使用M031來驅動SI4463

NuMicro® 32 位系列微控制器範例代碼介紹

文件資訊

應用簡述	此範例是 M031 通過 SPI 接口來驅動 SI4463 收發數據
BSP 版本	M031 BSP CMSIS V3.03.000
開發平台	NuMaker-M031SD V1.1

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

此範例代碼為M031通過SPI接口來驅動SI4463，實現無線（433M）收發數據。

1.1 原理

Silicon Laboratories的Si446x器件是高性能、低電流功耗收發器，覆蓋119到1050MHz的sub-GHz頻段。该器件都具有-126dBm的突出靈敏度，同時實現極低的工作和待機電流消耗。Si446x通過標準的4線SPI：SCLK，SDI，SDO和nSEL於主機MCU通訊，最大通訊速率可以達到10Mbps。

主機MCU通過SDI引腳寫數據，並可以通過SDO引腳從設備讀取數據。圖1-1為一個SPI寫命令時序。nSEL引腳為低來啟動一個SPI傳輸。SDI資料的第一個數據將是固件命令之一，後面是n位數據的參數資料，參數資料將根據特定的命令而變化。SCLK的上升沿應與SDI數據位的中心對齊。

SPI通訊寫時序：

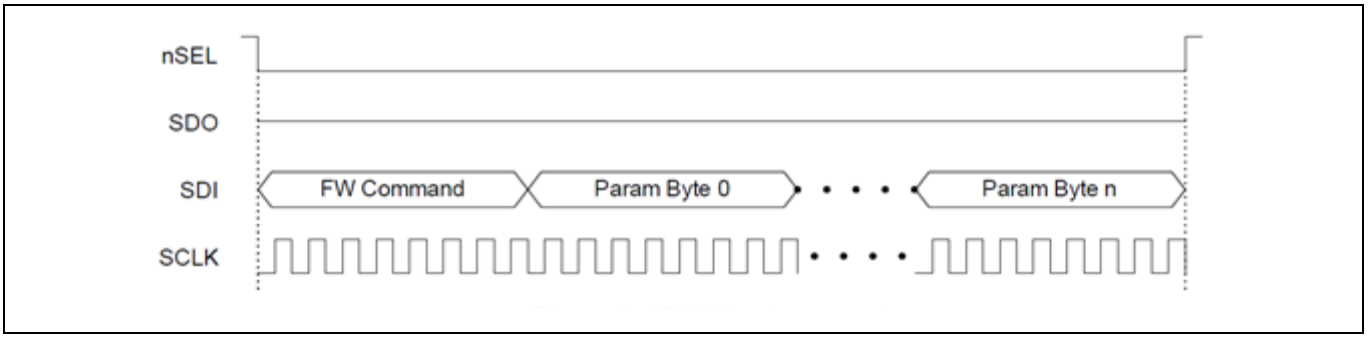

圖 1-1 SPI 寫命令時序

Si446x內含一個MCU，來控制RADIO的所有內部功能。對於SPI讀命令，先檢查流控命令CTS，以確保內部MCU已經執行了命令，並準備了要在SDO引腳上輸出的資料。圖1-2為SPI讀命令的流程。一旦CTS值讀取為FFh，那麼表示需要讀取的資料已準備好可以被時鐘輸出到主機MCU。有效FFh值CTS讀取的典型時間是20us。圖1-3演示了讀取CTS為FFh後剩餘的讀取週期時序。內部MCU將在時鐘下降沿從SDO輸出資料，因此主機MCU應該在SCLK的上升沿上處理SDO資料。

