



华中农业大学

HUAZHONG AGRICULTURAL UNIVERSITY

物联网工程
实验报告

题目: WSN-LoRa 通信实验

姓名: 高星杰

学号: 2021307220712

学院: 信息学院

指导老师: 朱容波

中国 武汉

2024 年 5 月

一、实验目的

1. 熟悉 Keil4 开发环境的使用
2. 熟悉使用 STM32 芯片控制 LoRa 通讯过程。
3. 加深对无线传感器的概念的理解及应用，通过实践掌握物联网相关知识。
4. **实现：**使用 2 个 LoRa 节点，一个作为 Master(主机)，一个作为 Slave(从机)。实现上位机发送数据给 Master，Master 串口接收数据后通过 LoRa 无线发送给 Slave，Slave 无线接收到数据以后，通过 LoRa 无线把数据原样发送给 Master；Master 通过 LoRa 无线接收到数据后，通过串口把数据发送给上位机，以实现 Master 与 Slave 之间的数据首发。
5. **实现：**用 python 实现端口数据读取和显示 GUI 界面

二、实验设备

1. **硬件：**Lora 节点两个、串口线，J-Link 下载线，物联网实验开发箱；
2. **软件：**Keil4 For ARM 开发软件，串口调试工具；
3. **参考资料：**协议栈 API 接口说明文档，LoRa 源码，STM32 工程源码

三、实验原理

要想完整彻底的完成实验，那么我们必须要先了解本次实验的原理和理论，下面将使用问答的形式介绍本次实验的原理。

1. 什么是 LoRa 通讯？

LoRa 的英文全名是 Long Range，它获得了两个单词的头两个字母，从字面上可以理解为一种长距离通讯技术，它是在一个自由工作频段(即非授权频段)工作，包括 433MHz、470MHz、868MHz、915MHz 等等。采用时不需申请，我国实际解决方案中使用频率最多的频段主要是 410MHz-510MHz，根据 1MHz 的间隔计算，可支持 100 多个信道，实际可缩短信道间隔，扩大频段以增加信道数目。

LoRa 在实际应用中通常采用两种方式，可以将其分为组网模式和非组网模式，即以星形网为主体的网关节点通讯方式。LoRaWan 是其中之一。非组网模式主要是点对点，即由单个 lora 设备和一个 LoRa 设备所建立的通信方式。肯定也有不同的用法，前面提到的两种用法只有两种使用场景的方法。**本次实验使用的是主从模式，以非组网模式实现（不包含 LoRa 网关和服务器）**

LoRa 由于功耗低、传输距离长、组网灵活等许多特点，与物联网具有碎片化、低成本、大连接等特点，因此，LoRa 非常适合部署在物联网、智能家居、智能物联等领域，应用广泛。

2. LoRa 和 WSN 的关系是怎样的？

WSN(Wireless Sensor Network)无线传感器网络是一种由多个传感器节点组成的网络，这些节点通过无线通信协议（如 Zigbee、Wi-Fi、LoRa 等）相互连接，协作完成数据采集、传输和处理任务。

所以我们可以得出结论：**LoRa 可以作为 WSN 的一种通信技术，用于节点之间的数据传输。**LoRa 的长距离、低功耗特点非常适合大规模、分布广泛的传感器网络。

3. 使用 LoRa 的基本要求（LoRa - WSN 基本架构）

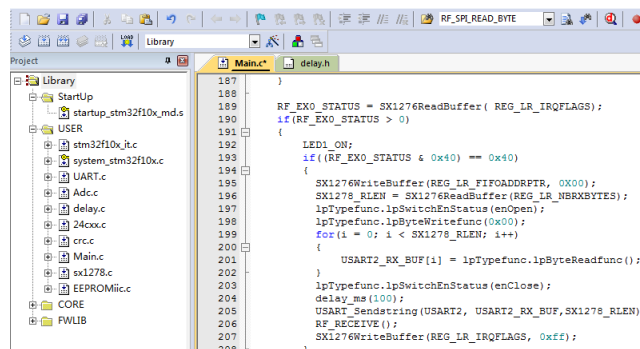
- ✧ **传感器节点**：传感器节点配备 LoRa 模块，负责数据采集和发送。节点可以是温度传感器、湿度传感器、气体传感器等。
- ✧ **LoRa 网关**：LoRa 网关负责接收传感器节点发送的数据，并通过互联网或局域网将数据转发到网络服务器。
- ✧ **网络服务器**：网络服务器处理和存储从 LoRa 网关接收的数据，进行数据分析和处理。
- ✧ **应用服务器**：应用服务器提供用户接口和数据应用，如可视化、报警、控制等功能。

而本次实验使用的并非这种架构，而是主从架构：

- ✧ **主设备 (Master)**：通常是 LoRa 网关，负责管理和控制整个网络的通信，收集从设备发送的数据，并将数据上传到上位机 (PC)。
- ✧ **从设备 (Slave)**：通常是分布在各处的传感器节点，负责采集环境数据并在接收到主设备的指令后，发送数据给主设备。

