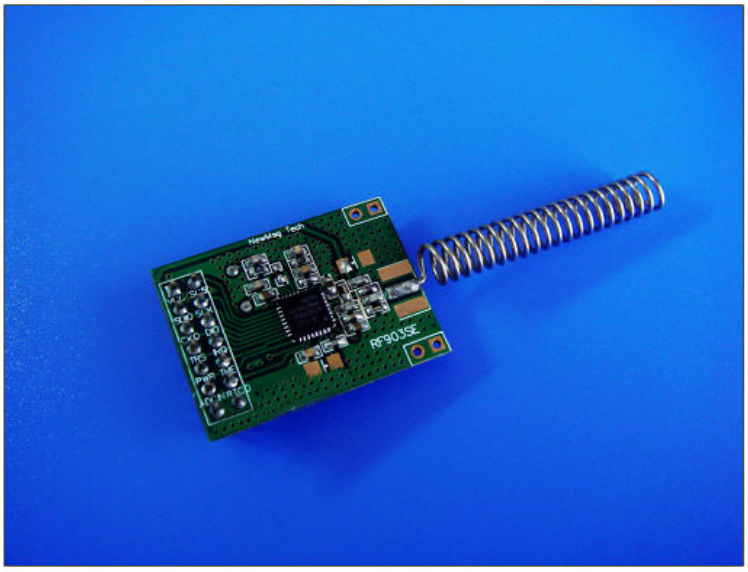
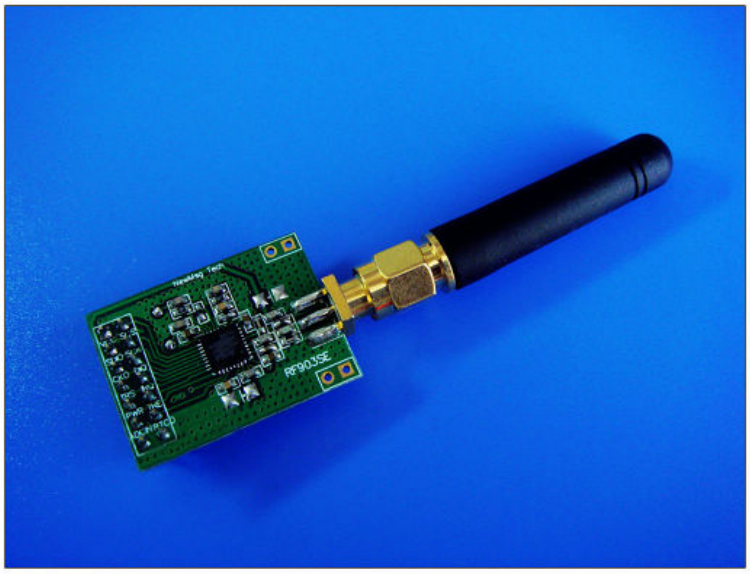
A7102 无线模块

用户手册





A7102 模块（尺寸：30mm X 22mm 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

产品介绍

台湾笙科公司出品的 A7102 采用 32 引脚 QFN32 5x5 封装，是适用于 315MHZ、433MHZ、868MHZ，915MHZ ISM 频段的 ASK/FSK 射频双向 CMOS 工艺的单片射频集成电路，A7102 双向无线数据收发器的出现彻底打破亚洲人在 1GHZ 以下无线产品只能依靠欧美国家的被动局面。