SPI通訊讀時序及流程：

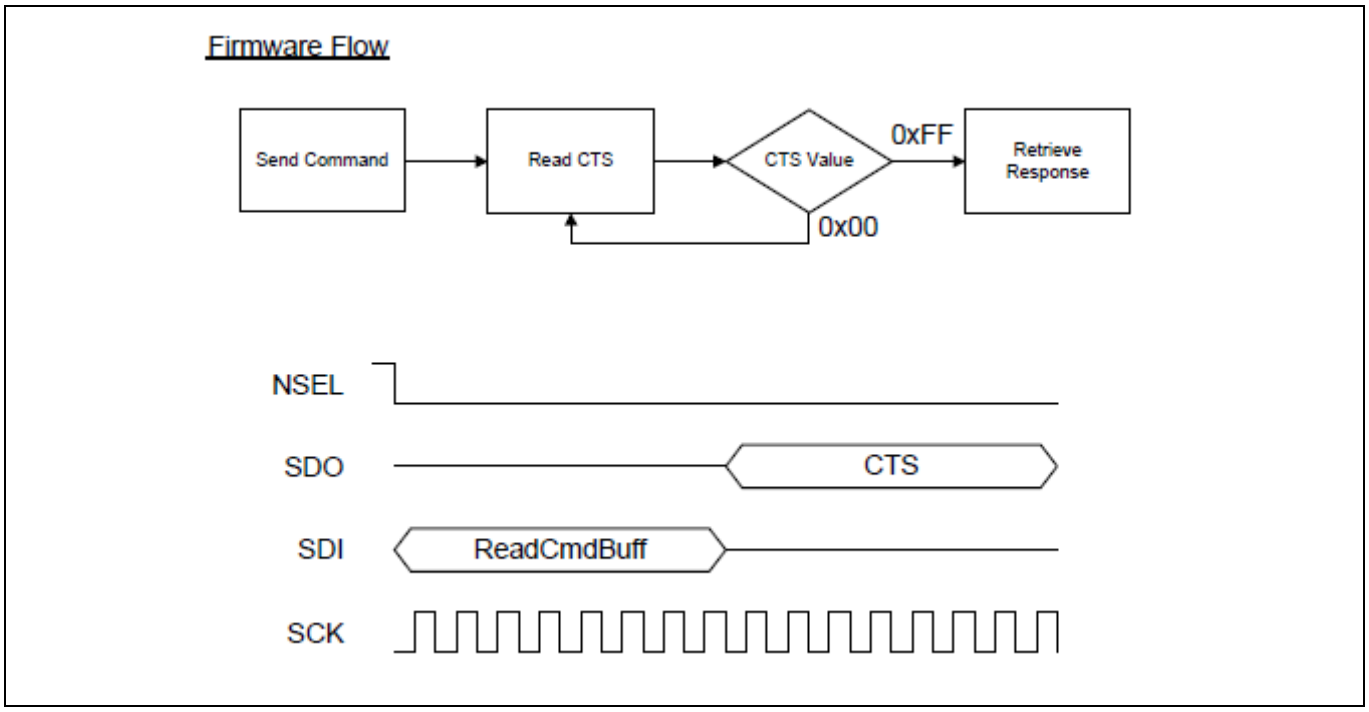

圖 1-2 SPI 讀命令-檢查 CTS 值

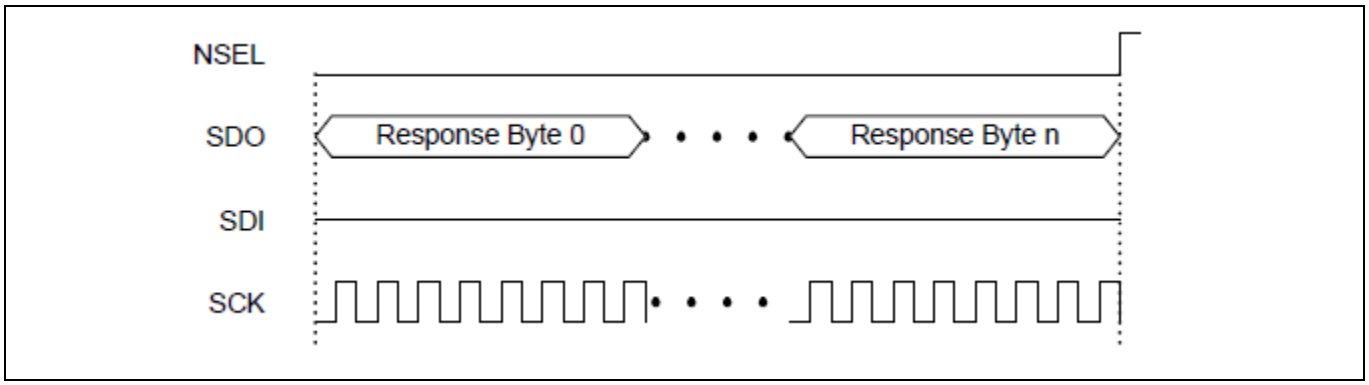

圖 1-3 SPI 讀命令-讀數據

以上圖1-1，圖1-2和圖1-3來源於SI4463數據手冊。

關於SI4463命令，寄存器等詳細資料可以參考以下文檔：
https://www.silabs.com/documents/public/data-sheets/Si4463-61-60-C.pdf