四、实验步骤

1. 打开 LoRa_STM32_Slave.ruvproj 工程文件。

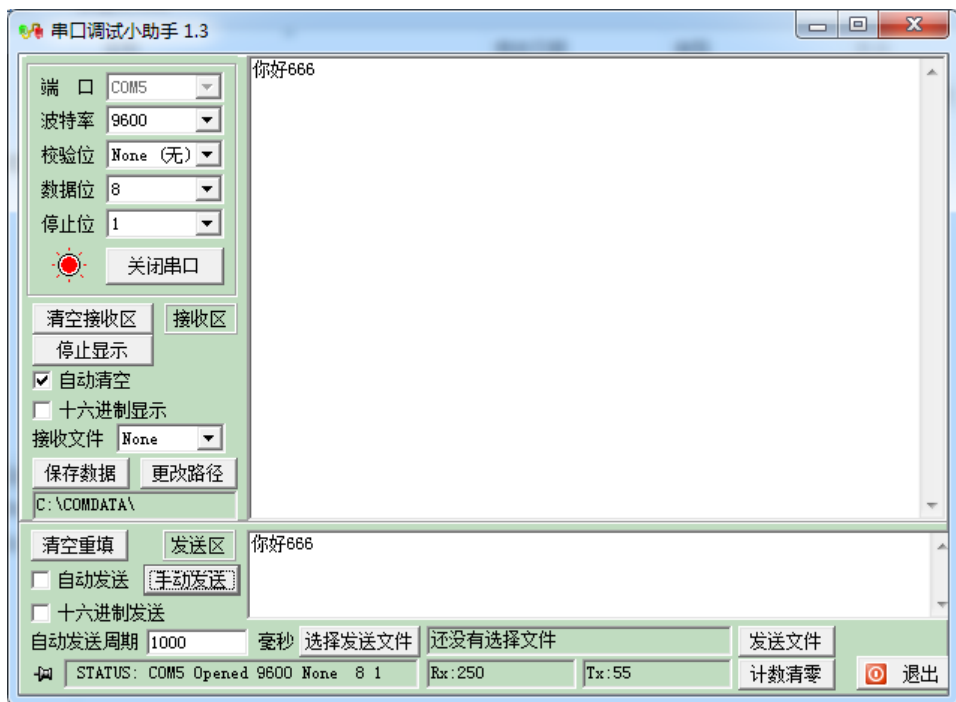


```
187 }
188
189 RF_EXO_STATUS = SX1276ReadBuffer( REG_LR_IRQFLAGS);
190 if(RF_EXO_STATUS > 0)
191 {
192     LED1_ON;
193     if((RF_EXO_STATUS & 0x40) == 0x40)
194     {
195         SX1276WriteBuffer(REG_LR_FIFOADDRPTR, 0x00);
196         SX1278_RLEN = SX1276ReadBuffer(REG_LR_NBRXBYTES);
197         lpTypefunc.lpSwitchEnStatus(enOpen);
198         lpTypefunc.lpByteWriteFunc(0x00);
199         for(i = 0; i < SX1278_RLEN; i++)
200         {
201             USART2_RX_BUF[i] = lpTypefunc.lpByteReadFunc();
202         }
203         lpTypefunc.lpSwitchEnStatus(enClose);
204         delay_ms(100);
205         USART_SendData(USART2, USART2_RX_BUF, SX1278_RLEN);
206         RF_RECEIVE();
207         SX1276WriteBuffer(REG_LR_IRQFLAGS, 0xff);
208     }
```


- 打开串口助手，选择电脑相应的串口号，串口参数：9600-8-N-1,不选 16 进制收发。



- 确定 Master 节点、Slave 节点这两个 LoRa 节点均上电。然后进行数据发送，在串口发送区发送“你好，666”。发送之后会在接收区得到“你好，666”，且在发送过程中主机和从机都会闪一下。

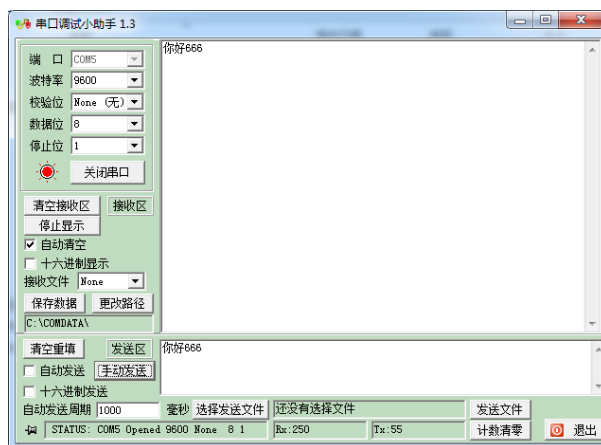


- 为证明是从机将数据返回，将从机芯片关闭，观看主机时候会闪，若不闪，则证明完成实验功能，实验正确。

五、实验结果

- 通过代码的编译、烧录、下载到 LoRa 传感器之后，实现了 Master 和 Slave 之间的数据收发功能，利用串口调试工具测试，上位机发送数据给 Master，Master 串口 2 接收到数据之后通过 LoRa 发送给 Slave，Slave

