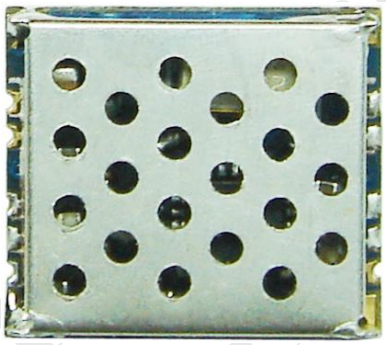
TWR-811S RF MODULE

Wireless Low Cost 900MHz RF Transceiver Module



Version History

Version	Date	Changes
V1.01	Aug 4, 2008	1st. Edition

Application

The RF transceiver is integrated with a highly configurable baseband modem. The modem supports various modulation formats and has a configurable data rate up to 250 Kbps. The communication range can be increased by enabling a Forward Error Correction option, which is integrated in the modem.
TRW-811S provides extensive hardware support for packet handling, data buffering, burst transmissions, clear channel assessment, link quality indication and wake-on-radio.
The main operating parameters and the 64-byte transmit/receive FIFOs of TRW-811S can be controlled via an SPI interface. In a typical system, the TRW-811S will be used together with a microcontroller and a few additional passive components.

Specification

● Remote Control Systems	● Wireless Data Transceiver
● 300MHz~928MHz ISM/SRD Band	● Remote Metering
● Wireless Security Systems	● Automatic Meter Reading
● Application Range：Remote Metering、Wireless Security Systems 、Automatic Meter 、Reading、Home Automation	

Key Feature

- Small size
- Separate 64-byte RX and TX data FIFOs
- Efficient SPI interface: All registers can be programmed with one "burst" transfer
- Programmable output power up to +10dBm
- High sensitivity (-111 dBm at 1.2 Kbps, 1% packet error rate)

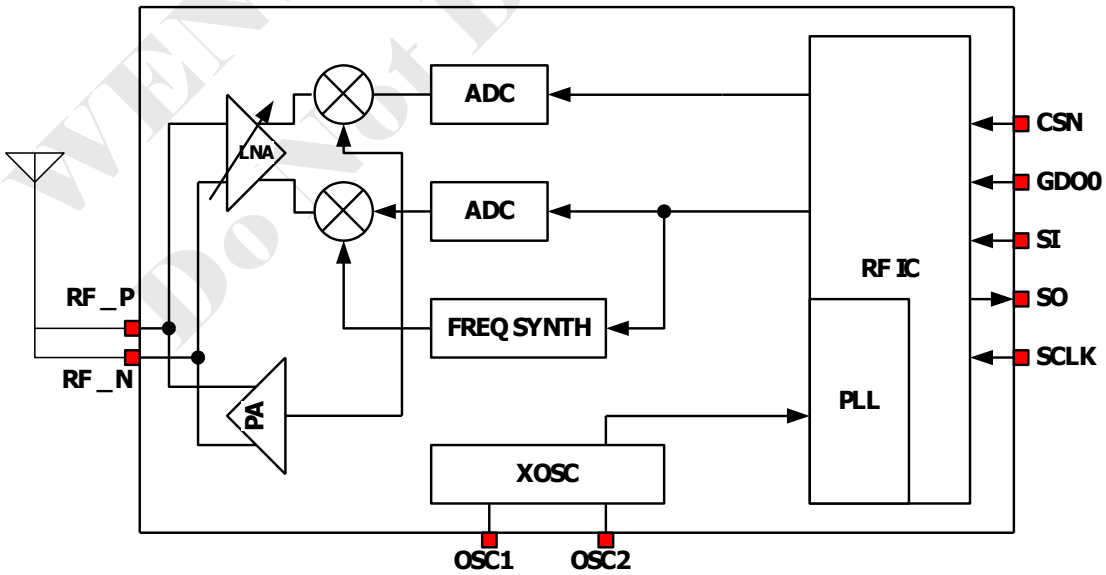
Absolute Maximum Rating

Parameter	Min	Max	Unit	Condition/Note
Supply Voltage	2.7V	3.3V	V	
Voltage on Any Digital Pin	-0.3	VDD+0.3	V	
Input RF Level		+10	dBm	
Storage Temperature Range	-50	150	℃	
ESD		<500	V	According to JEDEC STD 22, method A114, Human Body Model

