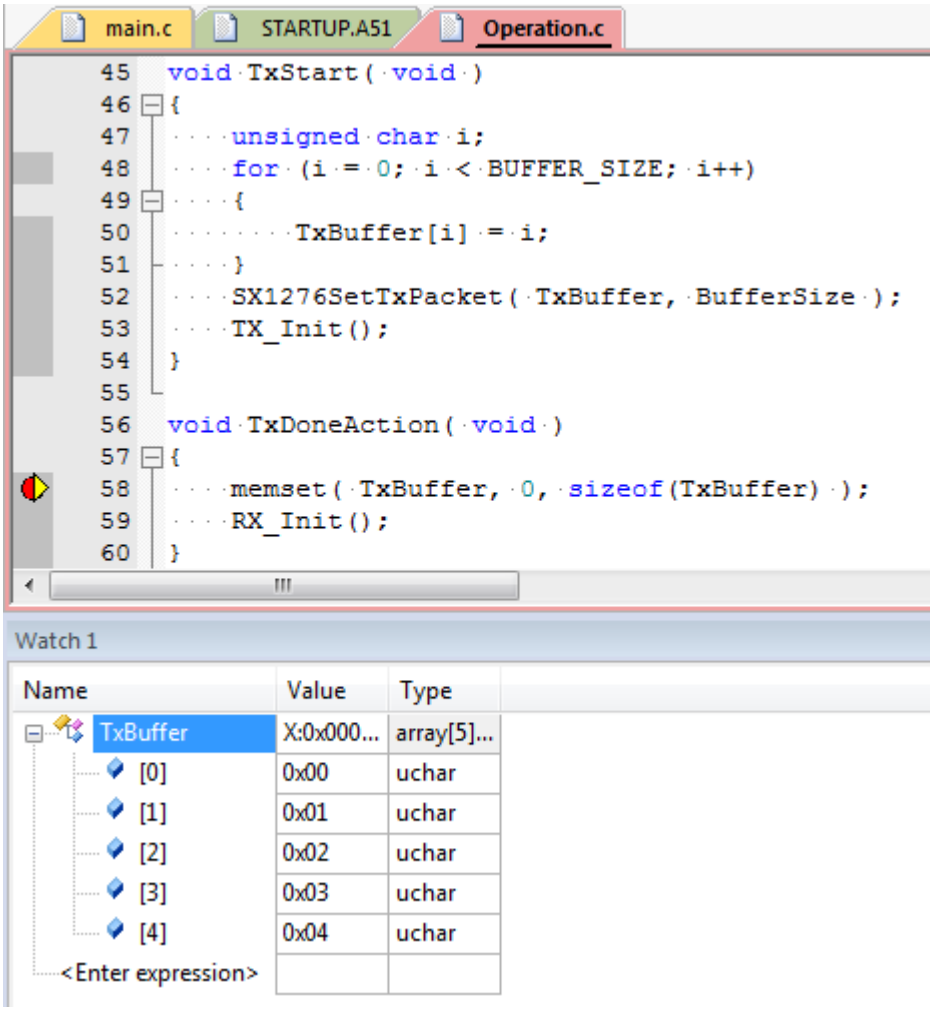
LoRa是LongRange的簡稱，應用於物聯網的低功耗廣域網路傳輸技術(Low Power Wide Area Network, LPWAN)。LoRa無線通訊技術由美商Semtech併購的法商Cycleo所開發，並與IBM合作制定規範，最後由Semtech、Cisco、IBM三大公司作為核心，組成LoRa聯盟推動相關發展，為現今最受產業支持的LPWAN技術。

LoRa的模式如Wi-Fi般，任何人都可以設置基地台來建置網路環境。其具有較高的傳輸頻寬，除了能進行單向傳輸的省電通訊外，也能夠進行數據交換，適合應用於一些較大型的智慧工廠中。除此之外，為了因應不同使用目的，LoRa有Class A, Class B, Class C 3個種類。Class A 做基本的定時傳輸用，強調省電、Class B 除基本傳輸功能外，還增加觸發性傳輸能力、Class C 則提供持續傳輸功能。

