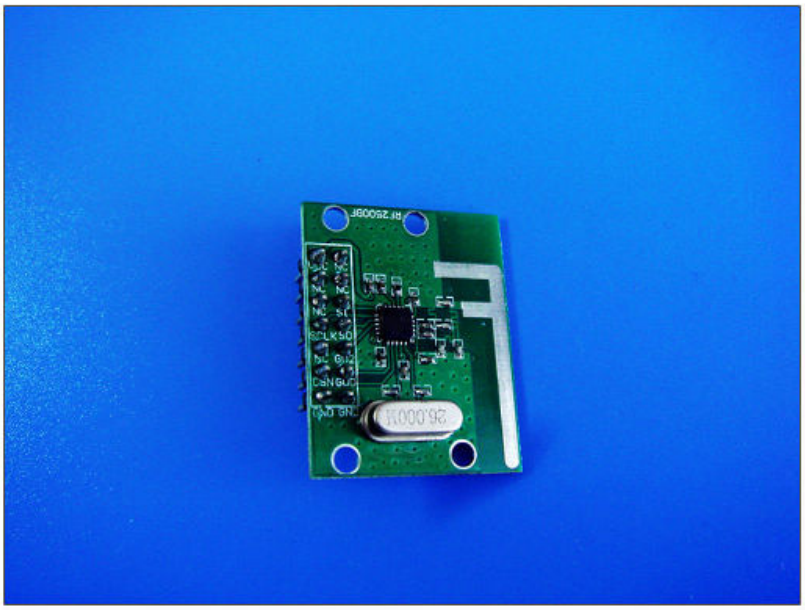
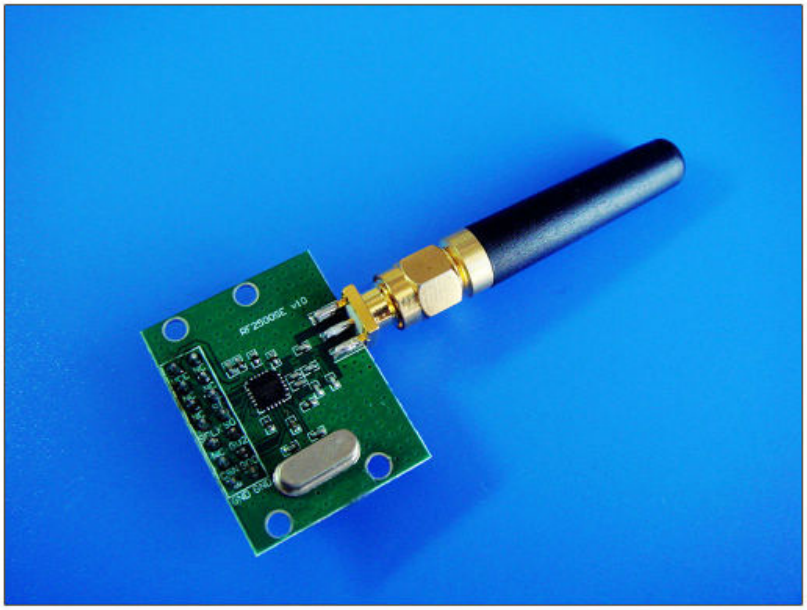
CC2500 无线模块

用户手册





联系电话：13704018223 陈 工 E-mail: chj_006@sina.com
在线咨询：QQ:35625400 474882985 MSN:1188mm88@hotmail.com

目录

联系电话：13704018223 陈 工 E-mail: chj_006@sina.com
在线咨询：QQ:35625400 474882985 MSN:1188mm88@hotmail.com

产品介绍

CC2500 芯片，是 TI（原 Chipcon 被 TI 收购）推出的一款超低功耗、低成本的无线收发模块，其载频范围在 2.400GHz～2.483GHz 内可调，可用来实现多信道通信。它支持多种调制方式，包括 FSK、GFSK、OOK 和 MSK，数据传输速率最高可达 500kb/s。CC2500 还为信息包处理、数据缓冲、脉冲传送、空闲信道评估、连接品质指示和电磁唤醒等功能提供了额外的硬件支持。它有四种主要的状态：接收（RX）、发送（TX）、空闲（IDLE）和休眠（SLEEP）