1.2 執行結果

通過主機/從機終端，相互傳遞數據，結果如下：

作為主機，SI4463發送數據包並接收ACK：

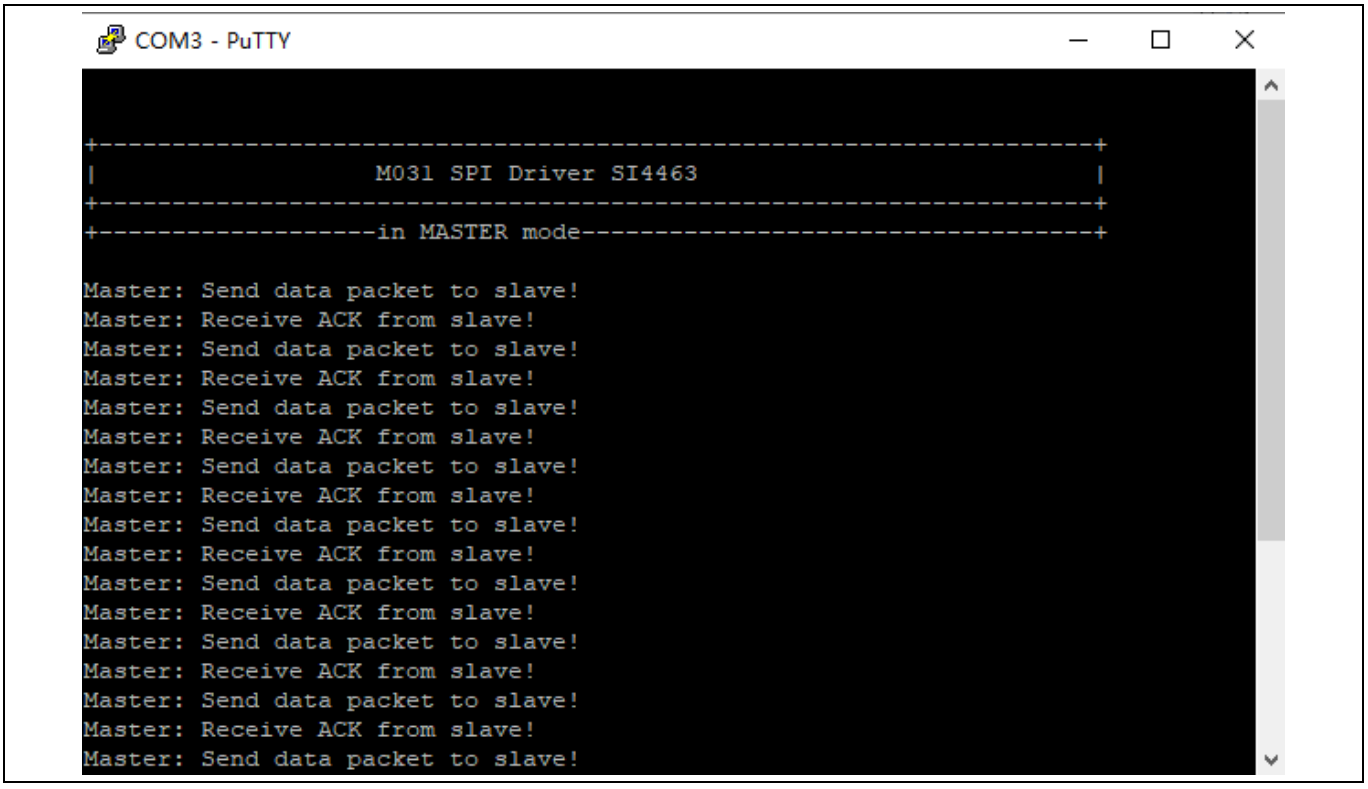

圖 1-4 主機通訊結果

作為從機，SI4463接收數據包並發送ACK:

