JF24D升级版

编程指南

V02

2010年12月9日更新

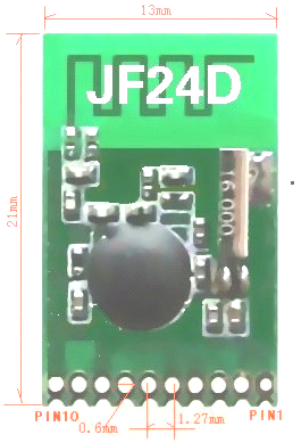
安阳新世纪电子研究所

电话：0372-5968708
传真：0372-5968993

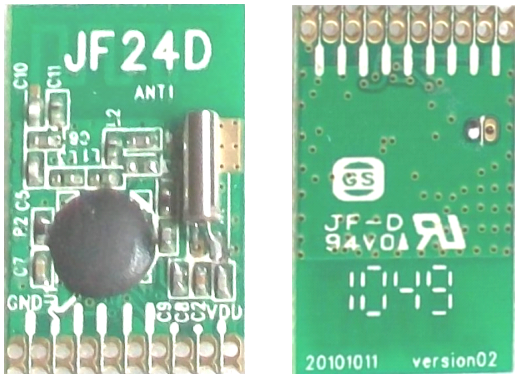
说明：

JF24D新版本与老版本尺寸及脚位定义完全一致，硬件参数有调整，软件程序需要升级，不支持和老版JF24D通讯。新版目前可以和本公司的(新版JF24D-MCU)和(JF24D-PA)通讯。

老版JF24D板图



新版JF24D板图



目录

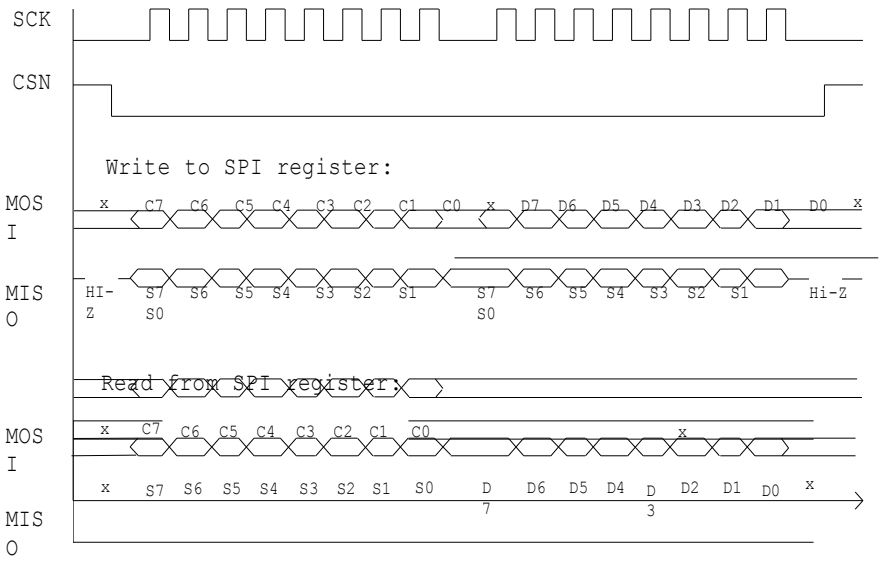
图表

1. 本文档主要内容

本文档主要目的为客户软件开发提供参考方案，其中主要包括SPI 接口操作，编程说明和参考 代码介绍。

2. SPI 接口

接口时序如下：



图表1 SPI 接口时序

当MCU 写SPI 时，MCU 应该在时钟SCK 下降沿写入数据，JF24D 会在时钟上升沿读取数据。

当MCU 读SPI 时，JF24D 应该在时钟SCK 下降沿输出数据，MCU 会在时钟上升沿读取数据。

当MCU 访问多字节的寄存器时，MCU 必须一次完成所有字节的读写。

3. 寄存器说明

JF24D 有两组寄存器Bank0 和Bank1，其中Bank 0 对应nRF24L01 的寄存器，Bank1 是JF24D的测试和功能扩展寄存器。

3.1. Bank0 寄存器

Bank0 的寄存器，详细请参见datasheet。

3.2. Bank1 寄存器

Bank1 的寄存器，详细请参见datasheet。

4. 编程说明

4.1. 初始化

(1) 上电

(2) 延时至少50 ms

(3) 如果当前不是Bank0，则切换到Bank0

(4) 写Bank0 寄存器，以下不分先后顺序：

a) CRC、中断屏蔽配置及芯片power up (REG0)