基本特点

(1) 2400-2483.5 MHz 的 ISM 和 SRD 频段

(2) 最高工作速率 500kbps，支持 2-FSK、GFSK 和 MSK 调制方式

(3) 高灵敏度（-101dBm 在 10Kbps 1%）

(4) 内置硬件 CRC 检错和点对多点通信地址控制

(5) 较低的电流消耗（RX 中，13.3mA）

(6) 可编程控制的输出功率，对所有的支持频率可达 1dBm

(7) 支持低功率电磁波激活功能

(8) 支持传输前自动清理信道访问（CCA），即载波侦听系统

(9) 快速频率变动合成器带来的合适的频率跳跃系统

(10) 模块可软件设地址，软件编程非常方便

(11) 标准 DIP 间距接口，便于嵌入式应用

联系电话：13704018223 陈 工 E-mail: chj_006@sina.com
在线咨询：QQ:35625400 474882985 MSN:1188mm88@hotmail.com

(12) 单独的64字节RX和TX数据FIFO

WOR功能：为了节约电能，射频芯片通常采用休眠模式。芯片在休眠时势必会丢失信息，CC22500的WOR（Wakeup-on-Radio）功能能很好地避免这点。WOR功能保证芯片在深度睡眠时周期性地苏醒，探听周围是否有信号，这个过程不需要CPU的中断，如果有数据包成功接收，芯片可通过引脚输出中断通知MCU读取。

RSSI和LQI功能：RSSI反映接收信号强度，LQI反映信号的连接质量，两者都可以通过读取芯片的寄存器得到。LQI虽然能够判断连接质量，但会因调制方式的不同而不同。RSSI是判断两个节点距离的很好的参数。在从RSSI寄存器中读到数值后我们需要进行一系列转化，才能得到接收强度值。

CCA功能：CCA（Clear Channel Assessment）能够指示当前信道是否处于空闲状态。其作用与CSMA相似。当芯片要转入发送模式时，会首先检查信道，只有当信道为空闲时，才进入发送模式，否则停留在原模式或由编程设定进入其他模式。

典型应用场合

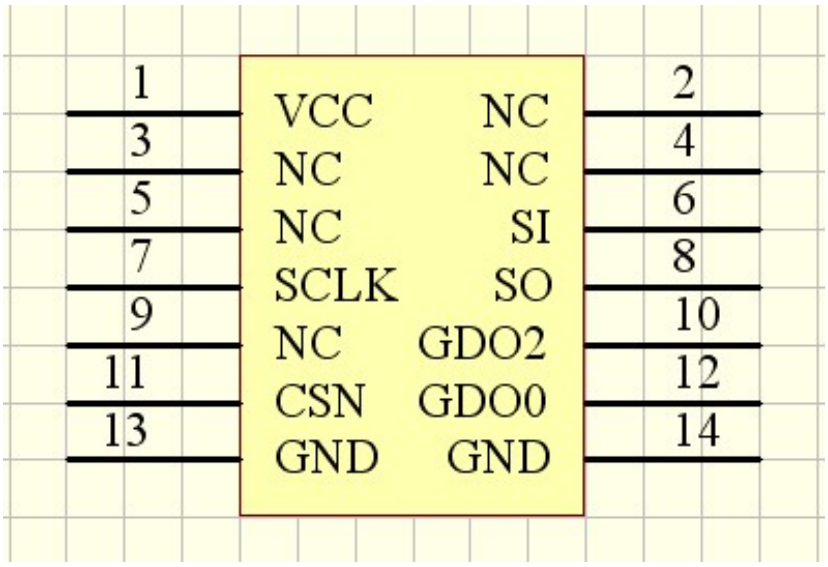
无线遥控，无线鼠标，无线键盘；

工业无线控制，自动化数据采集系统；

无线传感器，无线电子标签，遥控玩具；

水、气、热、电等居民计量表具无线自动抄表；

模块接口说明



引脚功能

引脚	引脚名	引脚类型	描述
1	VCC	电源输入	1.8V-3.6V之间
2-5	NC	无	悬空
6	SI	数字输入	SPI从设备数据输入
7	SCLK	数字输入	SPI从设备时钟输入
8	SO(GDO1)	数字输出	SPI从设备数据输出
9	NC	无	悬空
10	GDO2	数字输出	工作状态引脚
11	CSN	数字输入	连续配置接口，芯片选择
12	GDO0	数字输出	工作状态引脚
13-14	GND	电源地	和系统共地