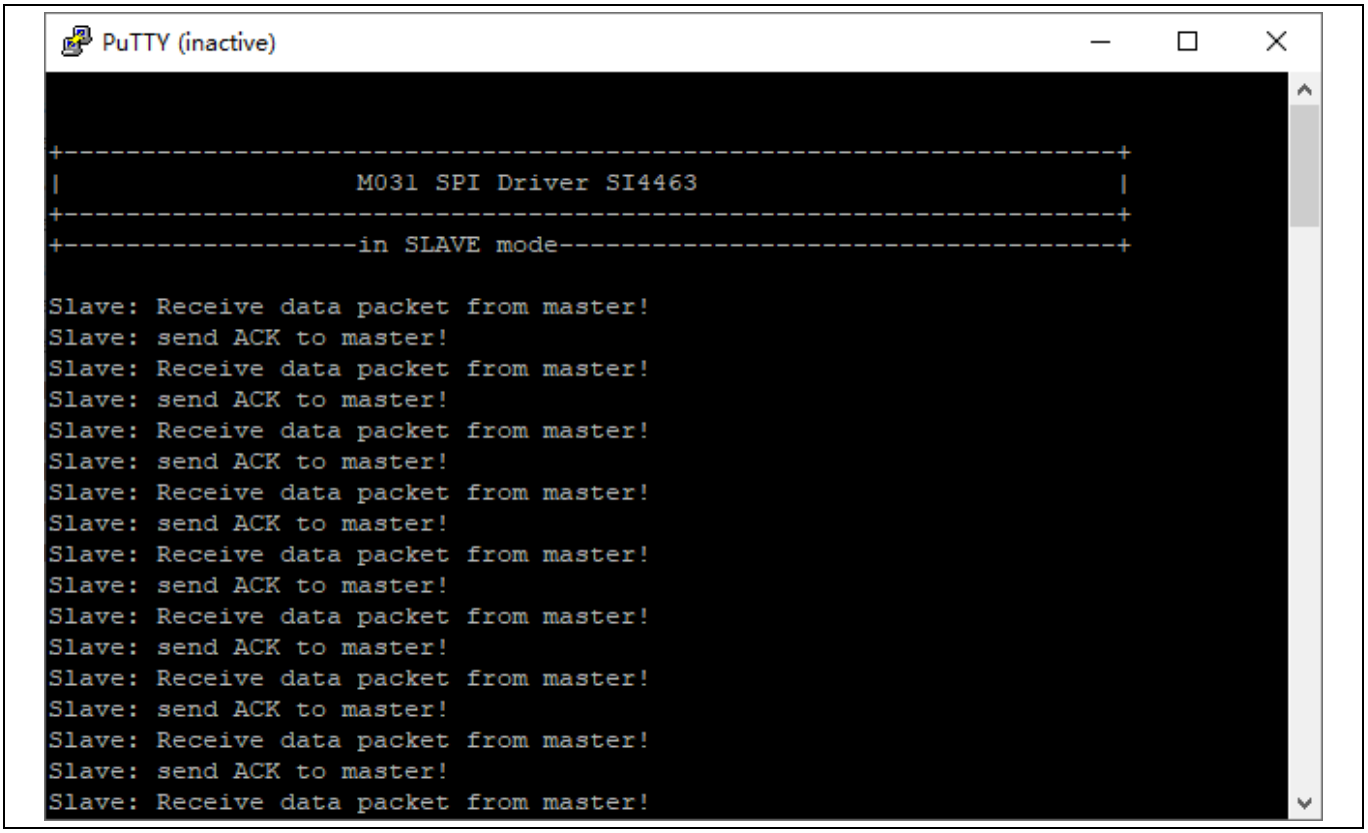

圖 1-5 從機通訊結果

2 代碼介紹

設置主機從機：

```
#define EnableMaster   1          //in master mode
//#define EnableMaster   0        //in slave mode
```

初始化 SI4463：

```
void vRadio_Init(void)
{
   uint16_t wDelay;

   /* Power Up the radio chip */
   vRadio_PowerUp();

   /*get  part information6 */
   si446x_part_info();

   /*part information check */
   if (Si446xCmd.PART_INFO.PART != 0x4463)
   {
      while (1)
      {
         /* ERROR!!!!  SPI communicate error  or part information error */
#ifdef RADIO_COMM_ERROR_CALLBACK
         RADIO_COMM_ERROR_CALLBACK();
#endif
      }
   }

   /* Load radio configuration */
   while (SI446X_SUCCESS != si446x_configuration_init(pRadioConfiguration->Radio_ConfigurationArray))
   {
      for (wDelay = 0x7FFF; wDelay--;) ;

      /* Power Up the radio chip */
      vRadio_PowerUp();
   }

   /* Read ITs, clear pending ones  */
   si446x_get_int_status(0, 0, 0);
}
```

