LoRa 通讯范例

本章将指引用户通过 HopeDuino 进行一对 LoRa 产品收发通讯实验。LoRa 是 Semtech 特有扩频技术，HopeRF 采用其设计出一系列超高性能无线收发模块，拥有+20dBm 发射功率，-139dBm 超高灵敏度，链路预算高达 159dB，是常规同等功率收发模块的 3 倍距离（相同指标和环境下测试），具体型号有 RFM92、RFM95、RFM96、RFM98 等。

一、需要准备的工具和软件

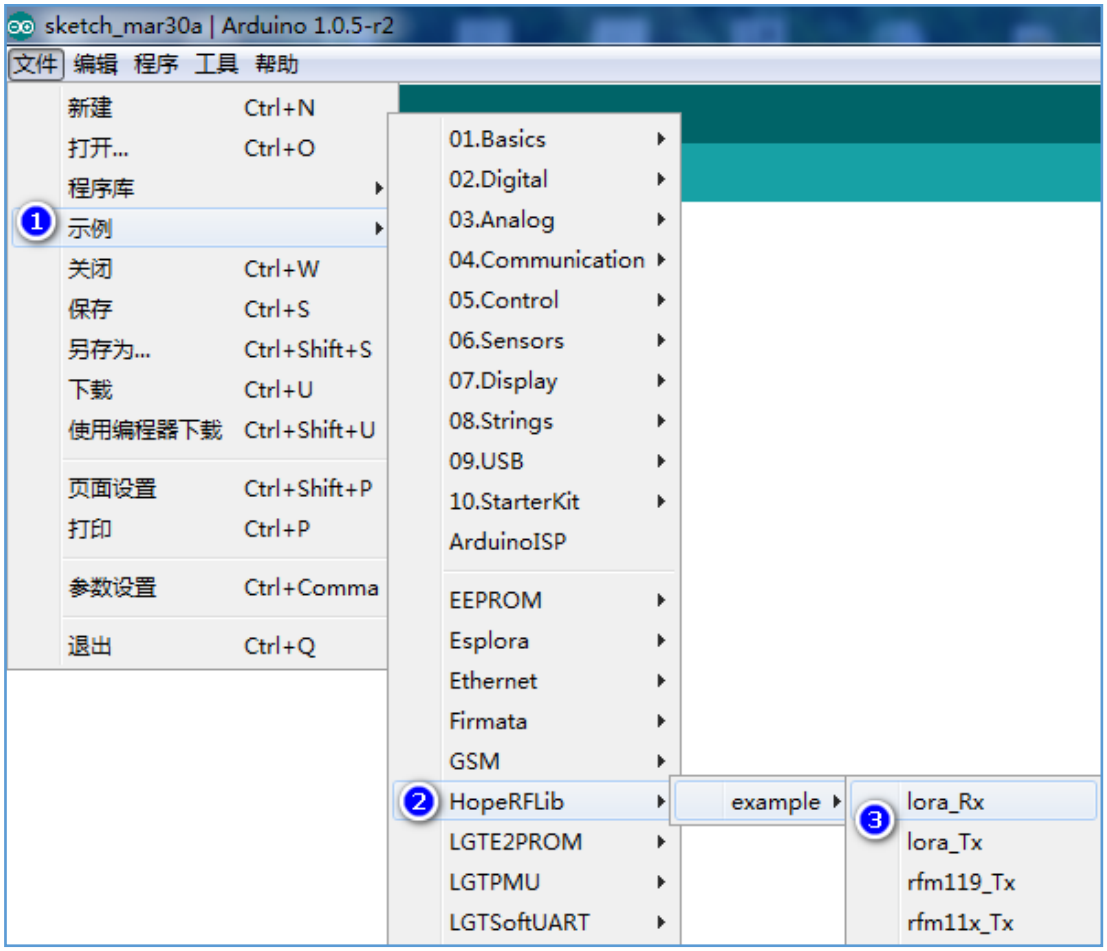
- Arduino 1.0.5 版本 IDE
- HopeDuino 板（2 个）
  （如果此前没有使用过 HopeDuino 板，请参见《AN0002-HopDuino 平台搭建指引》）
- USB 连接线（A 口转 B 口）
- 基于 RF9x 芯片设计的模块（或 RFM9x 模块）

RFM9x

二、动手实验

- 把两个 RFM9x 模块（带转换板）分别插入到各自对应的 HopeDuino 板上；
- 把两个 HopeDuino 板通过 USB 线连接到 PC；
- 打开 Arduino IDE 界面，点击【文件】→【示例】→【HopeRFLib】→【lora_Tx】，如下图操作；
- 打开 Arduino IDE 界面，点击【文件】→【示例】→【HopeRFLib】→【lora_Rx】，如下图操作。