联系电话：13704018223 陈 工 E-mail: chj_006@sina.com
在线咨询：QQ:35625400 474882985 MSN:1188mm88@hotmail.com

备注

1.VCC 引脚的电压范围为1.9-3.6V 之间，不能在这个区间之外，如超过 3.6V 将会烧毁模块。推荐电压 3.3V 左右；

2.硬件没有集成SPI功能的单片机也可以控制本模块，用普通单片IO口模拟 SPI 时序进行读写操作即可；

3. 模块接口采用标准2.54mmDIP插针，13 脚、14 脚为接地脚,需要和系统电路的逻辑地连接起来；

4.与 51 系列单片机 P0 口连接时候，需要加 10K 的上拉电阻,与其余口连接不需要。其他系列的5V单片机，如AVR、PIC，请参考该系列单片机 IO 口输出电流大小，如果超过 10mA，需要串联2-5K电阻分压，否则容易烧毁模块！如果是 3.3V 的MCU，可以直接和IO口连接。

联系电话：13704018223 陈 工 E-mail: chj_006@sina.com
在线咨询：QQ:35625400 474882985 MSN:1188mm88@hotmail.com

CC2500 模块工作方式

所有配置参数和收发数据都是单片机通过 SPI 对 CC2500 进行读写操作来完成的。SIP 接口的待机模式、发送模式以及接收等工作模式都通过 SPI 指令进行设置。并可以通过 GDO0 或 GDO2 引脚高低电平状态来判断数据的发送或接收是否完成。

工作模式寄存器介绍

<table>
<tr><th>比特</th><th>名称</th><th>描述</th></tr>
<tr><td>7</td><td>CHIP_RDYn</td><td>保持高，直到功率和晶体已稳定。当SPI操作时为低</td></tr>
<tr><td>6:4</td><td>STATE[2:0]</td><td>表明当前主状态机模式
<table>
<tr><th>值</th><th>状态</th><th>描述</th></tr>
<tr><td>000</td><td>空闲</td><td>空闲状态</td></tr>
<tr><td>001</td><td>RX</td><td>接收模式</td></tr>
<tr><td>010</td><td>TX</td><td>发送模式</td></tr>
<tr><td>011</td><td>FSTXON</td><td>快速TX准备</td></tr>
<tr><td>100</td><td>校准</td><td>频率合成器校准正运行</td></tr>
<tr><td>101</td><td>迁移</td><td>PLL正迁移</td></tr>
<tr><td>110</td><td>RXFIFO_OVERFLOW</td><td>RX FIFO已经溢出。读出任何有用数据，然后用SFRX冲洗FIFO。</td></tr>
<tr><td>111</td><td>TXFIFO_OVERFLOW</td><td>TX FIFO已经下溢。同SFTX应答</td></tr>
</table></td></tr>
<tr><td>3:0</td><td>FIFO_BYTES_AVAILABLE[3:0]</td><td>TX FIFO 中的自由比特数。若FIFO_BYTES_AVAILABLE=15，它表明有15或更多个比特是可用/自由的。</td></tr>
</table>

命令寄存器介绍

地址	滤波名	描述
0x30	SRES	重启芯片
0x31	SFSTXON	开启和校准频率合成器（若MCSM0.FSAUTOCAL=1）
0x32	SXOFF	关闭晶体振荡器

联系电话：13704018223 陈 工 E-mail: chj_006@sina.com
在线咨询：QQ:35625400 474882985 MSN:1188mm88@hotmail.com