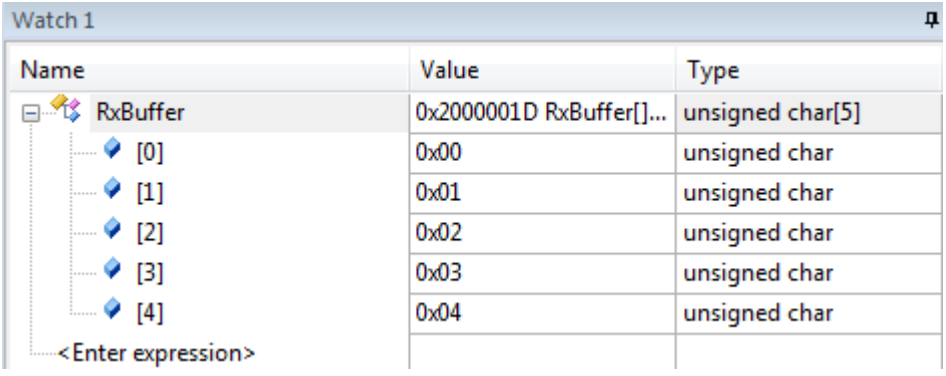
本範例代碼基於NuMicro ML51晶片中的SPI及GPIO功能，實現LoRa模組資料對傳。

1.2 執行結果

a. TX傳送0x00, 0x01, 0x02, 0x03, 0x04，如TxBuffer陣列中的資料



b. RX接收到0x00, 0x01, 0x02, 0x03, 0x04，如RxBuffer陣列中的資料



2 代碼介紹

設定#define TX 1，可以讓ML51控制LoRa模組作為TX傳送資料。

設定#define TX 0，可以讓ML51控制LoRa模組作為RX傳送資料。

範例中先呼叫BoardInit()與Radio_Init()進行GPIO與SPI等硬體的初始化，然後開始進行LoRa資料傳輸測試。

```
#include <stdio.h>
#include "platform.h"
#include "ML51.h"
#include "Operation.h"

/*---------------------------------------------------------------------------------------------------------*/
/*Define                                                                                                   */
/*---------------------------------------------------------------------------------------------------------*/
#define TX 1

/*---------------------------------------------------------------------------------------------------------*/
/*Global variables                                                                                         */
/*---------------------------------------------------------------------------------------------------------*/
extern unsigned char F_RxStart;
extern unsigned char F_TxStart;

/*---------------------------------------------------------------------------------------------------------*/
/* Functions.                                                                                              */
/*---------------------------------------------------------------------------------------------------------*/
int main(void)
{
   BoardInit();
   Radio_Init();

#if TX
    TxStart();
#else
    RX_Init();
#endif

    while (1)
    {
        if (F_RxStart)
        {
            F_RxStart = FALSE;
            RX_Done();
            RxDoneAction();
            RX_Init();
        }
        if (F_TxStart)
        {
            F_TxStart = FALSE;
            TX_Done();
            TxDoneAction();
            TxStart();
        }
    }
}
```