⚠注意：如果没有在【示例】中找到【HopeRFLib】，是因为没有安装好 HopeRF 提供的 HSP，具体操作请参见《AN0002-HopDduino 平台搭建指引》



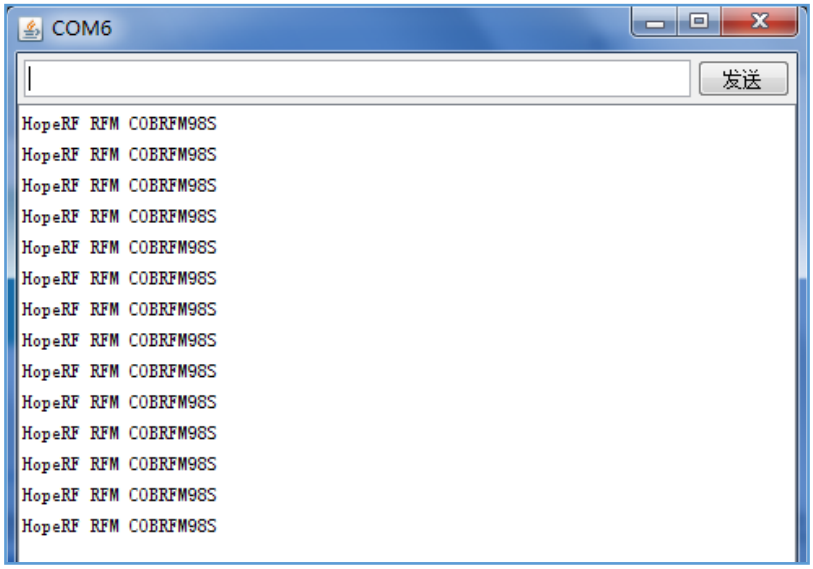
- 此时已经打开一个 Tx 的例程和一个 Rx 的例程，请根据对应的 COM 口进行编译下载程序；

注意：

1. 不清楚如何编译下载代码，请参见《AN0002-HopDuino 平台搭建指引》
2. HopeDuino 可以支持多个连接同一台 PC，需要使手动修改，所以每次下载程序需要特别留意当前界面下载是那个 COM 口的 HopeDuino。COM 口在 Arduino 界面的右下角，如下图所示：

- 两个例程下载完成后，Tx 端的 HopeDuino 就会周期性通过 RFM9x 模块发射一包数据；Rx 端的 HopeDuino 就会相应周期性通过 RFM9x 模块接收到一包数据，并同时把这包数据通过 UART（USB）上传到 PC。此时，可以把 Arduino IDE 的 COM 口设置为连接到 Rx 的 HopeDuino 板，并打开“串口监视器”，如下图所示：

- 点击“串口监视器”后，就会弹出串口助手界面，如下图所示，窗口就会显示出收到数据报文。



注意：

1. 接收例程启用 UART 库函数，关于 UART 库函数描述，请直接参见“HopeDuino_UART”库文件，它同样存储在 HopeRFLib 之中。

三、范例解释

- lora_Tx.ino 范例解释

```
#include <HopeDuino_LoRa.h>                          //调用对应库文件

loraClass radio;                                     //定义针对 LoRa 的类变量 radio
byte str[21] = {'H','o','p','e','R','F',' ','R','F','M',' ','C','O','B','R','F','M','9','8','S'};          //待发射报文
```

```
void setup()
{
 radio.Modulation          = LORA;          //调制模式为 LoRa
 radio.COB                 = RFM98;         //模块为 RFM98
 radio.Frequency           = 434000;        //目标频率 434MHz
 radio.OutputPower         = 17;            //17dBm  输出功率
 radio.PreambleLength = 16;                 //16Byte Preamble
 radio.FixedPktLength    = false;           //可变长度报文格式
 radio.PayloadLength     = 21;              //报文长度 21Byte
 radio.CrcDisable          = true;          //不用 CRC

 radio.SFSel               = SF9;           //扩频因子为 9
 radio.BWSel               = BW125K;        //扩频发射带宽 125KHz
 radio.CRSel               = CR4_5;         //CodeRate 为 4/5

 radio.vInitialize();                       //初始化
 radio.vGoStandby();                        //进入待机模式
}

void loop()
{
 radio.bSendMessage(str, 21);               //1 秒发射一次报文
 delay(1000);
}
```