0x33	SCAL	校准频率合成器并关断（开启快速启动）。在不设置手动校准模式（MCSM0.FS_AUTOCAL=0）的情况下，SCAL从空闲模式滤波。
0x34	SRX	启用RX。若上一状态为空闲且MCSM0.FS_AUTOCAL=1则首先运行校准。
0x35	STX	空闲状态：启用TX。若MCSM0.FS_AUTOCAL=1首先运行校准。若在RX状态且CCA启用：若信道为空则进入TX
0x36	SIDLE	离开RX/TX,关断频率合成器并离开电磁波激活模式若可用
0x37	SAFC	运行22.1节列出的频率合成器的AFC调节
0x38	SWOR	运行27.5节描述的自动RX选举序列（电磁波激活）
0x39	SPWD	当CSn为高时进入功率降低模式。
0x3A	SFRX	冲洗RX FIFO缓冲
0x3B	SFTX	冲洗TX FIFO缓冲
0x3C	SWORRST	重新设置真实时间时钟
0x3D	SNOP	无操作。可能用来为更简单的软件将滤波命令变为2字节。

功能配置寄存器介绍

地址	寄存器	描述	保存在休眠状态中
0x00	IOCFG2	GDO2输出脚配置	Yes
0x01	IOCFG1	GDO1输出脚配置	Yes
0x02	IOCFG0	GDO0输出脚配置	Yes
0x03	FIFOTHR	RX FIFO和TX FIFO门限	Yes
0x04	SYNC1	同步词汇，高字节	Yes
0x05	SYNC0	同步词汇，低字节	Yes
0x06	PKTLEN	数据包长度	Yes
0x07	PKTCTRL1	数据包自动控制	Yes
0x08	PKTCTRL0	数据包自动控制	Yes
0x09	ADDR	设备地址	Yes
0x0A	CHANNR	信道数	Yes
0x0B	FSCTRL1	频率合成器控制	Yes
0x0C	FSCTRL0	频率控制词汇，高字节	Yes
0x0D	FREQ2	频率控制词汇，中间字节	Yes
0x0E	FREQ1	频率控制词汇，低字节	Yes
0x0F	FREQ0	调制器配置	Yes
0x10	MDMCFG4	调制器配置	Yes
0x11	MDMCFG3	调制器配置	Yes
0x12	MDMCFG2	调制器配置	Yes
0x13	MDMCFG1	调制器配置	Yes
0x14	MDMCFG0	调制器背离设置	Yes

0x15	DEVIATN	主通信控制状态机配置	Yes
0x16	MCSM2	主通信控制状态机配置	Yes
0x17	MCSM1	主通信控制状态机配置	Yes
0x18	MCSM0	频率偏移补偿配置	Yes
0x19	FOCCFG	位同步配置	Yes
0x1A	BSCFG	AGC控制	Yes
0x1B	AGCTRL2	AGC控制	Yes
0x1C	AGCTRL1	AGC控制	Yes
0x1D	AGCTRL0	高字节时间0暂停	Yes
0x1E	WOREVT1	低字节时间0暂停	Yes
0x1F	WOREVT0	电磁波激活控制	Yes
0x20	WORCTRL	前末端RX配置	Yes
0x21	FREND1	前末端TX配置	Yes
0x22	FREND0	频率合成器校准	Yes
0x23	FSCAL3	频率合成器校准	Yes
0x24	FSCAL2	频率合成器校准	Yes
0x25	FSCAL1	频率合成器校准	Yes
0x26	FSCAL0	RC振荡器配置	Yes
0x27	RCCTRL1	RC振荡器配置	Yes
0x28	RCCTRL0	频率合成器校准控制	Yes
0x29	FSTEST	产品测试	No
0x2A	PTEST	AGC测试	No
0x2B	AGCTEST	不同的测试设置	No
0x2C	TEST2	不同的测试设置	No
0x2D	TEST1	不同的测试设置	No
0x2E	TEST0		No

