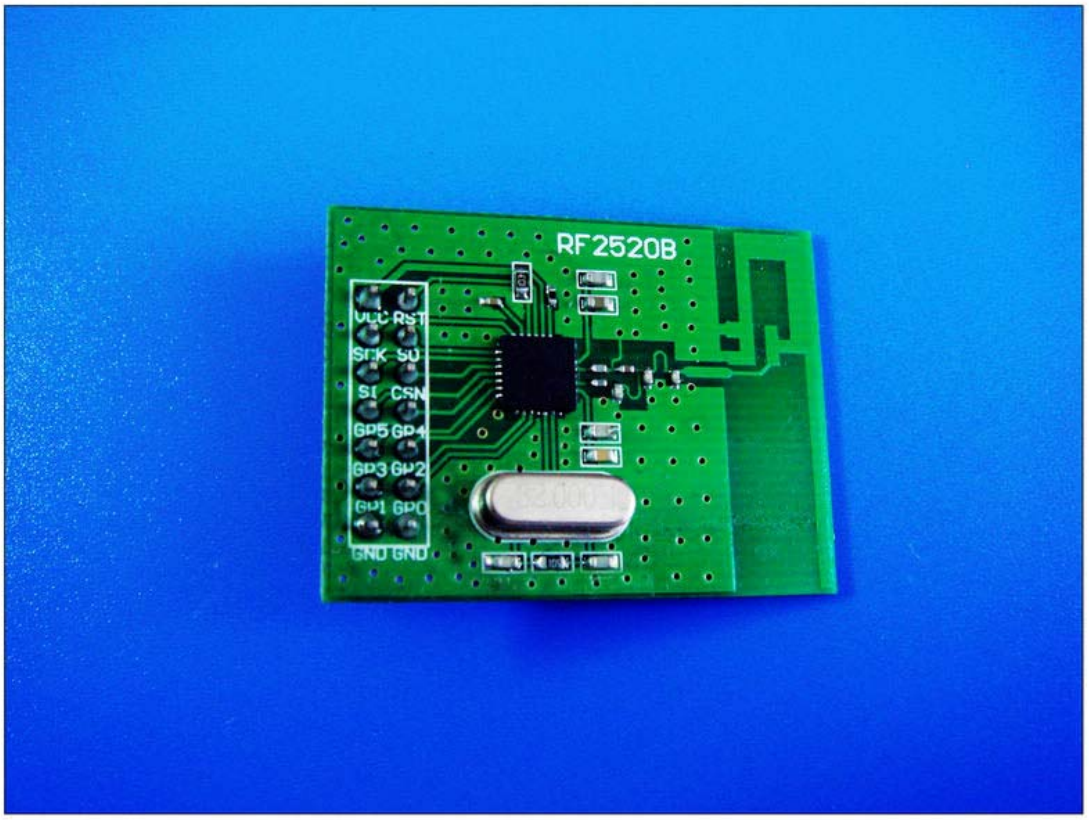
CC2520 无线模块

用户手册



CC2520 模块(尺寸：26*35mm　板厚：1mm)

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

产片介绍

CC2520是德州仪器(TI)推出2.4GHz免授权ISM频带专用的第2代ZigBee/IEEE 802.15.4无线射频收发器，凭借业界一流的选择性、出色的链路层、以及低工作电压等出色特性，CC2520还具有非常高的稳定性，2.4GHz 无线电理想适用于专有无线链路与网络， CC2520模块工作在2394-2507MHz 的ISM 频段由一个完全集成的频率调制器一个带解调器的接收器一个功率放大器一个晶体震荡器和一个调节器组成。可自动产生前导码 ， CRC 可以通过SPI 接口进行编程配置。CC2520 提供丰富的硬件支持电路，如封包处理、数据缓冲、爆发传输、数据加密、数据验证、净信道评估(CCA)、链路质量指示和封包时间信息，大幅减轻主机控制器的作业负荷。

基本特点

(1) 工作在2394-2507 MHz的ISM和SRD频段.

-采用直接序列扩频DSSS方式.

-工作速率250kbps，码片速率2 MChip/s.

-使用 O-QPSK调制方式.

-较低的电流消耗

RX(接收数据帧):18.5 mA

RX(等待接收数据):22.3 mA

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

TX(输出功率 5dBm):33.6 mA

TX(输出功率 0dBm):25.8 mA

-高灵敏度（-98dBm）.

-抗邻频道干扰能力强(49dB)

-内部集成有VCO、LNA、PA以及电源整流器.

-采用低电压供电(1.8～3.8V).

-输出功率编程可控，最大输出功率5dBm

(2) IEEE802.15.4-2006 标准 MAC 层硬件支持.

-前导码与同步字段自动生成与检测.

-CRC-16 自动生成与检测.

-帧过滤的支持，可选择接收帧的类型

-空闲信道检测，能量检测、接收信号强度与链路质量指示.

-128 位 AES 安全特性支持

(3)采用 4 线 SPI 标准接口,便于 MCU 配置.

(4)768字节片内RAM，独立的128字节RX和128字节TX数据FIFO.