General Characteristics and Electrical Specification

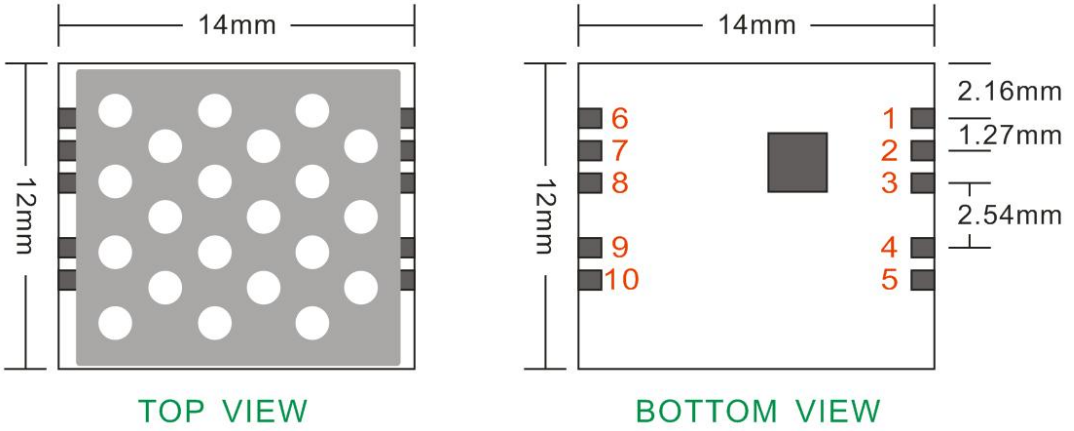
Parameter	Min	Type	Max	Unit	Condition/Note
Frequency Range	300		928	MHz	
Data Rate	1.2		250	Kbps	
Power Down Modes		400		nA	Voltage regulator to digital part off, register values retained (sleep state)
Current Consumption, RX		16		mA	
Current Consumption, TX		30		mA	Transmit mode, +10dBm output power
Receiver Sensitivity	1.2k data rate -111dBm				
Transmit Output Power			+10	dBm	Transmit mode

Block Diagram



View

Pin Assignment



序號	名稱	類型	說明
4	IRQ	輸出	平常狀態爲 0，當爲 1 時，說明 RF 正在傳送或接收資料
5	SET	輸入	低電平有效，爲選擇模組
2	SO	輸出	從模組內讀取的資料，包含資料跟配置資訊
1	SCLK	輸入	給 RF 讀寫提供時鐘
3	SI	輸入	向模組內寫入的資料，包含資料跟配置資訊
10	VDD		電源正極
6,8	GND		電源負極

時序說明

TRW-811S 採用 SPI 的資料傳送方式，每次傳送時都要加上位址，位址的 bit7 爲方向位，當爲 0 時，則是寫入資料。當爲 1 時，則是讀出資料，在配置時讀出的目的爲了驗證寫入的正確性。

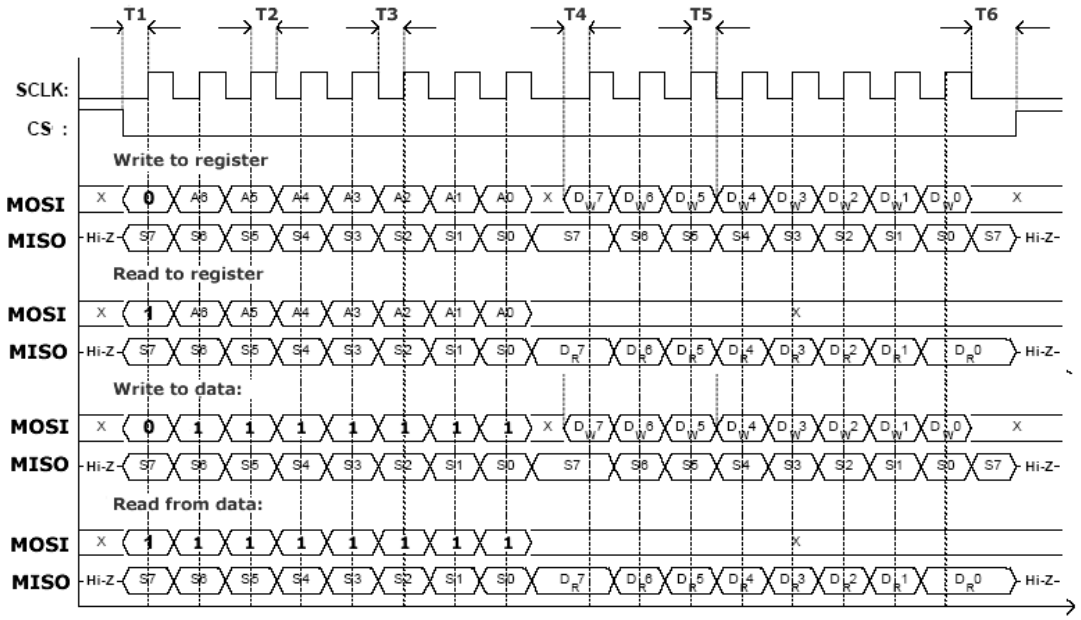
1． 配置時序說明：格式：address+data

2． 資料傳送/接收時序說明：其時序跟配置一樣，只時其位址跟寫讀的個資料不同。

發送格式：0x7F+N 個 BYTE（N<65）。

接收格式：0xFF+N 個 BYTE（N<61）。

3． 寫入要發送的資　料的位址爲 0x7F，一次最多可寫入 64 個 Byte 的資料給 RF。讀出收到的資料位元　址爲 0xFF，一次最多從 RF 讀出 64 個 Byte。



Fsclk<	6.5MHz				
T1	>	200us	T2	>	50ns
T3	>	50ns	T4	>	80ns
T5	>	20ns	T6	>	20ns

命令值說明

命令值	功能說明
0x30	重定 RF 模組
0x32	關閉 RF 的 OSC
0x33	自動鎖頻，鎖頻時間不超過 1ms
0x34	清除接收緩衝區
0x35	清除發送緩衝區
0x36	模組進入靜態模式