状态寄存器介绍

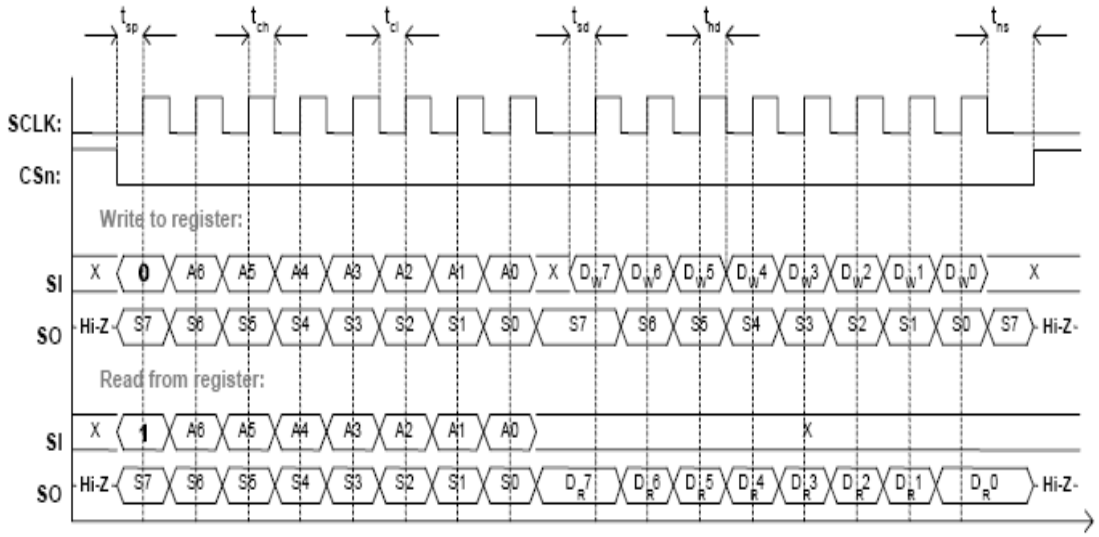
地址	寄存器	描述
0x30 (0xF0)	PARTNUM	
0x31 (0xF1)	VERSION	当前版本数
0x32 (0xF2)	FREQEST	频率偏移估计
0x33 (0xF3)	LQI	连接质量的解调器估计
0x34 (0xF4)	RSSI	接收信号强度指示
0x35 (0xF5)	MARCSTATE	控制状态机状态
0x36 (0xF6)	WORTIME1	WOR计时器高字节
0x37 (0xF7)	WORTIME0	WOR计时器低字节
0x38 (0xF8)	PKTSTATUS	当前GDOx状态和数据包状态
0x39 (0xF9)	VCOVCDAC	PLL校准模块的当前设定
0x3A (0xFA)	TXBYTES	TX FIFO中的下溢和比特数
0x3B (0xFB)	RXBYTES	RX FIFO中的下溢和比特数

程序参考设计

用CC2500模块无需掌握任何专业无线或高频方面的理论，读者只需要具备一定的C语言程序基础即可。本文档没有涉及到的问题，读者可以参考CC2500官方手册或向我们寻求技术支持。

同时，为便于用户开发，我们提供系列配套评估套件，为产品开发保驾护航，使无线应用开发大大加速，并避免不必要的误区。

SPI 时序示意图



SPI 接口时序规范

参数	描述	最小值	最大值
FSCLK	SCLK频率	0	10MHz
tsp,pd	CSn低到SCLK的正边缘，功率降低模式下	TBDus	-

联系电话：13704018223 陈 工 E-mail: chj_006@sina.com
在线咨询：QQ:35625400 474882985 MSN:1188mm88@hotmail.com

tsp	CSn低到SCLK的正边缘，活动模式下	TBDns	-
tch	时钟高	50ns	-
tcl	时钟低	50ns	-
trise	时钟上升时间	-	TBDns
tfall	时钟上升时间	-	TBDns
tsd	向SCLK的正边缘建立数据	TBDns	-
thd	在SCLK的正边缘之后保持数据	TBDns	-
tns	SCLK到CSn高时的负边缘	TBDns	-

参考例程

更多功率参数设置可详细参考 DATACC2500 英文文档中参数表
INT8U PaTabel[8] = {0xC0 ,0xC0 ,0xC0 ,0xC0 ,0xC0 ,0xC0 ,0xC0 ,0xC0};