基本特点

(1) 433Mhz 开放 ISM 频段免许可证使用；

(2) 最高工作速率 150Kb/s，高效 FSK 调制，抗干扰能力强，特别适合工业控制场合；

(3) 可软件设置频率、地址，适合跳频多点通信应用场合；

(4) 内置硬件 CRC 检错和地址码控制；

(5) 低功耗 2.2 - 3.6V 工作，休眠状态仅为 2uA 可满足低功耗设备的要求 ；

(6) 收发模式切换时间< 650us；

(7) 数据收发状态指示，可以和任何 MCU 配合使用；

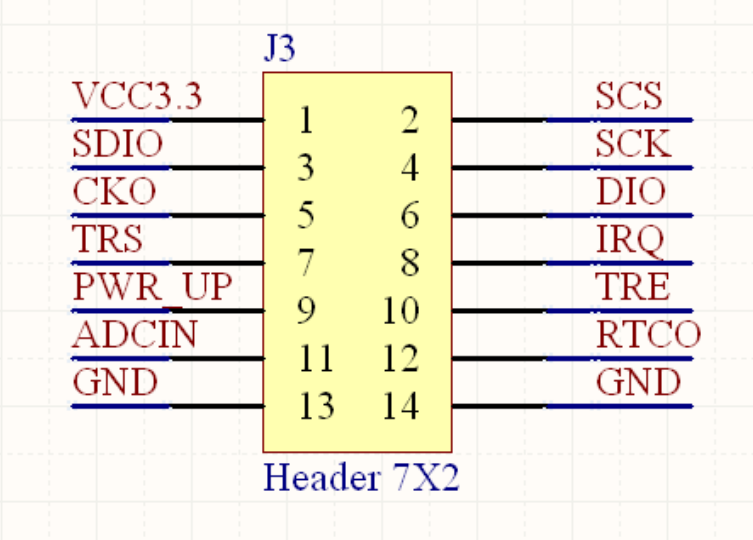
(8) TX Mode: 在+10dBm 情况下，电流为 34mA; RX Mode: 12-14mA;

(9) SPI 编程接口，收发数据长度高达 64 字节；

(10) 功率最大为+15dbm 具有发射距离远优势；

(11) 标准 DIP 间距接口，便于嵌入式应用；

模块接口说明



引脚功能

管脚	名称	管脚功能	说明
1	VCC	电源	3.3V电源
2	SCS	数字输入	SPI片选
3	SDIO	数字双向IO口	SPI双向数据线
4	SCK	SPI时钟	SPI时钟
5	CKO	始终输出	未启用
6	DIO	数据输入输出	未启用
7	TRS	数字输入	接收发送模式选择
8	IRQ	数字输出	数据完成指示
9	PWR_UP	数字输入	上电
10	TRE	数字输入	接收发送使能
11	ADCIN	外部输入ADC	未启用
12	RTCO	32K时钟输出	未启用
13	GND	地	接地
14	GND	地	接地

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。

模块电气参数

A7102模块性能参考数据

参数	数值	单位
最低工作电压	3.0	V
最大发射功率	15	dBm
最大数据传输率	150	kbps
输出功率为+10 dBm 时工作电流	34	mA
接收模式时工作电流	14	mA
温度范围	-40 to +85	
典型灵敏度	-110	dBm

A7102 工作流程

A7102 一共有两种工作模式，FIFO mode 和 Direct mode。通过 Mode control 寄存器的 FMS 位可设置，0 为时是 Direct mode（直接模式），为 1 时是 FIFO mode。一般不推荐用直接模式。

在 FIFO 模式下，A7102 自动处理字头和 CRC 校验码。在接收数据时，自动把字头和 CRC 校验码移去。在发送数据时，自动加上字头和 CRC 校验码，当发送过程完成后，IRQ 引脚通知微处理器数据发射完毕。