0x39	模組進入低功耗模式
0x3A	模組進入接收模式
0x3B	模組進入發送模式
0x7E	發送功率設定
0x7F/0xFF	寫發送/讀接收 緩衝區資料

暫存器說明

地址	1.2K	2.4K	4.8K	10K	38.4	76.8	100	250K	功能說明
	FSK	FSK	FSK	FSK	FSK	FSK	FSK	MSK	調變方式
0x0D	0x1E	0x1E	0x1E	0x1E	0x1E	0x1E	0x1E	0x1E	設置 TRW-811S 的基頻，工作基頻為 868MHz。
0x0E	0xC4	0xC4	0xC4	0xC4	0xC4	0xC4	0xC4	0xC4	
0x0F	0xEC	0xEC	0xEC	0xEC	0xEC	0xEC	0xEC	0xEC	
0x13	0x23	0x23	0x23	0x23	0x23	0x23	0x23	0x23	13H[6:4]為前導碼的個數
0x14	0x3B	0x3B	0x3B	0x3B	0x3B	0x3B	0x3B	0x3B	13H[1:0]14H 為頻道間隔的基頻。
0x0A	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	設置為第幾個頻道
0x0B	0x08	0x08	0x06	0x06	0x06	0x06	0x06	0x0B	0BH[3:0]IF 的設定
0x0C	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	固定
0x10	0xF5	0xF6	0xC7	0xC8	0xCA	0x7B	0x5B	0x2D	接收的帶寬設定
0x11	0x83	0x83	0x83	0x93	0x83	0x83	0xF3	0x3B	設置模組的工作速率
0x12	0x03	0x03	0x03	0x03	0x03	0x03	0x03	0x73	12H[7:4]選擇調製方式
0x15	0x15	0x15	0x15	0x34	0x34	0x42	0x47	0x00	調製度的設定
0x22	0x10	0x10	0x10	0x10	0x10	0x10	0x10	0x10	對應數率固定如下,當頻率<<430.5MHz 時，0x24 的值為 0x0A
0x21	0x56	0x56	0x56	0x56	0x56	0xB6	0xB6	0xB6	
0x18	0x08	0x08	0x08	0x08	0x08	0x08	0x08	0x08	
0x19	0x16	0x16	0x16	0x16	0x16	0x1D	0x1D	0x1D	
0x1A	0x6C	0x6C	0x6C	0x6C	0x6C	0x1C	0x1C	0x1C	
0x1B	0x03	0x03	0x43	0x43	0x43	0xC7	0xC7	0xC7	
0x1C	0x40	0x40	0x40	0x40	0x40	0x00	0x00	0x00	
0x1D	0x91	0x91	0x91	0x91	0x91	0xB2	0xB2	0xB2	
0x23	0xA9	0xA9	0xA9	0xA9	0xA9	0xEA	0xEA	0xEA	

0x24	0x2A	0x2A	0x2A	0x2A	0x2A	0x2A	0x2A	0x2A	
0x25	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	
0x26	0x11	0x11	0x11	0x11	0x11	0x11	0x11	0x11	
0x29	0x59	0x59	0x59	0x59	0x59	0x59	0x59	0x59	
0x2C	0x81	0x81	0x81	0x81	0x81	0x88	0x88	0x88	
0x2D	0x35	0x35	0x35	0x35	0x35	0x31	0x31	0x31	
0x2E	0x0B	0x0B	0x0B	0x0B	0x0B	0x0B	0x0B	0x0B	
0x08	0x05	0x05	0x05	0x05	0x05	0x05	0x05	0x05	封包控制
0x02	0x06	0x06	0x06	0x06	0x06	0x06	0x06	0x06	固定
0x00	0x06	0x06	0x06	0x1B	0x1B	0x1B	0x1B	0x1B	
0x06	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	封包長度的設定，在可變的情況下，其值為 0xFF

此表以基頻爲 868MHz，250K 的頻道間隔，選擇第 0 個頻道，工作在不同的工作速率下之參考。

1. **TRW-811S 的基頻設定：0DH，0EH，0FH**

   FREQ[23:0] = Fcarrier*2^16/26000000

   例 1：工作頻率爲 868MHz，則 FREQ[23:0] = 868000000*2^16/26000000 = 2187894 = 0x216276

   則 0DH = 0x21　0EH = 0x62　0FH = 0x76

   例 2：工作頻率爲 878.5MHz，則 FREQ[23:0] = 878500000*2^16/26000000 = 2214360= 0x21C9D8

   則 0DH = 0x21　0EH = 0xC9　0FH = 0xD8

2. **頻道與頻道的間隔頻率的設定：13H[1:0]14H**

   $f_{CHANNEL}$ = 26000000*(256+11H[7:0])*2^10[1:0]/2^18

   如果要設定頻道的間隔頻率爲 250K，則 14 = 0x3B,13[1:0] = 3;