SPI 读写操作

```
INT8U SpiTxRxByte(INT8U dat)
{
    INT8U i,temp;
    temp = 0;
    SCK = 0;
    for(i=0; i<8; i++)
    {
        if(dat & 0x80)
        {
            MOSI = 1;
        }
        else MOSI = 0;
        dat <<= 1;
        SCK = 1;
        _nop_();
        _nop_();
        temp <<= 1;
        if(MISO)temp++;
```

```
        SCK = 0;
        _nop_();
        _nop_();
    }
    return temp;
}
```

SPI 写寄存器操作

```
void halSpiWriteReg(INT8U addr, INT8U value)
{
    CSN = 0;
    while (MISO);
    SpiTxRxByte(addr);      //写地址
    SpiTxRxByte(value);     //写入配置
    CSN = 1;
}
```

SPI 读寄存器操作

```
INT8U halSpiReadReg(INT8U addr)
{
    INT8U temp, value;
    temp = addr|READ_SINGLE;//读寄存器命令
    CSN = 0;
    while (MISO);
    SpiTxRxByte(temp);
    value = SpiTxRxByte(0);
    CSN = 1;
    return value;
}
```

CC2500 软件复位

```
void RESET_CC2500(void)
{
    CSN = 0;
    while (MISO);
    SpiTxRxByte(CCxxx0_SRES);       //写入复位命令
    while (MISO);
    CSN = 1;
}
```

CC2500 初始化设置

```
const RF_SETTINGS rfSettings =
{
    0x00,
    0x07,    // FSCTRL1   Frequency synthesizer control.
    0x00,    // FSCTRL0   Frequency synthesizer control.
    0x5C,    // FREQ2     Frequency control word, high byte.
    0x58,    // FREQ1     Frequency control word, middle byte.
    0x9D,    // FREQ0     Frequency control word, low byte.
    0x0E,    // MDMCFG4   Modem configuration.
    0x3B,    // MDMCFG3   Modem configuration.
    0x73,    // MDMCFG2   Modem configuration.
    0x42,    // MDMCFG1   Modem configuration.
    0xF8,    // MDMCFG0   Modem configuration.

    0x00,    // CHANNR    Channel number.
    0x00,
 // DEVIATN   Modem deviation setting (when FSK modulation is enabled).
    0xB6,    // FREND1    Front end RX configuration.
    0x10,    // FREND0    Front end RX configuration.
    0x18,    // MCSM0     Main Radio Control State Machine configuration.
    0x1D,    // FOCCFG    Frequency Offset Compensation Configuration.
    0x1C,    // BSCFG     Bit synchronization Configuration.
    0xC7,    // AGCCTRL2  AGC control.
    0x00,    // AGCCTRL1  AGC control.
    0xB2,    // AGCCTRL0  AGC control.
    0xCA,    // FSCAL3    Frequency synthesizer calibration.
    0x0A,    // FSCAL2    Frequency synthesizer calibration.
    0x00,    // FSCAL1    Frequency synthesizer calibration.
    0x11,    // FSCAL0    Frequency synthesizer calibration.
    0x59,    // FSTEST    Frequency synthesizer calibration.
    0x88,    // TEST2     Various test settings.
    0x31,    // TEST1     Various test settings.
    0x0B,    // TEST0     Various test settings.
    0x0B,    // IOCFG2    GDO2 output pin configuration.
    0x06,    // IOCFG0D   GDO0 output pin configuration.
    0x05,    // PKTCTRL1  Packet automation control.  //地址检测
    0x45,    // PKTCTRL0  Packet automation control.
    //可变长数据包,通过同步词汇后的第一个位置配置数据包长度
    0x0A,      // 数据包过滤时使用的地址
    0xFF     // PKTLEN    Packet length.    最大
};
```

联系电话：13704018223 陈 工 E-mail: chj_006@sina.com
在线咨询：QQ:35625400 474882985 MSN:1188mm88@hotmail.com