SI4463 收發狀態切換及處理：

```
uint8_t bRadio_Process()
{
    switch (RFstate)
    {
        case RFLR_STATE_RX_INIT:
            radio_TX(0);
            radio_RX(1);
```

```
            vRadio_StartRX(pRadioConfiguration->Radio_ChannelNumber, pRadioConfiguration-
>Radio_PacketLength);
            TickCounter = 0;
            RFstate = RFLR_STATE_RX_RUNNING;
            break;


        case RFLR_STATE_RX_RUNNING:
            //Delay_Us(300);
            //si446x_get_modem_status_fast_clear_read();//RSSI
            bMain_IT_Status = bRadio_Check_Tx_RX();


            if (bMain_IT_Status ==
SI446X_CMD_GET_INT_STATUS_REP_PH_PEND_PACKET_RX_PEND_BIT)
            {
                if (EnableMaster)
                {
                    RFstate = RFLR_STATE_RX_ACK_DONE;
                }
                else
                {
                    RFstate = RFLR_STATE_RX_DONE;
                }
            }

            if (TickCounter > 2000)
            {
                RFstate = RFLR_STATE_RX_TIMEOUT;
            }
            break;


        case RFLR_STATE_TX_INIT:
            radio_TX(1);
            radio_RX(0);

            gSampleCode_SendVariablePacket();
            TickCounter = 0;
            RFstate = RFLR_STATE_TX_RUNNING;
            break;


        case RFLR_STATE_TX_ACK_INIT:
            radio_TX(1);
            radio_RX(0);
```

```
            printf("Slave: send ACK to master! \n") ;
            vSampleCode_SendAcknowledge();
            TickCounter = 0;
            RFstate = RFLR_STATE_TX_RUNNING;
            break;

        case RFLR_STATE_TX_RUNNING:
            bMain_IT_Status = bRadio_Check_Tx_RX();

            if (bMain_IT_Status ==
SI446X_CMD_GET_INT_STATUS_REP_PH_PEND_PACKET_SENT_PEND_BIT)
            {
                if (EnableMaster)
                {
                    RFstate = RFLR_STATE_TX_DONE;
                }
                else
                {
                    RFstate = RFLR_STATE_TX_ACK_DONE;
                }
            }

            if (TickCounter > 2000)
            {
                RFstate = RFLR_STATE_TX_TIMEOUT;
            }
            break;

        case RFLR_STATE_RX_DONE:
            if (gSampleCode_StringCompare(customRadioPacket, tx_data, sizeof(tx_data)) ==
true)
            {
                RFstate = RFLR_STATE_TX_ACK_INIT;
                printf("Slave: Receive data packet from master! \n") ;
                HAL_Delay_nMS(200);
            }
            else
            {
                RFstate = RFLR_STATE_RX_INIT;
            }
            break;
```

```
        case RFLR_STATE_RX_ACK_DONE:
            if (gSampleCode_StringCompare(customRadioPacket, ACK_data, 3) == true)
            {
                printf("Master: Receive ACK from slave! \n") ;
            }

            HAL_Delay_nMS(100);
            RFstate = RFLR_STATE_TX_INIT;
            break;

        case RFLR_STATE_TX_DONE:
            printf("Master: Send data packet to slave! \n") ;
            RFstate = RFLR_STATE_RX_INIT;
            break;

        case RFLR_STATE_TX_ACK_DONE:
            HAL_Delay_nMS(20);
            RFstate = RFLR_STATE_RX_INIT;
            break;

        case RFLR_STATE_TX_TIMEOUT:
            if (EnableMaster)
            {
                RFstate = RFLR_STATE_TX_INIT;
            }
            else
            {
                RFstate = RFLR_STATE_RX_INIT;
            }
            break;

        case RFLR_STATE_RX_TIMEOUT:
            if (EnableMaster)
            {
                RFstate = RFLR_STATE_TX_INIT;
            }
            else
            {
                RFstate = RFLR_STATE_RX_INIT;
            }
```

```
            break;

        default:
            break;
    }

    return RFstate;
}
```

3 軟體與硬體需求

- **軟體需求**
  - BSP 版本
    - M031_Series_BSP_CMSIS_V3.03.000
  - IDE 版本
    - Keil uVersion 5.26
- **硬體需求**
  - 電路元件
    - NuMaker-M031SD V1.1
    - SI4463 模塊
  - 引腳連接