b) 使能要使用的Pipe(REG2)

c) 初始频点(REG5)

d) 设置发射功率、LNA gain、空中传输速率(REG6)

e) 设置数据包中的地址域长度(REG3)

f) 设置相应的pipe 是否支持acknowledgement(REG1)

g) 设置要使用pipe 的RX 地址(REG10-REG15) ，及TX 地址(REG16)

h) 设置要使用pipe 的数据包长度(REG17-22)

i) 如果支持ACK 模式，设置ARC 和ARD(REG4)

j) 如果要支持动态长度或者Payload With ACK，需要先给芯片发送ACTIVATE 命令（数据为0x73），然后使能动态长度或者Payload With ACK (REG28,REG29)

(5) 切换到Bank1

(6) 写Bank1 的REG0-REG8(先写高字节，再写低字节)

(7) 写Bank1 的REG9-REG13(先写低字节，再写高字节)

(8) 写Bank1 的REG14(先写低字节，再写高字节)

(9) Toggle REG4<25，26>，即bit25，bit26 先写1，再写0 (10) 延时至少10 ms

(11) 切换回Bank0

注：

(1).Bank1 的**REG0** 到**REG8** 写时序：先高字节，再低字节；每个字节从高位再到低位。

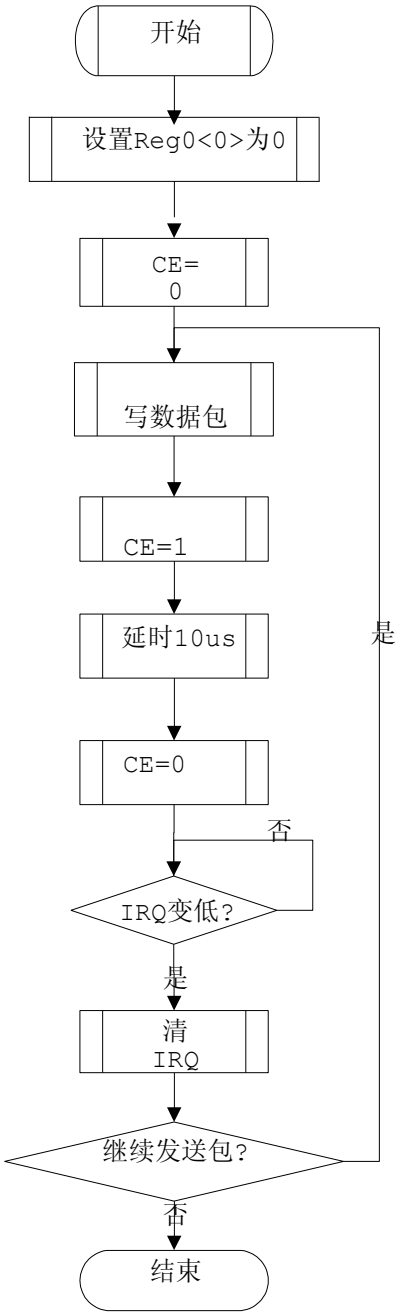
但是**Bank1** 的**REG9** 到**REG14** 及**Bank0** 的读写时序和**nRF24L01** 一致，都是先低字节，再高字节；每个字节仍然从高位到低位。

