Radio

1 Radio 驱动层介绍

图1.1 驱动框架

2 Radio 层接口

```
struct Radio_s
{
    void    ( *Init )( RadioEvents_t *events );
    RadioState_t ( *GetStatus )( void );
    void    ( *SetModem )( RadioModems_t modem );
    /*!...
    void    ( *SetChannel )( uint32_t freq );
    /*!...
    bool    ( *IsChannelFree )( RadioModems_t modem, uint32_t freq, int16_t rssiThresh, uint32_t maxCarrierSenseTime );
    /*!...
    uint32_t ( *Random )( void );
    /*!...
    void    ( *SetRxConfig )( RadioModems_t modem, uint32_t bandwidth,
                              uint32_t datarate, uint8_t coderate,
                              uint32_t bandwidthAfc, uint16_t preambleLen,
                              uint16_t symbTimeout, bool fixLen,
                              uint8_t payloadLen,
                              bool crcOn, bool freqHopOn, uint8_t hopPeriod,
                              bool iqInverted, bool rxContinuous );
    /*!...
    void    ( *SetTxConfig )( RadioModems_t modem, int8_t power, uint32_t fdev,...
    /*!...
    bool    ( *CheckRfFrequency )( uint32_t frequency );
    /*!...
    uint32_t  ( *TimeOnAir )( RadioModems_t modem, uint8_t pktLen );
    /*!...
    void    ( *Send )( uint8_t *buffer, uint8_t size );
    /*!...
    void    ( *Sleep )( void );
    /*!...
    void    ( *Standby )( void );
    /*!...
    void    ( *Rx )( uint32_t timeout );
    /*!...
    void    ( *StartCad )( void );
    /*!...
    void    ( *SetTxContinuousWave )( uint32_t freq, int8_t power, uint16_t time );
    /*!...
    int16_t ( *Rssi )( RadioModems_t modem );
    /*!...
    void    ( *Write )( uint16_t addr, uint8_t data );
    /*!...
    uint8_t ( *Read )( uint16_t addr );
    /*!...
    void    ( *WriteBuffer )( uint16_t addr, uint8_t *buffer, uint8_t size );
    /*!...
    void    ( *ReadBuffer )( uint16_t addr, uint8_t *buffer, uint8_t size );
    /*!...
    void    ( *SetMaxPayloadLength )( RadioModems_t modem, uint8_t max );
    /*!...
    void    ( *SetPublicNetwork )( bool enable );
    /*!...
    uint32_t  ( *GetWakeupTime )( void );
    /*!...
    void ( *IrqProcess )( void );
    /*...
    /*!...
    void    ( *RxBoosted )( uint32_t timeout );
    /*!...
    void ( *SetRxDutyCycle ) ( uint32_t rxTime, uint32_t sleepTime );
};
```

图2.1 函数接口

总共有 27 个接口函数；

调用案例：

```
{
    RadioInit,                  //初始化
    RadioGetStatus,             //获取状态
    RadioSetModem,              //设置模式
    RadioSetChannel,            //设置通道
    RadioIsChannelFree,         //通道是否空闲?
    RadioRandom,                //随机数
    RadioSetRxConfig,           //接收配置
    RadioSetTxConfig,           //发送配置
    RadioCheckRfFrequency,//检查射频频率是否合法
    RadioTimeOnAir,             //计算在数据在空中发送的时间
    RadioSend,                  //发送
    RadioSleep,                 //睡眠
    RadioStandby,               //待机
    RadioRx,                    //接收
    RadioStartCad,              //启动cad检测
    RadioSetTxContinuousWave,//发送连续波，用于功率检测
    RadioRssi,                  //读取RSSI
    RadioWrite,                 //写寄存器
    RadioRead,                  //读寄存器
    RadioWriteBuffer,           //写数组
    RadioReadBuffer,            //读数组
    RadioSetMaxPayloadLength,//设置最大负载长度
    RadioSetPublicNetwork,//设置公有、私有网络
    RadioGetWakeupTime,         //获取唤醒时间
    RadioIrqProcess,            //中断处理
    // Available on SX126x only
    RadioRxBoosted,             //设置boost
    RadioSetRxDutyCycle         //设置接收占空比
};
```

图2.2 接口函数赋值及各个函数意义

Radio 层目前包含两个文件：radio.c radio.h，具体某个芯片的驱动在 sx126x.c 里面；

Radio 的代码里没有关于 IO 口，寄存器读写等操作接口；

3 目前的问题

radio.c 中调用的赋值的函数接口是 radio.c 里写的，并不是直接调用的 sx126x.c 内部，比如

```
void RadioInit( RadioEvents_t *events )
{
    RadioEvents = events;

    SX126xInit( RadioOnDioIrq );//初始化
    SX126xSetStandby( STDBY_RC );//设置待机
    SX126xSetRegulatorMode( USE_DCDC );//使用DCDC，功率更大

    SX126xSetBufferBaseAddress( 0x00, 0x00 );//设置基地址
    SX126xSetTxParams( 0, RADIO_RAMP_200_US );//设置发送参数？200us
    SX126xSetDioIrqParams( IRQ_RADIO_ALL, IRQ_RADIO_ALL, IRQ_RADIO_NONE, IRQ_RADIO_NONE );

    // Initialize driver timeout timers
    TimerInit( &TxTimeoutTimer, RadioOnTxTimeoutIrq );//发送超时
    TimerInit( &RxTimeoutTimer, RadioOnRxTimeoutIrq );//接收超时

    IrqFired = false;
}
```