3. **頻道間隔的選擇：0x0A**

   $f_{carrier} = f_{BASE} + f_{CHANNEL}$*0AH[7:0]

   如果基頻爲 800MHz，f_{CHANNE} = 250K，0x0A=00 的話，則當前的工作頻率爲：800MHz

   如果基頻爲 800，f_{CHANNE} = 250K，0x0A=04 的話，則當前的工作頻率爲：801MHz

4. **前導碼的設定：13H[6:4]**

   13H[6:4] = 0 則爲 2 個 Byte 的前導碼

   13H[6:4] = 1 則爲 3 個 Byte 前導碼

   13H[6:4] = 2 則爲 4 個 Byte 的前導碼

   13H[6:4] = 3 則爲 6 個 Byte 的前導碼

   13H[6:4] = 4 則爲 8 個 Byte 的前導碼

   13H[6:4] = 5 則爲 12 個 Byte 的前導碼

   13H[6:4] = 6 則爲 16 個 Byte 的前導碼

   13H[6:4] = 7 則爲 24 個 Byte 的前導碼

5. 接收帶寬的設定：10H[7:4]，其下對應各值給出的對應頻率：

10H[7:4]	= 00	812K	10H[7:4]	= 01	625K
10H[7:4]	= 02	541K	10H[7:4]	= 03	464K
10H[7:4]	= 04	406K	10H[7:4]	= 05	325K
10H[7:4]	= 06	270K	10H[7:4]	= 07	232K
10H[7:4]	= 08	203K	10H[7:4]	= 09	162K
10H[7:4]	= 10	135K	10H[7:4]	= 11	116K
10H[7:4]	= 12	102K	10H[7:4]	= 13	81K
10H[7:4]	= 14	68K	10H[7:4]	= 15	58K

6. IF 頻率的設定：f_{IF} = 25390.625*0BH[3:0]

例如: 2.4K 的工作速率,0BH[3:0]=8,則 f_{IF} = 203.125KHz

7. 工作速率的設定：10H[3:0]，11H[7:0]，求最接近值:

RATE = (256+11H[7:0])*2^10H[3:0]*26000000/2^28

例：如工作速率爲 2.4K，則 11H = 0x83,10H[3:0] = 0xx6

例：如工作速率爲 1.2K，則 11H = 0x83,10H[3:0] = 0xx5

8. 調製選擇方式：12H[7:4]

12H[7:4] = 0，說明選擇 2-FSK 模式。

12H[7:4] = 1，說明選擇 MSK 模式。

9. 調製度的設定 15H

a. 在 MSK 時，其值永遠都爲 1。

b. 在 2-FSK 時，其調製度的計算公式如下，求最近值

f_{dev} =（8+15H[2:0]*2^15H[6:4]*198

10. 封包控制 08H

08H = 0x04，說明封包的長度是固定的，其長度的值存於 06H 中。

例: 當 06H 中的值爲 0x70 時,說明一個封包在發射和接收的 Byte 的爲 112 個,在發射和接收過程中需要分段發送跟接收才能把資料收完。

發射格式：0x7F+N 個 Byte (N = 06H 中的值)

08H = 0x05，說明封包的長度是可變的，其長度的值爲發射的第一個 Byte 的資料，06H 中的值爲 0xFF。

發射格式：0x7F+封包長度值+N 個 Byte 數據(N=封包的長度)

註：

a. 當在發射狀態時，如果封包的長度≤64，則一次可以把資料寫入 TRW-900C 模組中，如果封包的長度＞64，則需要分段傳送資料給 TRW-900C 模組，因爲 TRW-900C 模組內只有一個 64 個 Byte 的緩衝區。具體分段的方法如下：

10.1.1 第一次先寫入 64 個 Byte 給 TRW-900C 模組。

10.1.2 等待 DR 腳變成高。

10.1.3 讀 0xFA 的值，如果讀到的值爲 0x10，說明緩衝區還有 16Bytes 沒送完，如果讀到的值爲 0x20，說明緩衝區還有 32Bytes 沒送完。

10.1.4 根據讀到 0xFA 的值，用戶可決定什麼時候再次傳送資料給 TRW-900C 模組，但不能讀到其等於 0，否則會有錯。

10.1.5 如果再次傳送資料還沒有把資料送完，重複 10.1.13 與 10.1.14。

b. 當在接收狀態時，等待其 DR 線變成高，如果封包的長度小於 61，則等待 DR 線變低後多讀兩個 BYTE 的資料出來，多讀的兩個資料，前一個爲 RSSI 的值，後一個爲 CRC 的值，當後一個 CRC 的值 bit7 爲 1 時，說明封包正確，否則錯誤，RSSI 值是一個有符號的值，如果讀到的數值越大，說明接收信號越好.請注意：0xFF 爲 -1＜0 的。如果封包的長度大於 60，則具體讀數方法如下：

10.2.1 先等待 DR 線變成 1。

10.2.2 讀 0xFB 中的值如果等於 51，則從 TRW-900C 模組中讀出 50 個 BYTE 資料(注 51/50 可由用戶自己定義，只是要比 0xFB 中的值小 1 就好了。