(2).Bank1 的**REG0** 和**REG1** 在**BER** 测试时可读，读时序为：先读低字节，再读高字节；每个字节从高位再低位。

4.2. 数据包发送

4.2.1. NoACK 包

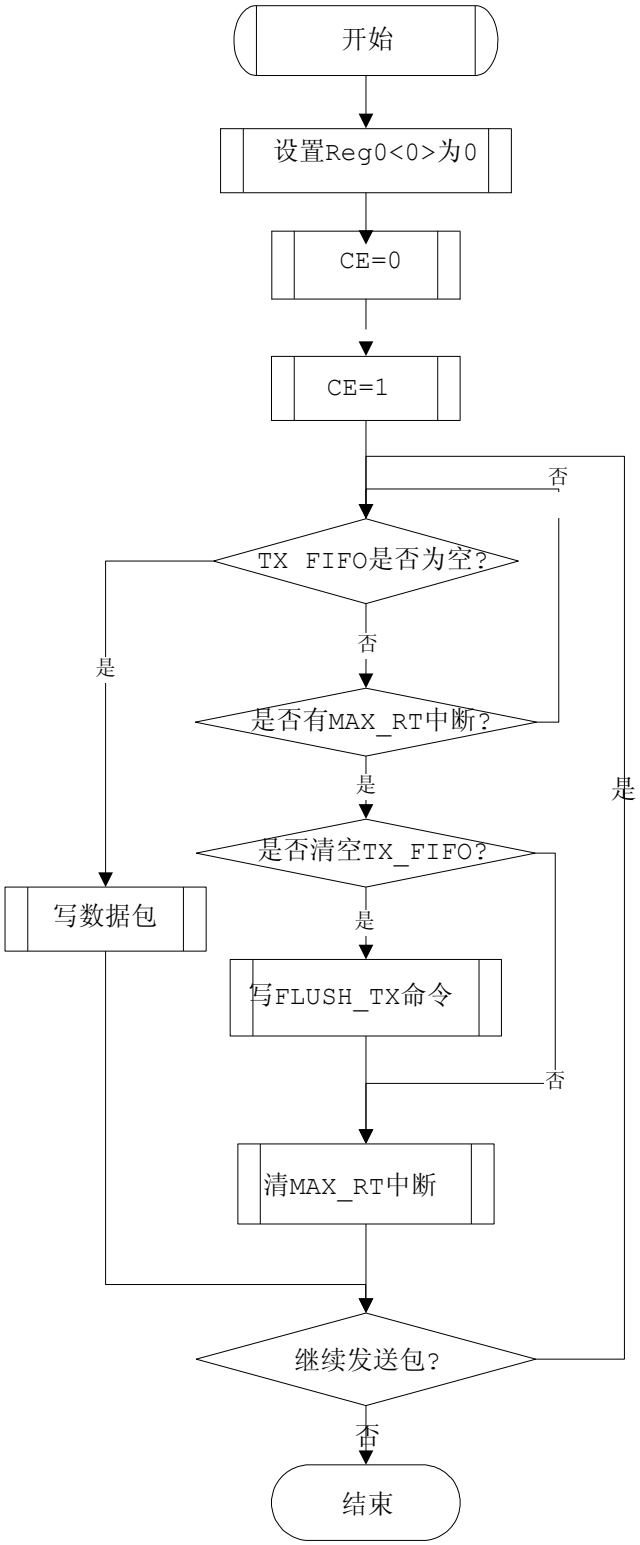
图表3 为NoACK 包发送的一般流程。



图表**2 NoACK** 包发送常见流程

4.2.2. ACK 包

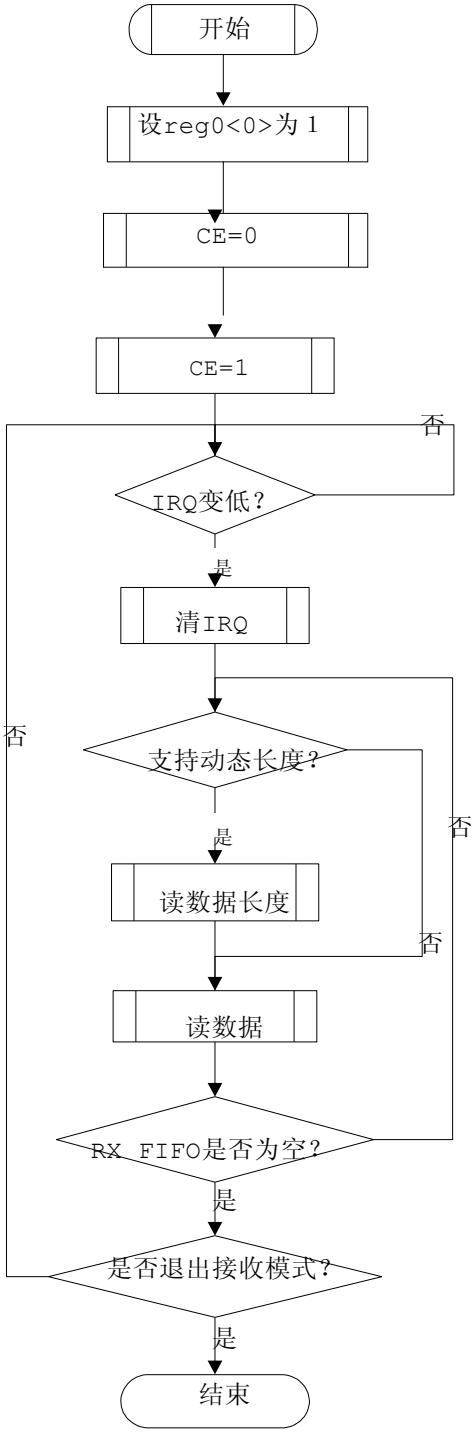
图表4 为ACK 包发送的常见流程。



图表3 ACK 包发送常见流程

4.3. 数据包接收

图表 5 为数据包的常见接收流程。



图表**4** 数据包接收常见流程

5. 参考代码

5.1. 简介

(1) 参考代码包含以下4 个文件：

a) JF24D_Initialize.c：
提供寄存器读写，芯片初始化，收发模式切换和channel 切换等函数。

b) JF24D_TxRx.c： 提供收
包和发包的函数。

c) Test_Func.c：
提供载波发射，BER 测试等测试和扩展函数。

d) JF24D.h： 包含宏定义
等的头文件

(2) 参考代码在Keil C51 uVision2 环境下编译通过。

(3) 参考代码提供的初始值是常用的典型设置(支持 6 个pipe，动态长度，NoACK 和ACK 模式，Payload With ACK 模式， 2 bytes 的CRC，1M 模式)。

(4) 用户应根据实际需要对代码进行移植并做相应修改和功能添加。

注：JF24D 的软件编程中，必须要考虑躲避干扰问题，请参考JF24D ***Application Notes*** 相关章节。

5.2. 函数

