

SX126X_HC32L110C4 Demo_V8.2 程序说明文档

一 . 引脚配置

SX126X	MCU-HC32L110C4	
SCK	P15	SPI的时钟引脚
NSS	P14	SPI的片选
MISO	P23	主机输入从机输出--数据线
MOSI	P24	主机输出从机输入--数据线
REST	P25	SX126X复位引脚
BUSY	P26	SX126x忙 状态引脚
RXEN	P25	射频天线接收控制端口--高电平使能

射频天线的发送开关由射频芯片控制，用户只需配置相应的寄存器即可，只需控制接收天线，即RXEN引脚
操作方法：在 "sx126x_driver.h" 的第22行，开启宏定义"#define USE_DIO2"即可

二 . 软件说明

1. 开发环境

软件：KEIL5 V530 版本

硬件：华大单片机：HC32L110C4

程序烧录：可以使用 Jlink 的 SWD 方式，将程序烧录到测试底板中。Jlink 的 VCC GND SWCLK SWDIO 分别接测试底板的 V G C D 。测试底板的烧录口在左下角的 4P*1.27 接口。

SX1262、SX1268、LLCC68 可共用此套 Demo

测试底板型号：ASDS-V2.3

2. 软件配置

(1) SPI 配置

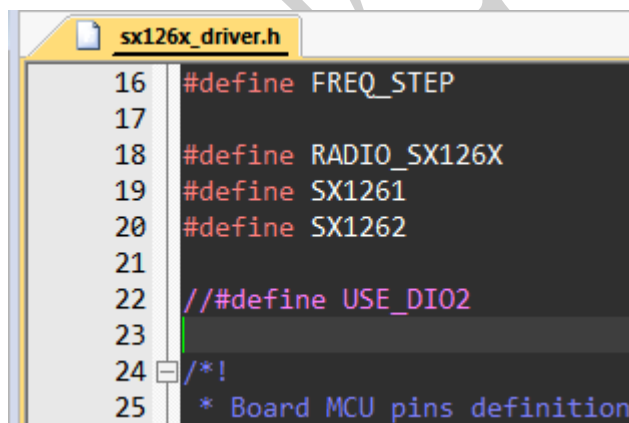
MCU 与射频芯片 SX126X 采用 SPI 通信，Demo 里面提供了硬件 SPI 和软件 SPI 两种方式，通过宏 `#define SOFT_SPI` 来控制，此宏在 `drv_spi.h` 里面。在宏前面加上 `//`，则使用硬件 SPI，不加 `//`，则使用软件 SPI。

```
6 // #define SOFT_SPI //使能软件SPI，屏蔽掉后则使用硬件SPI
```

SPI 的引脚配置在 `drv_spi.c` 和 `drv_spi.h` 文件里。

(2) 射频发送控制引脚的选择(TXEN 与 DIO2)

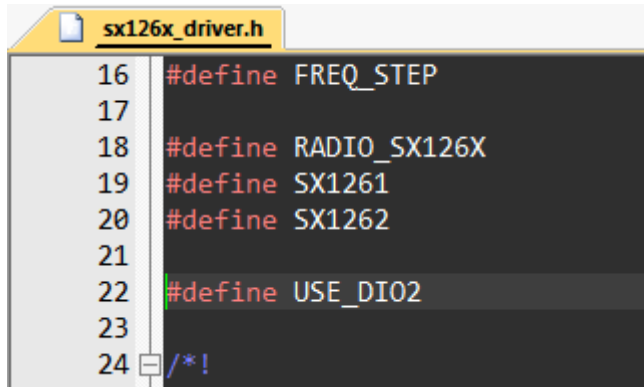
如果发送控制引脚 (TXEN) 由用户 MCU 控制，那么在程序中应该注释掉一个宏 “USE_DIO2”，如下图。在发送时，TXEN 引脚为高电平，RXEN 为低电平；在接收时，TXEN 为低电平，RXEN 为高电平。



```
16 #define FREQ_STEP
17
18 #define RADIO_SX126X
19 #define SX1261
20 #define SX1262
21
22 // #define USE_DIO2
23
24 /*!
25 * Board MCU pins definition
```

如果发送控制引脚 (TXEN) 由 RF 芯片的 DIO2 引脚自己控制，(用户 MCU 的 IO 不够用的情况下)，应该释放宏 “USE_DIO2” 即可,如下图。在这种情况下，硬件上需要将 DIO2 引脚

与 TXEN 引脚连接在一起，用户的 MCU 只需要控制 RXEN 引脚即可。发送时，只需要将 RXEN 引脚拉低，在接收时，只需要将 RXEN 引脚置高。



```
16 #define FREQ_STEP
17
18 #define RADIO_SX126X
19 #define SX1261
20 #define SX1262
21
22 #define USE_DI02
23
24 /*!
```

(3) SX126X 其他引脚配置

SX126X 的 REST ,BUSY ,RXEN 引脚在 sx126x_driver.c 和 sx126x_driver.h 文件里面

(4) 串口参数

波特率：9600；数据位：8 位；停止位：1 位

3. 程序功能

(1) 初始化

对 SX126X 初始化以后，有一段程序是测试 SPI 通信是否正常。通过读取 SX126X 的寄存器的值来判断。运用到项目中，则可以将此段程序删除掉。

```

55 //测试 SPI 通信是否成功
56 SX126xReadRegisters( REG_LR_SYNCWORD , buff , 2 );
57 Spi_ok_Falg = (buff[0] << 8 )+ buff[1];
58
59 if( Spi_ok_Falg == 0X1424 )          // Spi_ok_Falg = 0X1424 ,即SPI通信OK , 两个led 400ms闪烁 3次;
60 {
61     ledAll_tog( 3 , 400 );
62     printf("SPI-OK = %X \r\n" , Spi_ok_Falg);
63 }
64 else                                // Spi_ok_Falg != 0X1424 , SPI通信失败 , 两个LED , 一直闪烁, 100ms
65 {
66     printf("SPI-Fail = %X \r\n" , Spi_ok_Falg);
67     while(1)
68         ledAll_tog( 3 , 100 );
69 }
70

```

读取出来的值等于 0X1424，即 SPI 通信正常，接着 LED 以 400ms 周期闪烁 3 次，然后进入主程序中。如果读取出来的值，不是 0X1424，那么 SPI 通信失败，LED 就会一直闪烁，并且闪烁频率很快。

SPI 通信失败的话，先检查 SPI 的引脚是否配置正确，SPI 的速度是否设置合理。

(2) 射频收发功能

程序完成了射频的接收和发送功能，有三种工作模式：程序默认进入的自动发送功能

TX_MODE_Auto: 自动发送模式，以 500ms 的间隔，持续发出“ashining”字符串，并且 LED 发生翻转。

TX_MODE_Uart: 发送串口数据。在自动发送模式 **TX_MODE_Auto** 下，通过串口助手向测试底板发送切换发送模式的命令：AF AF 00 00 5A 00 FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 50 50 FF FF A5 0D 0A，（16 进制，32 字节）。即可从 **TX_MODE_Auto** 切换到 **TX_MODE_Uart** 模式。在此模式下，RF 发送 串口接收到的数据。

RX_MODE: 接收模式。接收相同配置下射频芯片发送的数据，然后通过串口打印出来。

只有在 **TX_MODE_Auto** 和 **RX_MODE** 模式下，才能通过按键切换模式，只要进入 **TX_MODE_Uart** 模式，就会一直处于这个模式，只能按复位按键，切换到 **TX_MODE_Auto** 模式。