➢ lora_Rx.ino 范例解释

```
#include <HopeDuino_LoRa.h>                 //调用对应库文件，因为本来用到 UART，所以增加 UART 调用
#include <HopeDuino_UART.h>

loraClass radio;                            //定义针对 LoRa 的类变量 radio
uartClass uart;                             //定义针对 UART 的类变量 uart
byte getstr[21];                            //定义待收数据缓冲

void setup()
{
 radio.Modulation          = LORA;          //调制模式为 LoRa
 radio.COB                 = RFM98;         //模块为 RFM98
 radio.Frequency           = 434000;        //目标频率 434MHz
 radio.OutputPower         = 17;            //17dBm  输出功率
 radio.PreambleLength = 16;                 //16Byte Preamble
 radio.FixedPktLength = false;              //可变长度报文格式
 radio.PayloadLength    = 21;               //报文长度 21Byte
 radio.CrcDisable          = false;         //不用 CRC
```

```
  radio.SFSel          = SF9;          //扩频因子为 9
  radio.BWSel          = BW125K;       //扩频发射带宽 125KHz
  radio.CRSel          = CR4_5;        //CodeRate 为 4/5

  radio.vInitialize();                 //初始化
  radio.vGoRx();                       //进入接收模式
  uart.vUartInit(9600, _8N1);          //执行 UART 配置初始化，初始化为 9600 波特率、8N1 格式
}

void loop()
{
  if(radio.bGetMessage(getstr)!=0)     //执行 radio 查询是否收到数据函数，收到就解析数据
      {
      uart.vUartPutNByte(getstr, 21);  //把收到的数据通过 UART 打印到 PC 上
      uart.vUartNewLine();             //UART 换行，便于阅读
      }
}
```

四、LoRa 库函数说明

“LoRa.h”和“LoRa.cpp”库文件存放路径在 Arduino IDE 文件夹\ libraries \ HopeRFLib；

- **FreqStruct**

  类型：Union 类型

  功能：针对 LoRa 芯片（模块）频率寄存器定义

  内容：Freq，长整型，4 字节，频率值；
  FreqL，字节，针对 Freq 拆分的频率值低 8 位[0:7]；
  FreqM，字节，针对 Freq 拆分的频率值中 8 位[8:15]；
  FreqH，字节，针对 Freq 拆分的频率值高 8 位[16:23]；
  FreqX，字节，冗余，凑整 4 字节，没有意义。

- **modulationType**

  类型：枚举类型

  功能：选择调制解调制式

  ⚠**内容：**OOK、FSK、GFSK、LORA
  OOK——On-Off-Key 属于 ASK 调制，是 ASK 调制一种特例；
  FSK——Frequency-Shift-Key，跳频键控 ，相对 ASK 有更强抗干扰效果；但同等功率情况下，比 ASK 电流大；
  GFSK——带高斯滤波的 FSK 调制；
  LoRa——Semtech 的特有扩频调制技术。

- **moduleType**

  类型：枚举类型

  功能：选择目标模块型号

  内容：RFM92, RFM93, RFM95, RFM96, RFM97, RFM98

- **sfType**
  类型：枚举类型
  功能：定义 LoRa 模式下，扩频因子（SF——spreading factor）
  内容：SF6, SF7, SF8, SF9, SF10, SF11, SF12

- **bwType**
  类型：枚举类型
  功能：定义 LoRa 模式下，扩频发射带宽（BW——band width）
  内容：BW62K, BW125K, BW250K, BW500K

- **crType**
  类型：枚举类型
  功能：定义 LoRa 模式下，扩频编码 CodeRate 选择
  内容：CR4_5, CR4_6, CR4_7, CR4_8

- **Modulation**
  类型：modulationType 类型
  功能：定义调制解调制式，在 OOK、FSK、GFSK、LoRa 四者中选择其一。

- **COB**
  类型：moduleType 类型
  功能：定义模块型号，COB 表示 Chip-On-Borad，可以在 RFM92, RFM93, RFM95, RFM96, RFM97, RFM98 中择其一。

- **Frequency**
  类型：lword 型（unsigned long）
  功能：目标工作频率，单位 KHz，例如：Frequency = 433920，代表 433.92MHz。

- **SymbolTime**
  类型：lword 型（unsigned long）
  功能：目标工作速率，单位 ns，例如：SymbolTime = 416000，代表每个符号 416us，即 2.4kbps。

- **Devation**
  类型：lword 型（unsigned long）
  功能：目标工作频偏，针对 FSK 和 GFSK 发射需要定义，单位 KHz，例如：Devation = 45，代表工作频率 45KHz。