5.2.1. 函数列表

SPI_RW	写一个byte 到芯片，并返回一个byte
SPI_WRITE_Reg	写一个byte 到一个寄存器
SPI_Read_Reg	从寄存器读一个byte
SPI_Read_Buf	从寄存器读多个byte
SPI_Write_Buf	写多个byte 到一个寄存器
SwitchToRxMode	切换芯片到RX 模式
SwitchToTxMode	切换芯片到TX 模式
jf2_Initialize	芯片初始化，并进入到RX 模式
Send_Packet	发送包
Receive_Packet	接受包

SetChannelNum	设置新频点
SwitchCFG	切换到Bank1 寄存器操作，仅为载波测试、BER 测试或者其它特殊功能时使用
Carrier_Test	载波测试 开始发射载波时，调用Carrier_Test(1) 停止发射载波时，调用Carrier_Test(0)
BER_Test	调用该函数可以读出硬件计算出的BER 数值 输入参数ms：为硬件BER 持续的毫秒数 输出参数received_total_bits：为在ms 时间内，硬件接收到的所有bits 数目 输出参数received_error_bits：为在ms 时间内，硬件接收到的错误bits 数目 received_error_bits/received_total_bits 即为BER
SPI_Analog_Write_Reg	写Bank1 寄存器
SPI_Analog_Read_Reg	读Bank1 寄存器
Set_LowPower_Mode	芯片进入低输出power 模式
Close_CD_Detect	如果不需要CD 功能，可以关闭该功能，减少大约1mA 电流
Get_Chip_ID	得到芯片的ID 号