10.2.3 重複 10.2.2，如果在所在讀 0xFB 時，其 DR 線變成低，說明此次的資料不夠 50 個 Byte，此時須退出讀 0xFB 的狀態，讀出剩餘的資料（注：讀出的資料比實際要多兩個）。

10.2.4 當 DR 線變成低後，怎樣知道還要讀多少的資料：

在固定封包的狀態下，由 0x06 中的值-N 次 50+2。

在可變封包的狀態下，由讀到的第一個 BYTE 的值-N 次 50+2。

11. 功率設定：

頻率範圍 Frequency	**功率 Data Rate（dBm）**													
	+10	+8	+6	+4	+2	+0	-2	-4	-6	-8	-10	-12	-15	-52
300~400MHz	C2	C8	CE	89	8D	3F	54	57	6B	35	34	33	1D	00
400~750MHz	C0	C5	81	88	8D	3F	53	57	2A	35	34	33	1C	00
750~999MHz	C3	C9	CE	87	8C	3F	54	57	2B	28	34	33	23	00

設定功率的位址爲 0x7E+8 個 BYTE 的資料，8 個 Byte 設成一樣，選擇以上的功率值。

讀設定功率的值爲 0xFE+8 個 BYTE 的資料，8 個 Byte 是否跟設成的一樣。

應用程式範例

1．重定 TRW-900C　　副程式：　Reset-TRW-900C。

2．配置 TRW-900C　　副程式：　Config-TRW-900C。

3．選擇工作模式：

發射模式：　　副程式：　TRW-900C-TxMode。

接收模式：　　副程式：　TRW-900C-RxMode。

空閒模式：　　副程式：　TRW-900C-TdleMode。

4．資料傳遞：

發送資料　　副程式：　TRW-900C-Send-Data。

接收資料　　副程式：　TRW-900C-Receive-Data。

5．其他模式：

設置功率　　副程式：　TRW-900C-Set-Power。

低功耗模式　　副程式：　TRW-900C-LowPowerMode。

```
/* ********************************************************************
/* 公司名稱 ：文星電子股份有限公司
/* 模組名稱：TRW-900C
/* 創建人 ：LHX 日期 ：2006-11-03
/* 修改人 ：LHX 日期 ：2006-11-03
/* 功能描述 ：如何使用TRW-900C
/* 版本 ：Ver1.00
/********************************************************************
#include <C8051F320.H>
/********************************************************************
/********************************************************************
/* ********************************************************************
const unsigned char code TRW-900C_Table[68];
/* ********************************************************************
sbit CLK = P1^1;
sbit MISO = P1^2;
sbit MOSI = P1^3;
sbit CS = P1^6;
sbit DR = P1^4;
sbit TX_LED = P2^1;
/********************************************************************
/* 函數名 ：W-TRW-900C- Byte */
/* 功能描述 ：向TRW-900C寫入一個Byte的資料 */
/* 輸入 ：x */
/* 返回 ：無 */
/********************************************************************
void W-TRW-900C_Byte(char x)
{
unsigned char i;
for(i=0;i<8;i++)
{
CLK = 0;
MOSI = 0;
if(x&0x80)
MOSI = 1;
x<<=1;
CLK = 1;
}
CLK = 0;
```

```
}
/*********************************************************************
/* 函數名 ：R-TRW-900C- Byte  */
/* 功能描述 ：從TRW-900C讀出一個Byte的資料 */
/* 輸入 ：無 */
/* 返回 ：x */
/*********************************************************************
char R-TRW-900C-Byte(void)
{
unsigned char i,x;
for(i=0;i<8;i++)
{
CLK = 0;
x <<= 1;
x &= 0xFE;
if(MISO)
x|= 0x01;
CLK = 1;
}
CLK = 0;
return(x);
}
/*********************************************************************
/* 函數名 ：Config-TRW-900C_ Byte */
/* 功能描述 ：配置TRW-900C模組的一個Byte */
/* 調用函數 ：W-TRW-900C- Byte */
/* R-TRW-900C- Byte */
/* 函數說明 ：當address的第七位爲1時，讀取其 */
/* 其內的值來判斷寫入的資料是否正確
/* 輸入 ：address */
/* 返回 ：c_data */
/* 功能描述 ：配置TRW-900C模組的一個Byte */
*********************************************************************************************************
char Config-TRW-900C-Byte(char address,char c_data)
{
CS = 0;
while(MISO); // 判斷MISO是否爲低電平，否則等待變成低電平後退出
W-TRW-900C-Byte(address);
if(address&0x80)
c_data = R-TRW-900C-Byte();
else
W-TRW-900C-Byte(c_data);
CS = 1;
```

```
return(c_data);
}
/*********************************************************************
/* 函數名 ：Reset-TRW-900C*/
/* 功能描述 ：復位TRW-900C*/
/* 調用函數 ：W-TRW-900C- Byte */
/* 輸入 ：無*/
/* 返回 ：無*/
/*********************************************************************
void Reset-TRW-900C(void)
{
unsigned char i;
CS = 1;
for(i=0;i<100;i++);
CS = 0;
for(i=0;i<200;i++);
CS = 1;
for(i=0;i<200;i++);
CS = 0;
while(MISO); // 判斷MISO是否爲低電平，否則等待變成低電平後退出