- **BandWidth**
  类型：word 型（unsigned int）
  功能：目标工作接收带宽，针对接收需要定义，单位 KHz，例如：BandWidth = 100，代表接收带宽 100KHz。

- **OutputPower**
  类型：unsigned char
  功能：目标输出功率，针对发射需要定义，范围 2-20，单位 dBm，例如：设置为 10，表示 10dBm。

- **PreambleLength**
  类型：word 型（unsigned int）
  功能：数据包的 Preamble 长度设置，针对发射需要配置，单位字节。

- **CrcDisable**
  类型：bool 类型
  功能：选择数据包功能中是否带 CRC，设置 true 代表禁止 CRC 功能；设置 false 代表开启 CRC 功能。

- **FixedPktLength**
  类型：bool 类型
  功能：定义数据包是固定包长度，还是变长数据包，设置 true 代表固定包长度；设置 false 代表变长数据包格式。

- **SyncLength**
  类型：字节
  功能：无线数据包格式中，同步字长度，设置范围是 1～8 字节；不能设置为 0 字节。

- **SyncWord[8]**
  类型：字节数组
  功能：设置数据包格式中，同步字的内容，需要与 SyncLength 设置符合（长度）。

- **PayloadLength**
  类型：字节
  功能：在固定包长模式中，定义固定包长度。

- **SFSel**
  类型：sfType 类型
  功能：设置 LoRa 模式下的扩频因子，可以选择 SF6, SF7, SF8, SF9, SF10, SF11, SF12 其中之一。

- **BWSel**
  类型：bwType 类型
  功能：设置 LoRa 模式下的发射带宽参数，可以选择 BW62K, BW125K, BW250K, BW500K 其中之一。

- **CRSel**
  类型：crType 类型
  功能：设置 LoRa 模式下的 CodeRate，可以选择 CR4_5, CR4_6, CR4_7, CR4_8 其中之一。

- **vInitialize**
  类型：函数
  入参：无
  出参：无
  功能：初始化模块（芯片），适用于 RFM66 模块，在程序最开始调用；调用前，需要把上述关联的变量设置完整。初始化函数配置完毕后（内含调用 vConfig 函数），让模块（芯片）出于 Standby 状态，即非发、非收、非睡眠。

- **vConfig**

  类型：函数

  入参：无

  出参：无

  功能：配置参数到模块（芯片）中，适用于程序工作过程中需要重新配置参数的场合。调用前同样需要把关联变量设置完整。如果此前关联变量设置完整，后续无需改动，仅想重新配置一次参数，可以直接调用；如果需要在工作过程中，切换频点工作等，需要重新修改相关参数，然后在调用。调用完毕后，需要用到工作模式切换函数，以便让芯片准确到具体工作模式中，模式切换函数有：vGoRx、vGoStandby、vGoSleep 等。

- **vGoRx**

  类型：函数

  入参：无

  出参：无

  功能：配置模块（芯片）进入接收模式。

- **vGoStandby**

  类型：函数

  入参：无

  出参：无

  功能：配置模块（芯片）进入待机模式。

- **vGoSleep**

  类型：函数

  入参：无

  出参：无

  功能：配置模块（芯片）进入睡眠模式。

- **bSendMessage**

  类型：函数

  入参：**msg[ ]**，unsigned char 类型指针，待发射数据数组调用入口（指针）；

  length，unsigned char 类型，待发射数据长度，单位是 byte；

  出参：返回 bool 类型，true 表示发射成功；false 表示发射失败，诸如：推送超时等情况；

  功能：把待发射数据发送出去，仅发送一次（一帧）；发送完毕后自动返回待机状态（Standby 模式）。

- **bGetMessage**

  类型：函数

  入参：msg[ ]，unsigned char 类型指针，待接收数据数组调用入口（指针）；

  出参：返回接收数据的长度值，如果返回 0 表示没有收到数据；

  功能：查询是否接收到数据，查询对象是芯片输出的 IO 状态，如果没有收到数据，返回 0；如果收到数据，返回收到的数据长度，收取完毕后，模块（芯片）仍旧处于接收状态。

五、引脚分配表

HopeDuino	MCU	RF9x
13	PB5	SCK
12	PB4	MISO
11	PB3	MOSI
10	PB2	nCS
9	PB1	POR
8	PB0	DIO0
7	PD7	DIO1（跳线）
6	PD6	DIO2（跳线）
5	PD5	DIO3（跳线）
4	PD4	DIO4（跳线）

六、版本记录

版本	修改内容	日期
1.0	初始版本	2016-03-31
1.1	订正文字描述错误、增加水印、例程程序解释描述	2016-04-06