图表**5** 函数列表

注**:**背景带蓝色的部分涉及到读写**Bank1** 的寄存器，建议客户根据需要直接使用或者移植相关代码。

5.2.2. 函数说明

(1) **UINT8 SPI_RW(UINT8 value)**

描述：通过SPI 的MOSI 线向芯片写入8 个bit 数，并从MISO 线上获取8 个bit。

参数：value： 向MOSI 线上输出的8bit 数

返回值：从MISO 线上获取的8bit 数

(2) **void SPI_Write_Reg(UINT8 reg, UINT8 value)**

描述：通过SPI 向芯片的某个寄存器写入一个字节。

参数：reg： 需要写入的寄存器序号

value： 需要写入的值

返回值：无

(3) **UINT8 SPI_Read_Reg(UINT8 reg)**

描述：通过SPI 从芯片的某个寄存器读出一个字节。

参数：reg： 需要读取的寄存器序号 返回值：读出的寄存器值

(4) **void SPI_Read_Buf(UINT8 reg, UINT8 *pBuf, UINT8 length)**

描述：通过SPI 从多字节的寄存器中读出多个字节数据。

参数：reg： 需要读取的寄存器序号

pBuf： 数据存放的数组

length： 需要读取的数据长度

返回值：无

(5) **void SPI_Write_Buf(UINT8 reg, UINT8 *pBuf, UINT8 length)**

描述：通过SPI 的MOSI 线向芯片写入8 个bit 数，并从MISO 线上获取8 个bit。

参数：reg： 需要写入的寄存器序号

pBuf： 数据存放的数组

length： 需要写入的数据长度

返回值：无

(6) **void SwitchToRxMode()**

描述：将芯片切换为接收模式。

参数：无

返回值：无

(7) **void SwitchToTxMode()**

描述：将芯片切换为发送模式。

输入：无

返回值：无

(8) **void JF24D_Initialize()** 描述：完成芯片初始化，配置所有寄存器，并切换为接收模式。 参数：无

返回值：无

(9) **void Send_Packet(UINT8 type,UINT8* pbuf,UINT8 len)**

描述：发送一个数据包。

参数：type： WR_TX_PLOAD：以带ACK 模式发送数据包
W_TX_PAYLOAD_NOACK_CMD：以NOACK 模式发送数据包
pbuf： 数据所存放的数组
len： 数据包的长度
返回值：无

(10) **void Receive_Packet()** 描述：循环检测状态寄存器，若有接收到数据，则将数据从FIFO 中读出。 参数：无
返回值：无

(11) **void SetChannelNum(UINT8 ch)**
描述：设置channel 号。
参数：ch： 0-127：channel 号
返回值：无

(12) **void SwitchCFG(char _cfg)**
描述：寄存器bank 切换。
参数：_cfg： 0：切换到bank0；1：切换到bank1
返回值：无

(13) **void Carrier_Test(UINT8 b_enable)**
描述：发射连续载波模式。
参数：b_enable： 0：打开连续载波模式；1：关闭连续载波模式。
返回值：无

(14) **void BER_Test(UINT16 ms,UINT32* received_total_bits,UINT32* received_error_bits)**
描述：BER 测试函数。
参数：ms： BER 测试进行的时间，单位为毫秒。
received_total_bits：存放测试时间内接收到的总bit 数的指针。
received_error_bits：存放测试时间内接收到的错误bit 数的指针。
返回值：无

(15) **void SPI_Bank1_Write_Reg(UINT8 reg, UINT8 *pBuf)**
描述：向bank1 寄存器写入数据。

参数：reg： 需要写入的寄存器序号

pBuf： 数据存放的数组

返回值：无

(16) **void SPI_Bank1_Read_Reg(UINT8 reg, UINT8 *pBuf)**

描述：从bank1 寄存器中读出数据。

参数：reg： 需要读取的寄存器序号

pBuf： 数据存放的数组

返回值：无

(17) **void Set_LowPower_Mode()**

描述：进入低功率模式。

参数：无

返回值：无

(18) **void Close_CD_Detect()**

描述：关闭CD 检测功能。

参数：无

返回值：无

(19) **UINT8 Get_Chip_ID()**

描述：读取芯片的ID。

参数：无

返回值：芯片的ID 号

5.3. 移植顺序

(1) 添加SPI_RW，SPI_WRITE_Reg，SPI_Read_Reg，SPI_Write_Buf，SPI_Read_Buf 函数，调试SPI 读写。例如，写REG0，然后读出来，看看是否是写入的值

(2) 添加Initialize 相关的代码

(3) 调用Send_Packet 和Receive_Packet，看看能否收发包

(4) 根据应用需要，增加其它功能

5.4. 其它建议

(1) 建议调试程序时，最好是先用我们的带程序的JF24D-MCU或者JF24D-PA测试模块的性能再修改程序。

(2) 寄存器的初始值最好先用我们提供的初始值，该值和我们测试板的初始值一样。程序能够 收发包后，再修改初始值为产品需要的。