图3.1 初始化函数

初始化函数中再一次调用了 sx126xInit()，并调用了其他函数；然而在 sx1276 中又是直接调用的 sx1276 里的初始化函数

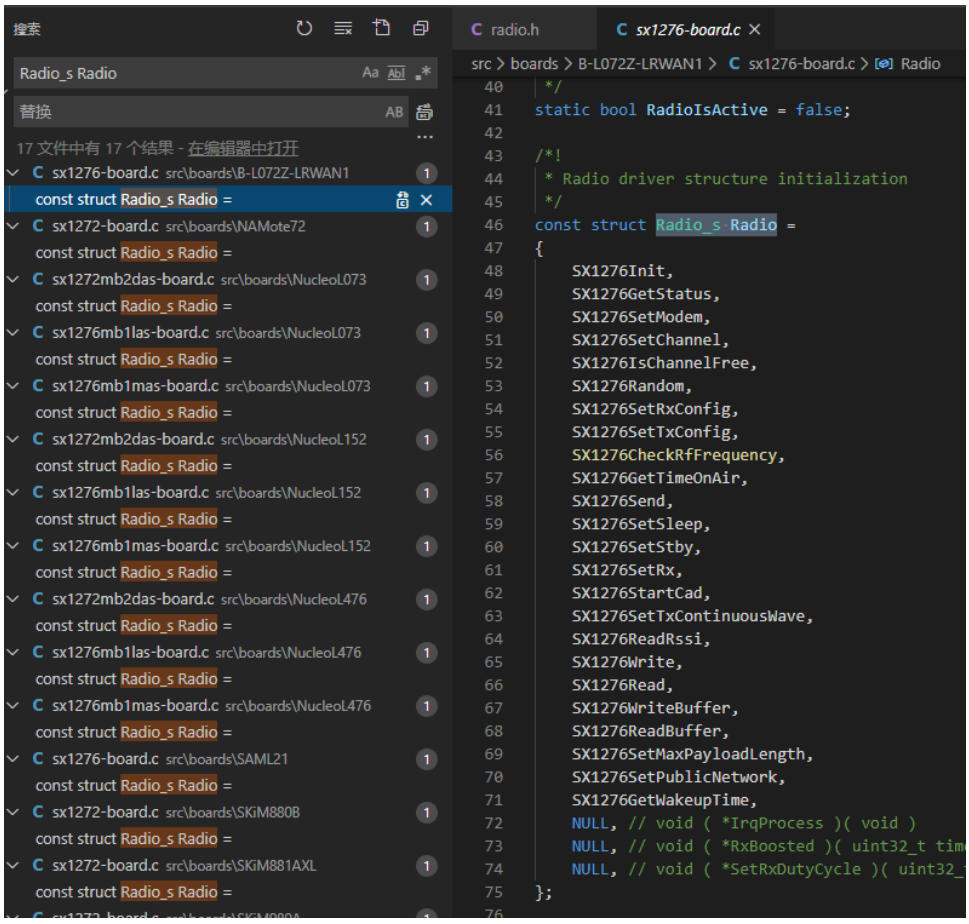

图3.2 Sx176 驱动文件

如上，sx1276 的又是在板级包中定义；该文件中包含了 IO 初始化、spi 初始

化等。

4 驱动需用到的资源

1) 定时器，超时回调；主要用于发送超时和接收超时
2) Spi 驱动：
3) Pin 驱动，需要中断；
4) 全局缓冲区数组

5 计划文件组织

用于	文件	实现功能	描述
射频抽象	lora_radio.c	射频操作的抽象，实现现有 radio 内的代码	
	lora_radio.h	头文件定义，函数、定义导出	
sx1278/6 驱动代码	sx1278.c	radio 抽象层中每个函数的具体实现	
	sx1278.h	函数导出	
sx12762 驱动代码	sx126x.c	radio 抽象层中每个函数的具体实现	现有代码和抽象层分离的不是很好，是在 1262 代码基础上又封装了一层，提提思路？
	sx126x.h	函数导出	
射频驱动配置文件	lora_radio_config.c	用于 IO 口初始化、复位芯片、SPI 接口配置、寄存器读写等。	
	lora_radio_config.h	函数导出，IO 口定义、寄存器定义	

6 最后讨论

是否要做成注册设备的方式？

比如 radio 做成设备，可以用读、写操作的；

这样是否会影响后面的协议栈移植？