W_TRW-900C_Byte(0x30);
CS = 1;
}
/*********************************************************************
/* 函數名 ：TRW-900C-Set-Power*/
/* 功能描述 ：設置TRW-900C功率*/
/* 調用函數 ：W-TRW-900C- Byte */
/* 函數說明 ：一次要寫8個Byte的數據*/
/* 輸入 ：x */
/* 返回 ：無*/
/*********************************************************************
void TRW-900C_Set_Power(char x)
{
unsigned char i;
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x7E);
for(i=0;i<8;i++)
W_TRW-900C_Byte(x);
CS = 1;
}
/*********************************************************************
/* 函數名 ：Config_TRW-900C*/
```

```
/* 功能描述 ：配置TRW-900C*/
/* 調用函數 ：Config_TRW-900C_ Byte */
/* : W_TRW-900C_ Byte */
/* 輸入 ：無*/
/* 返回 ：無*/
/*********************************************************************
void Config_TRW-900C(void)
{
unsigned char i,x;
do
{
for(i=0;i<68;)
Config_TRW-900C_Byte(TRW-900C_Table[i++],TRW-900C_Table[i++]);
TRW-900C_Set_Power(0xC3); //  設置最高功率發射
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x33);
CS = 1;
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x37);
CS = 1;
for(i=0;i<200;i++);
i = Config_TRW-900C_Byte(0x80,x);
}while(i!=0x1B);
}
/*************************************************************************************************
/* 函數名 ：TRW-900C_Tx Mode*/
/* 功能描述 ：配置TRW-900C工作在發射模式*/
/* 函數說明 ：此函數可使TRW-900C直接從接收模式或空閒模式切回發射模式*/
/* 調用函數 ：W_TRW-900C_ Byte */
/* 輸入 ：無*/
/* 返回 ：無*/
/*************************************************************************************************
void TRW-900C_TxMode(void)
{
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x36);
CS = 1;
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x3B);
```

```
CS = 1;
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x35);
CS = 1;
}
/*********************************************************************
/* 函數名 ：TRW-900C_Rx Mode*/
/* 功能描述 ：配置TRW-900C工作在接收模式*/
/* 函數說明 ：此函數可使TRW-900C直接從發射模式或發射模式切回發射模式*/
/* 調用函數 ：W_TRW-900C_Byte*/
/* 輸入 ：無*/
/* 返回 ：無*/
*********************************************************************/
void TRW-900C_Rx Mode(void)
{
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x36);
CS = 1;
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x3A);
CS = 1;
CS = 0;
while(MISO);

W_TRW-900C_Byte(0x34);
CS = 1;
}
/*********************************************************************
/* 函數名 ：TRW-900C_Idle Mode*/
/* 功能描述 ：配置TRW-900C工作在空閒模式*/
/* 函數說明 ：此函數可使TRW-900C直接從發射*/
); 模式或接收模式切回空閒模式,應用場合屬於不想接收，但要求快速返回發射或接收狀態*/
/* 調用函數 ：W_TRW-900C_Byte*/
/* 輸入 ：無*/
/* 返回 ：無*/
*********************************************************************/
void TRW-900C_Idle Mode(void)
{
CS = 0;
while(MISO);
```

```
W_TRW-900C_Byte(0x36);
CS = 1;
}
/*********************************************************************
void Send_FIFO_Pointer(void)
{
unsigned char i;
while(Config_TRW-900C_Byte(0xFA,0x00)>0x10)
for(i=0;i<200;i++);
}
/*****************************************************************************************************
/*  函數名：TRW-900C_Send_Data */
/*  功能描述：讓TRW-900C發送資料  */
/*  函數說明：固定發送資料88H，在發送過程中，盡可能的避免一次性多個Byte的資料爲
0x00,0xFF  否則接收有錯或收不到。*/
/*  調用函數：W_TRW-900C_Byte */
/*  輸入：x指一個封包需要發送的Byte數，以可變封包爲例,但一次封包小於110個Byte*/
/*  返回：無*/
/*****************************************************************************************************
void TRW-900C_Send_Data(unsigned char x)
{
unsigned char i;
TRW-900C_TxMode();
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x7F);
W_TRW-900C_Byte(x);
if(x<65)
{
for(i=0;i<x;i++)
W_TRW-900C_Byte(0x88); //  發送88H的資料
CS = 1;
}
else