典型应用场合

无线传感器网络

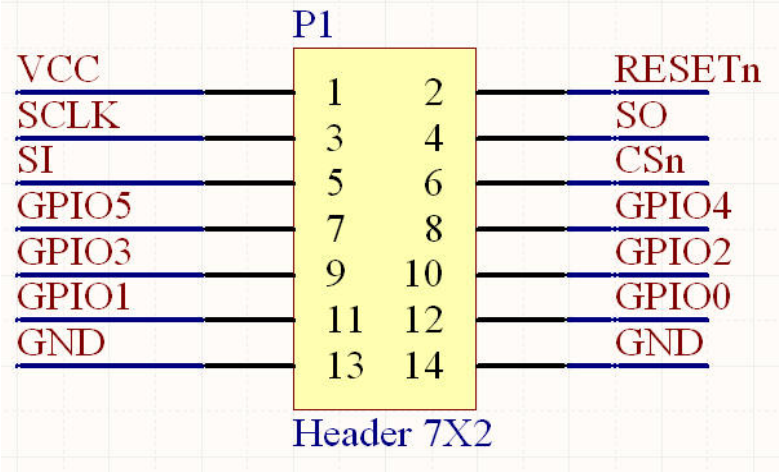
住宅、建筑物（智能家居）控制

工业仪器仪表无线数据采集和控制

无线鼠标、无线键盘、无线类玩具等消费电子

无线门禁、物流跟踪、仓库巡检等 RFID 有源电子标

模块接口



模块引脚功能

引脚	引脚名	引脚类型	描述
1	VCC	电源输入	1.9V-3.6V之间
2	RESET	数字输入	复位，低电平有效
3	SCLK	数字输入	SPI，时钟输入
4	SO	数字输出	SPI数据输出
5	SI	数字输入	SPI数据输入
6	CSn	数字输入	SPI片选
7-12	GPIOn	数字输入输出	可配置GPIO
13-14	GND	电源地	模拟接地

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

编程指南

用CC2520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CC2520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。以下为范例程序中的部分相关代码段。

CC2520 寄存器读写配置

CC2520 通过 SPI 接口与单片机通讯，因此必须首先了解 SPI 接口。SPI 外围串行接口由四条线构成：

MOSI 主机输出从机输入　　（主机写操作）

MISO 主机输入从机输出　　（主机读操作）

SCK 串行时钟信号，由主机控制

CSN 片选信号，低电平有效

SPI 读操作代码

```
uint8 SPI_Read(void)
{
    uint8 i,rxdata;
    rxdata = 0x00;
    for (i = 0;i < 8;i++)
```

```
    {
        rxdata = rxdata<<1;
        SCLK_ON();
        if (MISO_IN)
       {
            rxdata |= 0x01;
        }
        else
        {
            rxdata &= ~0x01;
        }
        SCLK_OFF();
     }
     return rxdata;
}
```

SPI 写操作代码

```
void SPI_Write(uint8 txdata)
{
    uint8 i;
    for (i = 0;i < 8;i++)
    {
        if (txdata&0x80)
        {
            MOSI_ON();
        }
        else
        {
            MOSI_OFF();
        }
        SCLK_ON();
        txdata = txdata<<1;
        SCLK_OFF();
    }
}
```

CC2520 配置寄存器读操作

```
uint8 CC2520_ReadReg(uint8  addr)
{
    uint16 value;
    CSN_OFF();
    SPI_Write(addr|REG_READ);
    value = SPI_Read();
    CSN_ON();
    return value;
}
```

CC2520 配置寄存器写操作

```
void  CC2520_WriteReg(uint8  addr, uint8 value)
{
    CSN_OFF();
    SPI_Write(addr|REG_WRITE);
    SPI_Write(value);
    CSN_ON();
}
```

CC2520 RAM 读操作

```
uint8 CC2520_ReadRAM(uint8 addrH,uint8 addrL)
{
    uint8 value;
    CSN_OFF();
    SPI_Write(addrH|MEM_READ);
    SPI_Write(addrL);
    value = SPI_Read();
    CSN_ON();
    return value;
}
```

CC2520 RAM 写操作

```
void CC2520_WriteRAM(uint8 addrH,uint8 addrL, uint8 value)
{
```

```
    CSN_OFF();
    SPI_Write(addrH|MEM_WRITE);
    SPI_Write(addrL);
    SPI_Write(value);
    CSN_ON();
}
```