A7102 收发流程

FIFO TX 发送流程

A. 当微控制器有数据要发送时，通过 SPI 接口，按时序把接收机的地址和要发送的数据送传给 A7102，SPI 接口的速率在通信协议和器件配置时确定；

B. 微控制器置 Modecontrol 寄存器，且 TRS 为高，激发 A7102 的 FIFO 发送模式；

C. A7102 的 FIFO 模式发送：

(1) 射频寄存器自动开启；

(2) 数据打包(加字头和 CRC 校验码)；

(3) 发送数据包；

(4) 当数据发送完成，IRQ 有相应指示（具体可配置）；

D. A7102 发送过程完成，可选着进入任何模式，可通过 SPI 或管脚控制。

FIFO RX 接收流程

A. 当通过 SPI 指令（或者管脚控制）使 A7102 进入接收模式；

B. A7102 不断监测，等待接收数据；

C. 当 RF935 检测到同一频段的载波时，载波检测引脚被置高（根据配置 IRQ 有不同的表现）；

D. 当一个正确的数据包接收完毕， A7102 自动移去字头、地址和 CRC 校验位，然后把 IRQ 引脚置为高

E. 微控制器通过 SPI 口，以一定的速率把数据移到微控制器内；

A7102 功能配置

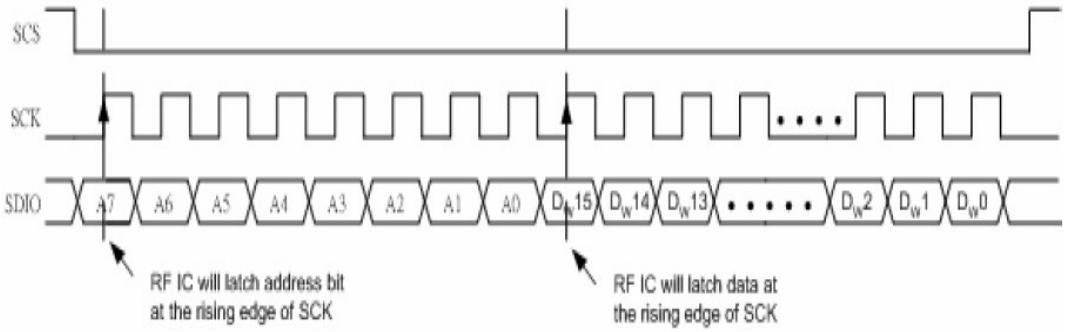
所有配置字都是通过 SPI 接口送给 A7102。SIP 接口的工作方式可通过 SPI 指令进行设置。

SPI 指令介绍

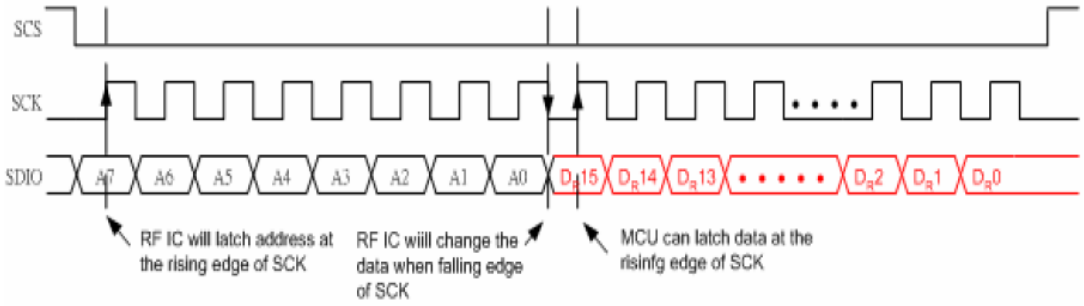
当SCS为低时,SPI接口开始等待一条指令。任何一条新指令均由SCS 的由高到低的转换开始。SPI 接口相关命令见下表：

Address Byte								说明
b7	b6	b5	B4	b3	b2	b1	b0	
0	0	0	X	A3	A2	A1	A0	Write data to control register A[3:0]
1	0	0	X	X	X	X	X	Read out data A[3:0] from control register
0	0	1	X	X	X	X	X	Write ID code command
1	0	1	X	X	X	X	X	Read out ID code command
0	0	1	X	X	X	X	X	TX FIFO write command
1	1	0	X	X	X	X	X	RX FIFO read command
1	1	1	X	X	X	X	X	RF chip Reset command
0	0	1	0	X	X	X	X	TX FIFO address pointer reset command
1	1	1	0	X	X	X	X	RX FIFO address pointer reset command

SPI 时序分析



SPI 写操作时序图



SPI 读操作时序图

A7102 编程指南

使用A7102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，读者可以向我们公司索取技术资料和技术支持，同时为便于用户开发，我们提供配套评估套件，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 写操作函数

```
void    Byte_Write(uchar byte)
{
        uchar i;
        declare_sdo_output();
        for(i = 0x00; i < 0x08; i++)
        {
            if(byte&0x80)
                RF903_Sdi = positive;
            else
                RF903_Sdi  = negative;
            delay_RF903_us();
            RF903_Sck = positive;
            delay_RF903_us();
            RF903_Sck = negative;
            delay_RF903_us();
            byte <<= 0x01;
        }
}
```