LoRa 接收通過GPIO中斷喚醒，模塊數據通過SPI進行，初始化步驟如下

```
void SX1276InitIo( void )
{
```

```
    /* SPI1 IO define */
    MFP_P32_SPI1_CLK;
    P32_PUSHPULL_MODE;
    MFP_P31_SPI1_MISO;
    P31_INPUT_MODE;
    P31_PULLUP_ENABLE;
    MFP_P30_SPI1_MOSI;
    P30_PUSHPULL_MODE;
    MFP_P33_GPIO;        /*setting SS use gpio mode */
    P33_PUSHPULL_MODE;

    /*Init DIO0 */
    MFP_P00_GPIO;
    P00_INPUT_MODE;
    P00_PULLDOWN_ENABLE;
    GPIO_EnableInt(PIT0, RISING, EDGE, Port0, 0); /*set DIO0 use pin interrupt */
```

```
void SpiInit( void )
{
    unsigned char u8MasterSlave = SPI_MASTER;
    unsigned char u8SPICLKDIV = 9;
    unsigned char u8SPIMode = SPI_MODE_0;
    unsigned char u8MSBLSB = MSB_FIRST;

    SX1276InitIo();
    set_SPI1SR_DISMODF;       /*Mode fault error detection disable*/
    clr_SPI1CR0_SSOE;         /*SS pin use as GPIO*/
    SFRS = 0;
    SPI1CR0 = 0;
    SPI1CR0 |= (u8MasterSlave << 4) | (u8SPICLKDIV & 0x03) | (u8SPIMode << 2) | (u8MSBLSB
<< 5);
    SPI1CR1 = 0 ;
    SPI1CR1 |= (u8SPICLKDIV & 0x0C) << 2;
    set_SPI1CR0_SPIEN;
```

LoRa模塊發送及接收buffer的動作如下

```
void SX1276WriteBuffer( unsigned char addr, unsigned char *buffer, unsigned char size_in )
{
    unsigned char i;

    SPI_SS0_PIN = 0;
    SpiInOut( addr | 0x80 );
    for ( i = 0; i < size_in; i++ )
    {
        SpiInOut( buffer[i] );
    }
    SPI_SS0_PIN = 1;
}

void SX1276ReadBuffer( unsigned char addr, unsigned char *buffer, unsigned char size_in )
{
    unsigned char i;

    SPI_SS0_PIN = 0;
    SpiInOut( addr & 0x7F );

    for ( i = 0; i < size_in; i++ )
    {
        buffer[i] = SpiInOut( 0 );
    }

    SPI_SS0_PIN = 1;
}
```

3 軟體與硬體環境

- 軟體環境
  - BSP 版本
    - ML51_BSP_Keil_C51_V1.0.0
  - IDE 版本
    - Keil uVersion 4.6 及 PK51 Development Kit V9.52
- 硬體環境
  - 硬體需求
    - NT-ML51PC V1.1 tiny board
    - SX1276 LoRa module board
  - 接線圖

NuTiny-SDK-ML51 V1.1		LoRa Module
VCC	↔	VCC
(GPIO INPUT)P00	↔	LORA_DIO0
(GPIO INPUT)P01	↔	LORA_DIO1
(GPIO INPUT)P02	↔	LORA_DIO2
(GPIO INPUT)P03	↔	LORA_DIO3
(GPIO INPUT)P14	↔	LORA_DIO4
(GPIO INPUT)P15	↔	LORA_DIO5
(GPIO OUTPUT)P33	↔	LORA_CS
(SPI CLK)P32	↔	LORA_CLK
(SPI MISO)P31	↔	LORA_MISO
(SPI MOSI)P30	↔	LORA_MOSI
GND	↔	GND

4 目錄資訊

- EC_ML51_LoRa_Control
  - Library: Sample code header and source files.
    - Device: ML51 device header files.
    - Startup: Keil51 startup files
    - StdDriver: All peripheral driver header and source files.
  - Sample_Code
    - Example: Source file of example code.
      - LoRa_Control LoRa module SX1276 TX/RX driver sample code

5 如何執行範例程式

1. 本專案支援Keil uVersion 4.6 及PK51 Development Kit V9.52 以上版本
2. 根據目錄資訊章節進入ExampleCode\LoRa_Control\KEIL資料夾，雙擊 LoRa_Control.uvproj開啟專案
3. 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
4. 進入除錯模式介面
   a. 執行代碼

6 修訂紀錄

日期	版本	描述
Oct. 24, 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.