CC2520 初始化

```
void CC2520_Init(void)
{
    RESET_OFF();
    delay_ms(10);
    RESET_ON();
    delay_ms(10);
    CC2520_Command(CMD_SXOSCON);
    delay_ms(10);
CC2520_PSDU[1] =
(PAN_ID_COMPRESSION<<6)|(ACKNOWLEDGMENT_REQUEST<<5)|
(FRAME_PENDING<<4)|(SECURITY_ENABLE<<3)|(FRAME_TYPE_DATA<<0);
CC2520_PSDU[2] =
(SOURCE_ADDRESSING_MODE<<6)|(FRAME_VERSION<<4)|
(DEST_ADDRESSING_MODE<<2);
    CC2520_PSDU[ 3 ] = SEQUENCE_NUMBER;
    CC2520_PSDU[ 4 ] =  CC2520_Destination_PANID[0];
    CC2520_PSDU[ 5 ] =  CC2520_Destination_PANID[1];
    CC2520_PSDU[ 6 ] =  CC2520_Destination_IEEEAddr[0];
    CC2520_PSDU[ 7 ] =  CC2520_Destination_IEEEAddr[1];
    CC2520_PSDU[ 8 ] =  CC2520_Destination_IEEEAddr[2];
    CC2520_PSDU[ 9 ] =  CC2520_Destination_IEEEAddr[3];
    CC2520_PSDU[10] =  CC2520_Destination_IEEEAddr[4];
    CC2520_PSDU[11] =  CC2520_Destination_IEEEAddr[5];
    CC2520_PSDU[12] =  CC2520_Destination_IEEEAddr[6];
    CC2520_PSDU[13] =  CC2520_Destination_IEEEAddr[7];
    CC2520_PSDU[14] =  CC2520_Source_PANID[0];
    CC2520_PSDU[15] =  CC2520_Source_PANID[1];
    CC2520_WriteRAM(0x03,RAM_PANID,  CC2520_Source_PANID[0]);
    CC2520_WriteRAM(0x03,RAM_PANID+1, CC2520_Source_PANID[1]);
    CC2520_PSDU[16] =  CC2520_Source_IEEEAddr[0];
    CC2520_PSDU[17] =  CC2520_Source_IEEEAddr[1];
    CC2520_PSDU[18] =  CC2520_Source_IEEEAddr[2];
    CC2520_PSDU[19] =  CC2520_Source_IEEEAddr[3];
```

```
    CC2520_PSDU[20 ] = CC2520_Source_IEEEAddr[4];
    CC2520_PSDU[21] =  CC2520_Source_IEEEAddr[5];
    CC2520_PSDU[22] =  CC2520_Source_IEEEAddr[6];
    CC2520_PSDU[23] =  CC2520_Source_IEEEAddr[7];
    CC2520_WriteRAM(0x03, RAM_IEEEADR,CC2520_Source_IEEEAddr[0]);
    CC2520_WriteRAM(0x03, RAM_IEEEADR+1,CC2520_Source_IEEEAddr[1]);
    CC2520_WriteRAM(0x03, RAM_IEEEADR+2, CC2520_Source_IEEEAddr[2]);
    CC2520_WriteRAM(0x03, RAM_IEEEADR+3, CC2520_Source_IEEEAddr[3]);
    CC2520_WriteRAM(0x03, RAM_IEEEADR+4, CC2520_Source_IEEEAddr[4]);
    CC2520_WriteRAM(0x03, RAM_IEEEADR+5, CC2520_Source_IEEEAddr[5]);
    CC2520_WriteRAM(0x03, RAM_IEEEADR+6, CC2520_Source_IEEEAddr[6]);
    CC2520_WriteRAM(0x03, RAM_IEEEADR+7, CC2520_Source_IEEEAddr[7]);
    CC2520_WriteReg(GPIOCTRL0,0x2A);
    CC2520_WriteReg(GPIOCTRL1,0x27);
    CC2520_Command(CMD_SFLUSHRX);
    CC2520_Command(CMD_SFLUSHTX);
    delay_ms(10);

}
```

CC2520 FIFO 发送流程

FIFO 写数据操作

```
void  CC2520_WriteTXFIFO(void)

{

    uint8 i;

    CC2520_Command(CMD_SFLUSHTX);

    CSN_OFF();

    SPI_Write(TXFIFO_WRITE);

    SPI_Write(CC2520_PSDU[0]);

    for(i=0;i<CC2520_PSDU[0];i++)
```

```
    {
        SPI_Write(CC2520_PSDU[1+i]);
    }
    CSN_ON();
}
```

FIFO 数据发送操作

```
void CC2520_TxPacket(void)
{
    CC2520_Command(CMD_SRFOFF);
    CC2520_Command(CMD_STXON);
    while(!GPIO0_IN);
    while(GPIO0_IN);
}
```

CC2520 FIFO 接收流程

接收模式设置

```
void CC2520_SetRxMode(void)
{
    CC2520_Command(CMD_SRFOFF);
    CC2520_Command(CMD_SRXON);
}
```

FIFO 接收数据

```
uint8 CC2520_RxPacket(void)
```

```
{
    if((!GPIO0_IN)&&(GPIO1_IN))
    {
        return TRUE;
    }
        return FALSE;
 }
```

收到数据后读取 FIFO 数据

```
void CC2520_ReadRXFIFO(void)
{
    uint8 i;
    CSN_OFF();
    SPI_Write(RXFIFO_READ);
    CC2520_PSDU[0] = SPI_Read();
    for(i=0;i<CC2520_PSDU[0];i++)
    {
        CC2520_PSDU[1+i] = SPI_Read();
    }
    CSN_ON();
    CC2520_Command(CMD_SFLUSHRX);
}
```

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