SPI 读操作函数

```
uchar   Byte_Read(void)
{
        uchar i;
        uchar byte;
        declare_sdo_input();
        for(i = 0x00; i < 0x08; i++)
        {
            byte<<= 0x01;
            delay_RF903_us();
            if(RF903_Sdo)
                byte|= 0x01;
```

```
            else
                byte&= 0xfe;
            RF903_Sck= positive;
            delay_RF903_us();
            RF903_Sck= negative;
        }
        return    (byte);
}
```

A7102 软件复位操作

```
void    RF903_Reset_Chip(void)
{
        RF903_Scs = negative;
        Byte_Write(0x7a);       /* 写 RF903-Reset 命令 */
        RF903_Scs = positive;
}
```

向指定寄存器写数据操作

```
void    RF903_Register_Write(uchar addr,uint parameter)
{
        RF903_Sck = negative;
        delay_RF903_us();
        RF903_Scs = negative;
        addr &= 0x0f;
        Byte_Write(addr);       /* 写 RF903-参数配置命令 */
        Word_Write(parameter); /* 写 RF903-参数配置数据 */
        RF903_Scs = positive;
}
```

从指定寄存器读取数据操作

```
uint    RF903_Register_Read(uchar addr)
```

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

```
{
        uint parameter;
        RF903_Sck = negative;
        delay_RF903_us();
        RF903_Scs = negative;
        addr&= 0x0f;
        addr|= 0x80;
        Byte_Write(addr);       /* 读 RF903-参数配置命令 */
        declare_sdo_input();
        parameter= Word_Read(); /* 读 RF903-参数配置数据 */
        declare_sdo_output();
        RF903_Scs = positive;
        return (parameter); /* 返回读取数值 */
}
```

A7102 工作方式设置

```
void    RF903_Config_Chip(void)
{
        RF903_Tre = negative;
        RF903_Trs = negative;
        RF903_Register_Write(Reg_SystemClock,0x0079);
        RF903_Register_Write(Reg_PLL_I,0x0043);
        RF903_Register_Write(Reg_PLL_II,0xcfff);
        RF903_Register_Write(Reg_PLL_III,0x0000);
        RF903_Register_Write(Reg_PLL_IV,0x066c);
        RF903_Register_Write(Reg_TXI,0x1520);
        RF903_Register_Write(Reg_TXII,0x0337);
        RF903_Register_Write(Reg_RXI,0x1813);
        RF903_Register_Write(Reg_RXII,0x500b);
        RF903_Register_Write(Reg_ADC,0x0000);
        RF903_Register_Write(Reg_FIFO,0x4000|(C_FIFO_Byte-1));
        // 数据包长度 C_FIFO_Byte
        RF903_Register_Write(Reg_Code,0x155f);
```

```
        RF903_Register_Write(Reg_PinControl,0x0122);
        RF903_Register_Write(Reg_Calibration,0x4886);
       RF903_Register_Write(Reg_ModeControl,0x00e0);
       delay(80); //delay 80us
}
```

发送模式设置

```
void    RF903_Status_Transmit(void)
{
  RF903_Register_Write(Reg_ModeControl,0x00d0);
  RF903_Register_Write(Reg_PLL_II,C_Tx_Frequency);
  RF903_Register_Write(Reg_Calibration,TX_Freq_Calibration);
  RF903_Register_Write(Reg_ModeControl,0x00d8);
  while(RF903_Irq);
  while(!RF903_Irq);
}
```

接收模式设置

```
void    RF903_Status_Receiver(void)
{
    RF903_Register_Write(Reg_ModeControl,0x00c0);
    RF903_Register_Write(Reg_PLL_II,C_Rx_Frequency);
    RF903_Register_Write(Reg_Calibration,RX_Freq_Calibration);
    //RF903_Register_Write(Reg_ADC, 0x0000);
    //RF903_Register_Write(Reg_ModeControl,0x00c9);
    RF903_Register_Write(Reg_ModeControl,0x00c8);
}
```

数据发送流程操作

```
void    RF903_Send(uchar *txbuf)
{
  RF903_Reset_FifoTX();
```

```
  RF903_Fifo_Write(txbuf);
  RF903_Status_Transmit();
  RF903_Status_Receiver();
}
```

数据接收流程操作

```
uchar   RF903_Receive(uchar *rxbuf)
{
  if(1==RF903_Fifo_Read(rxbuf))
  {
    RF903_Status_Receiver();
    return 1;
  }
  return 0;
}
```

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