{
for(i=0;i<60;i++)
W_TRW-900C_Byte(0x88); //  發送88H的資料
CS = 1;
while(!DR);
Send_FIFO_Pointer();
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x7F);
```

```
for(i=0;i<(x-60);i++)
W_TRW-900C_Byte(0x88); //  發送88H的資料
CS = 1;
}
while(!DR); //  等待發射完成，發射完成後，自動進入IDLE模式，如果發送完後想進入接收模式的話，就寫TRW-900C_Rx Mode()函數
}
/*********************************************************************
/*  函數名：Receive_FIFO_Pointer */
/*  功能描述：讀TRW-900C接收資料指標，當收到51個資料時，退出  */
/*  調用函數：W_TRW-900C_Byte */
/*  輸入：無  */
/*  返回：0 ,說明不到50個Byte的資料就接收結束  */
/*  返回：1 ,說明有收到50個Byte的資料  */
/*********************************************************************
unsigned char Receive_FIFO_Pointer(void)
{
unsigned char i = 0;
while(Config_TRW-900C_Byte(0xFB,0x00)<51)
{
TX_LED = ~TX_LED;
for(i=0;i<200;i++)
if(!DR)
i = 210;
if(i==211)
break; //退出while語句
}
if(i==211)
return(0);
else
return(1);
}
/*********************************************************************
/*  函數名：TRW-900C_Receive_Data */
/*  功能描述：讓TRW-900C接收資料  */
/*  函數說明：以可變封包爲例  */
/*  調用函數：W_TRW-900C_Byte */
/*  輸入：無  */
/*  返回：0 ,說明沒有收到資料  */
/*  返回：1 ,說明有收到錯誤的資料  */
/*  返回：2 ,說明有收到正確的資料  */
/*********************************************************************
char TRW-900C_Receive_Data(void)
```

```
{
unsigned char i = 0, x = 0,y = 0,z,CRC_Value;
unsigned char xdata RF_Buffer[100],RF_Pointer = 0;
while(DR)
{
if(Receive_FIFO_Pointer())
{
x++;
CS = 0;
while(MISO);
W_TRW-900C_Byte(0xFF);
i = 0;
if(x==0)
{
RF_Pointer = 0;
y = R_TRW-900C_Byte();
y ++ ; // 在接收資料時，要比實際發射的資料多兩個，前一個爲RSSI，後一個BIT7爲CRC值，
如CRC正確，則BIT7 = 1，否則爲0
RF_Buffer[RF_Pointer++] = y-1;
}
for(;i<50;i++)
RF_Buffer[RF_Pointer++] = R_TRW-900C_Byte();
CS = 1;
z = y-50;
}
else
{
CS = 0;
while(MISO);
W_TRW-900C_Byte(0xFF);
if(x==0)
{
RF_Pointer = 0;
y = R_TRW-900C_Byte();
y ++;
RF_Buffer[RF_Pointer++] = y-1;
for(i=0;i<y;i++)
RF_Buffer[RF_Pointer++] = R_TRW-900C_Byte);I
}
else
{
for(i=0;i<z;i++)
RF_Buffer[RF_Pointer++] = R_TRW-900C_Byte();
```

```
}
CRC_Value = R_TRW-900C_Byte();
RF_Buffer[RF_Pointer++] = CRC_Value;
CS = 1;
z = 0x7F;
}
}
if(z == 0x7F)
{
if(CRC_Value&0x80)
return(2);
else
return(1);
}
else
return(0);
}
/*********************************************************************
/* 函數名：TRW-900C_Low Power Mode */
/* 功能描述：讓TRW-900C工作在低功耗狀態 */
/* 調用函數：W_TRW-900C_Byte */
/* 輸入：無 */
/* 返回：無 */
/*********************************************************************
void TRW-900C_Low Power Mode(void)
{
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x36);
CS = 1;
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x39);
CS = 1;
CS = 0;
while(MISO);
W_TRW-900C_Byte(0x32);
CS = 1;
}
/*********************************************************************
void Init_MCU_Status (void)
{
P1MDOUT = 0x4B;
```

```
P2MDOUT = 0x01;
P1 = 0xF4;
XBR1 = 0x40;
OSCICN |= 0x03; //  工作頻率爲24M
}

/*************************************************************************
/*  函數名：Main函數  */
/* MCU Body：C8051F32x */
/*  輸入：無  */
/*  返回：無  */
/*************************************************************************
void main(void)
{
unsigned int x;
unsigned char i;
PCA0MD = 0x00;
for(x=0;x<30000;x++);
Init_MCU_Status();
Reset_TRW-900C();
Config_TRW-900C();
for(x=0;x<1000;x++);
while(1)
{
TRW-900C_Send_Data(90);
TRW-900C_RxMode();
for(x=0;x<60000;x++)
TRW-900C_Receive_Data();
}
}
const unsigned char code TRW-900C_Table[68] =
{
0x0D,0x21, // 1.2K 2-FSK 868M
0x0E,0x62,
0x0F,0x76,
0x0B,0x08,
0x0C,0x00,
0x10,0xF5,
0x11,0x83,
0x12,0x03,
0x13,0x43,