数据接收流程操作

```
INT8U halRfReceivePacket(INT8U *rxBuffer, INT8U *length)
{
    INT8U status[2];
    INT8U packetLength;
   INT8U i=(*length)*4;  // 具体多少要根据 datarate 和 length 来决定
    halSpiStrobe(CCxxx0_SRX);       //进入接收状态
   delay(2);
   while (GDO0)
   {
       delay(2);
       --i;
       if(i<1)
          return 0;
   }
    if ((halSpiReadStatus(CCxxx0_RXBYTES) & BYTES_IN_RXFIFO))
        //如果接的字节数不为 0
   {
        packetLength = halSpiReadReg(CCxxx0_RXFIFO);
          //读出第一个字节，此字节为该帧数据长度
        if (packetLength <= *length)
          //如果所要的有效数据长度小于等于接收到的数据包的长度
       {
            halSpiReadBurstReg(CCxxx0_RXFIFO, rxBuffer, packetLength);
          //读出所有接收到的数据
            *length = packetLength;
          //把接收数据长度的修改为当前数据的长度
// Read the 2 appended status bytes (status[0] = RSSI, status[1] = LQI)
            halSpiReadBurstReg(CCxxx0_RXFIFO, status, 2);
          //读出 CRC 校验位
          halSpiStrobe(CCxxx0_SFRX);      //清洗接收缓冲区
            return (status[1] & CRC_OK);  //如果校验成功返回接收成功
        }
        else
       {
            *length = packetLength;
            halSpiStrobe(CCxxx0_SFRX);     //清洗接收缓冲区
            return 0;
        }
    }
   else
   return 0;
```

联系电话：13704018223 陈 工 E-mail: chj_006@sina.com
在线咨询：QQ:35625400 474882985 MSN:1188mm88@hotmail.com

```
}
```

数据发送流程操作

```
void halRfSendPacket(INT8U *txBuffer, INT8U size)
{
    halSpiWriteReg(CCxxx0_TXFIFO, size);
    halSpiWriteBurstReg(CCxxx0_TXFIFO, txBuffer, size); //写入要发送的数据
    halSpiStrobe(CCxxx0_STX);       //进入发送模式发送数据
    // Wait for GDO0 to be set -> sync transmitted
    while (!GDO0);
    // Wait for GDO0 to be cleared -> end of packet
    while (GDO0);
    halSpiStrobe(CCxxx0_SFTX);
}
```

无线应用注意事项

(1) 无线模块的 VCC 电压范围为 1.8V-3.6V 之间，不能在这个区间之外，超过 3.6V 将会烧毁模块。推荐电压 3.3V 左右。

(2) 除电源 VCC 和接地端，其余脚都可以直接和普通的 51 单片机 IO 口直接相连，无需电平转换。当然对 3V 左右的单片机更加适用了。

(3) 硬件上面没有 SPI 的单片机也可以控制本模块，用普通单片机 IO 口模拟 SPI 不需要单片机真正的串口介入，只需要普通的单片机 IO 口就可以了，当然用串口也可以了。模块按照接口提示和母板的逻辑地连接起来

(4) 标准 DIP 插针，如需要其他封装接口，或其他形式的接口，可联系我们定做。

(5) 任何单片机都可实现对无线模块的数据收发控制，并可根据我们提供的程序，然后结合自己擅长的单片机型号进行移植;

(6) 频道的间隔的说明：实际要想 2 个模块同时发射不相互干扰，两者频道间隔应该至少相差 1MHZ，这在组网时必须注意,否则同频比干扰。

(7) 实际用户可能会应用其他自己熟悉的单片机做为主控芯片，所以，建议大家在移植时注意以下 4 点：

A:确保 IO 是输入输出方式，且必须设置成数字 IO;

B:注意与使用的 IO 相关的寄存器设置，尤其是带外部中断、带 AD 功能的 IO，相关寄存器一定要设置好;

C:调试时先写配置字，然后控制数据收发

D:注意工作模式切换时间

我们的承诺

最后，欢迎您使用我们的产品，在应用中有技术问题请及时向我们联系，我们会予以技术知道，同时运输中出现产品问题我们会全面责任并予以更换。

愿与您一起走向成功

联系电话：13704018223 陈 工 E-mail: chj_006@sina.com
在线咨询：QQ:35625400 474882985 MSN:1188mm88@hotmail.com