硬件電路使用M031連接SI4463模塊，電路如下圖3-1：

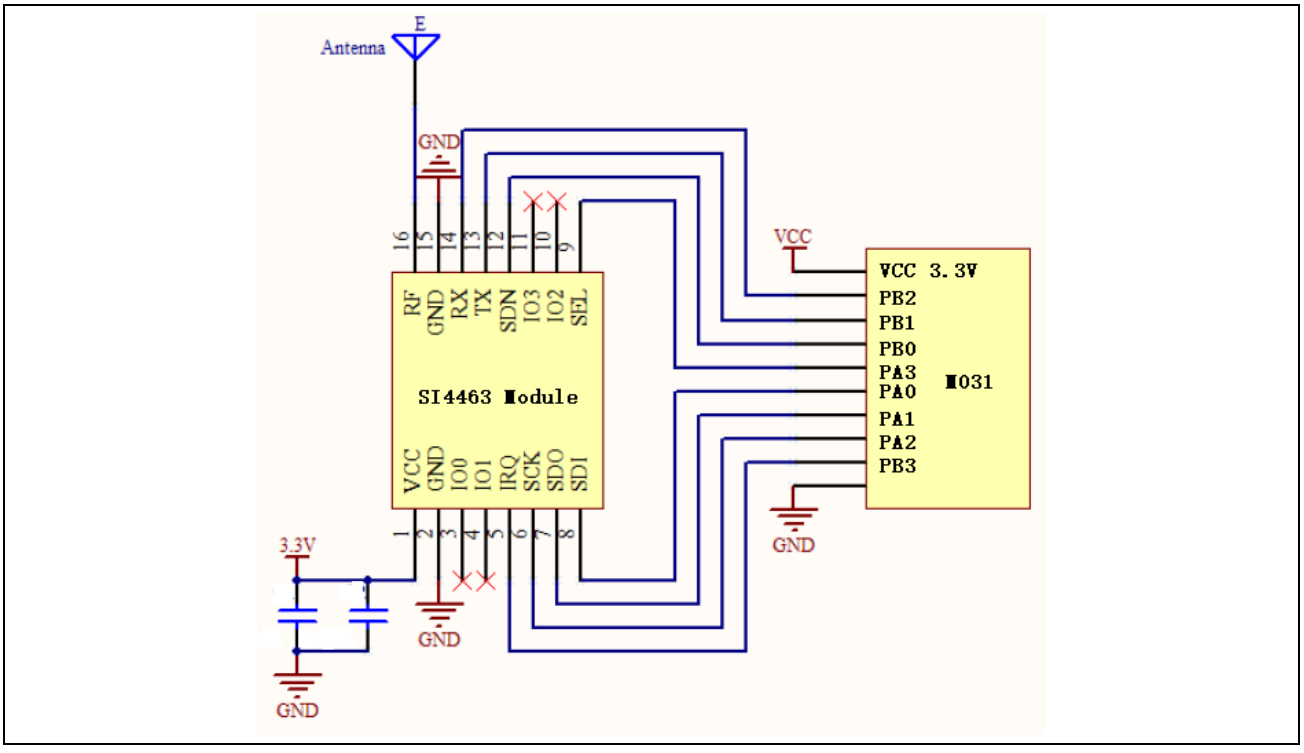

圖 3-1 引腳連接

SI4463模塊相關引腳功能說明如下表3-1：

引腳名	I/O	描述
IO0	I/O	通用數字 I/O
IO1	I/O	通用數字 I/O
IRQ	O	中斷狀態輸出 當 Si446x 發生任何中斷事件，IRQ 引腳將置低
SCK	I	SPI 時鐘輸入
SDO	O	SPI MISO 輸出
SDI	I	SPI MOSI 輸入
SEL	I	SPI 片選信號輸入
IO2	I/O	通用數字 I/O
IO3	I/O	通用數字 I/O
SDN	I	關閉輸入引腳。 除 Shutdown 模式外的所有模式下，SDN 都應該是低。 當 SDN=1 時，晶片將完全關閉，寄存器內容將會丟失。
TX	I	發送輸出控制端。發送置高時 RX 必須為低
RX	I	接收輸入控制端。接收置高時 TX 必須為低
RF	I/O	RF 信號輸入/輸出, 連接 50Ω天線

表 3-1 引腳描述

4 目錄資訊

EC_M031_SPI_SI4463_V1.00	
Library	Sample code header and source files
CMSIS	Cortex®Microcontroller Software Interface Standard (CMSIS) V4.5.0 definitions by ARM® Corp.
Device	CMSIS compliant device header file
StdDriver	All peripheral driver header and source files
SampleCode	
ExampleCode	Source file of example code

圖 4-1 目錄資訊

5 範例程式執行

1. 通過目錄資訊章節進入 ExampleCode 路徑中的 KEIL 資料夾，點擊兩下 M031_SPI_SI4463.uvproj
2. 進入編譯模式介面：
   a. 編譯
   b. 下載
   c. 進入/離開仿真模式
3. 進入仿真模式介面
   a. 執行代碼

6 修訂紀錄

Date	Revision	Description
2021.06.25	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.

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.