0x14,0x3B,
0x0A,0x00,
```

```
0x15,0x15,
0x22,0x10,
0x21,0x56,
0x18,0x08,
0x19,0x16,
0x1A,0x6C,
0x1B,0x03,
0x1C,0x40,
0x1D,0x91,
0x23,0xA9,
0x24,0x2A,
0x25,0x00,
0x26,0x11,
0x29,0x59,
0x2C,0x81,
0x2D,0x35,
0x2E,0x0B,
0x08,0x05,
0x07,0x04,
0x02,0x06,
0x00,0x1B,
0x09,0x00,
0x06,0xFF
/* 0x0D,0x21, ; 250K MSK 868
0x0E,0x62,
0x0F,0x76,
0x0B,0x0B,
0x0C,0x00,
0x10,0x2D,
0x11,0x3B,
0x12,0x73,
0x13,0x43,
0x14,0x3B,
0x0A,0x00,
0x15,0x00,
0x22,0x10,
0x21,0xB6,
0x18,0x08,
0x19,0x1D,
0x1A,0x1C,
0x1B,0xC7,
0x1C,0x00,
0x1D,0xB2,
```

```
0x23,0xEA,
0x24,0x2A,
0x25,0x00,
0x26,0x11,
0x29,0x59,
0x2C,0x88,
0x2D,0x31,
0x2E,0x0B,
0x08,0x05,
0x07,0x04,
0x02,0x06,
0x00,0x06,
0x09,0x00,
0x06,0xFF
*/};


/* ********************************************************************************** */
// 此程式可用於TWS-900C與TRW-900C與TRW-400配合使用的參考程式
/* ********************************************************************************** */
void main (void)
{
Delayms(200)
Init_MCU ();
Config_TWS-900C ();
while(1)
{
TRW-400_Send_Data ();
}
}
/* ********************************************************** */
void Write_Word_TRW-400(char x,char y,char z)
{
unsigned char i;
do
{
for(i=0;i<8;i++)
{
CLK = 0;
WR = 0;
if(y&0x80)
WR = 1;
CLK = 1;
```

```
y<<=1;
}
y=z;
}while(x-->0)
}
/* ************************************************************ */
void RESET_TRW-400 (void)
{
unsigned char i;
CE = 1;
for(i=0;i<10;i++);
CE = 0;
for(i=0;i<10;i++);
CE = 1;
for(i=0;i<100;i++);
CE = 0;
while(RD);
Write_Word_TRW-400(1,0x30,0x00);
while(!RD);
CE = 1;
}
/* ************************************************************ */
void Config_TWS-900C(void)
{
unsigned char i;
RESET_TRW-400 ();
for(i=0;i<50;)
{
CE = 0;
while(RD);
Write_Word_TRW-400(2,Config_Table[i++],Config_Table[i++]);
CE = 1;
}
Write_Word_TRW-400 (9,0x7E,0xC3);
// 0xC3 +10dBm輸出
// 0xC6 + 9dBm輸出
// 0xC9 + 8dBm輸出
// 0xCC + 7dBm輸出
// 0xCE + 6dBm輸出
// 0x86 + 5dBm輸出
// 0x89 + 4dBm輸出
// 0x8C + 3dBm輸出
// 0x8D + 2dBm輸出
```

```
// 0x3F + 0dBm輸出
CE = 0;
while(RD);
Write_Word_TRW-400(1,0x37);
CE = 1;
Delayms(1);
CE = 0;
while(RD);
Write_Word_TRW-400(1,0x33);
CE = 1;
Delayms(5);
}
/* ************************************************************ */
void TRW-400_Send_Data(void)
{
CE = 0;
while(RD);
Write_Word_TRW-400(1,0x3B,0x00);
CE = 1;
Delayms(1);
CE = 0;
while(RD);
Write_Word_TRW-400(1,0x35,0x00);
CE = 1;
CE = 0;
while(RD);
Write_Word_TRW-400(14,0x7F,Send_Table[i]);
CE = 1;
while(!FLAG);
while(FLAG);
Delayms(1);
}
/* ************************************************************ */
unsigned char code Send_Table[] =
{
0x12,0x34,0x56,0x78,
0x01,0x02,0x03,0x04,
0x05,0x06,0x07,0x5D,0x5D
}
/* ************************************************************ */
// 配置說明：
// 工作頻率：434MHz
// 傳送速率：4.8K
```

```
// 封包BYTE：位址的個數+資料的個數+CRC的個數
//  ：4+7+2 = 13
/* ************************************************************** */
unsigned char code Config_Table[] =
{
0x02,0x06, //
0x04,0x55, //
0x05,0x55, //
0x06,0x0E, // 14
0x08,0x00, //
0x0A,0x00, //
0x0D,0x10, //
0x0E,0xB1, //
0x0F,0x3B, //
0x10,0x87, //
0x11,0x83, //
0x12,0x03, //
0x13,0x02, //
0x14,0xF8, //
0x15,0x04, //
0x18,0x08, //
0x22,0x10, //
0x23,0xA9, //
0x24,0x2A, //
0x25,0x00, //
0x26,0x11, //
0x29,0x59, //
0x2C,0x81, //
0x2D,0x35, //
0x2E,0x0B //
}
```


問與答

1、問：TRW-811S 之功率地址是否爲 0x7E/7F/80/81/82/83/84/85 中存入數據?

答：不是，0x7E 位址裏本身就有 8 個 BYTE 的資料緩衝（這樣理解就好了），不是 0x7E/7F/80/81/82/83/84/85 分別一個。

2、問：miso 是否再 cs = 0 時會 = 0

答：是，當 CS = 0 時，MISO 會變成低（但前提是 MCU 接這個腳必須爲輸入，否則會有影響）後，再往模組內寫寫入資料。