无线接收到数据之后通过 LoRa 无线把数据原样发送给 Master, Master 通过 LoRa 接收到数据以后, 通过串口 2 把数据发送给上位机。结果如图



2. 通过 Python 实现 GUI 界面, 从端口进行数据读取。



实验代码:

```
import tkinter
from tkinter import ttk

import serial
import serial.tools.list_ports

class SerialAchieve:
    def __init__(self,band=115200,check="无校验位",data=8,stop=1):
        self.port = None
        # 获取可用串口
        self.port_list = list(serial.tools.list_ports.comports())
        assert (len(self.port_list) != 0),"无可用串口"
```

```

self.bandRate = band
self.checkbit = check
self.databit = data
self.stopbit = stop

# 读写的数据
self.read_data = None
self.write_data = None

pass

def show_port(self):
    for i in range(0, len(self.port_list)):
        print(self.port_list[i])

def show_other(self):
    print("波特率: " + self.bandRate)
    print("校验位: " + self.checkbit)
    print("数据位: " + self.databit)
    print("停止位: " + self.stopbit)

# 返回串口
def get_port(self):
    return self.port_list

# 打开串口
def open_port(self, port):
    self.port = serial.Serial(port, self.bandRate, timeout = None)

def delete_port(self):
    if self.port != None:
        self.port.close()
        print("关闭串口完成")
    else:
        pass

def Read_data(self): # self.port.read(self.port.in_waiting) 表示全部接收串口中的数据
    self.read_data = self.port.read(self.port.in_waiting) # 读取数据
    return self.read_data.decode("utf-8")

def Write_data(self, data):
    if self.port.isOpen() == False:
        print("串口打开错误")
    else:
        self.port.write(data.encode("utf-8")) # 返回的是写入的字节数

```



```
class MainSerial:
    def __init__(self):

        # 定义串口变量
        self.port = None
        self.band = None
        self.check = None
        self.data = None
        self.stop = None
        self.myserial = None
        self.store = None

        # 初始化窗体
        self.mainwin = tkinter.Tk()
        self.mainwin.title("串口调试工具")
        self.mainwin.geometry("600x400")

        # 标签
        self.label1 = tkinter.Label(self.mainwin, text = "串口号:", font = ("宋体", 15), fg = 'red', bg = 'blue')
        self.label1.place(x = 5, y = 5)
        self.label2 = tkinter.Label(self.mainwin, text="波特率:", font=("宋体", 15), fg = 'red', bg = 'blue')
        # self.Label2.place(x=5, y=45)
        self.label3 = tkinter.Label(self.mainwin, text="校验位:", font=("宋体", 15), fg = 'red', bg = 'blue')
        # self.Label3.place(x=5, y=85)
        self.label4 = tkinter.Label(self.mainwin, text="数据位:", font=("宋体", 15), fg = 'red', bg = 'blue')
        # self.Label4.place(x=5, y=125)
        self.label5 = tkinter.Label(self.mainwin, text = "停止位:", font = ("宋体", 15), fg = 'red', bg = 'blue')
        # self.Label5.place(x = 5, y = 165)

        # 文本显示, 清除发送数据
        self.label6 = tkinter.Label(self.mainwin, text="发送数据:", font=("宋体", 15), fg = 'blue')
        self.label6.place(x=230, y=5)

        self.label7 = tkinter.Label(self.mainwin, text="接收数据:", font=("宋体", 15), fg = 'blue')
        self.label7.place(x=230, y=200)
```

```

# 串口号
self.com1value = tkinter.StringVar() # 窗体中自带的文本, 创建一个值
self.combobox_port = ttk.Combobox(self.mainwin, textvariable=self.com1value,
                                   width = 10, font = ("宋体", 13))

# 输入选定内容
self.combobox_port["value"] = ["COM1"] # 这里先选定

self.combobox_port.place(x = 105, y = 5) # 显示

# 波特率
self.bandvalue = tkinter.StringVar() # 窗体中自带的文本, 创建一个值
self.combobox_band = ttk.Combobox(self.mainwin, textvariable=self.bandvalue, width=10, font=("宋体", 13))

# 输入选定内容
self.combobox_band["value"] = ["4800", "9600", "14400", "19200", "38400", "57600", "115200"] # 这里先选定

self.combobox_band.current(1) # 默认选中第0个
# self.combobox_band.place(x=105, y=45) # 显示

# 校验位
self.checkvalue = tkinter.StringVar() # 窗体中自带的文本, 创建一个值
self.combobox_check = ttk.Combobox(self.mainwin, textvariable=self.checkvalue, width=10, font=("宋体", 13))

# 输入选定内容
self.combobox_check["value"] = ["无校验位"] # 这里先选定
self.combobox_check.current(0) # 默认选中第0个
# self.combobox_check.place(x=105, y=85) # 显示

# 数据位
self.datavalue = tkinter.StringVar() # 窗体中自带的文本, 创建一个值
self.combobox_data = ttk.Combobox(self.mainwin, textvariable=self.datavalue, width=10, font=("宋体", 13))

# 输入选定内容
self.combobox_data["value"] = ["8", "9", "0"] # 这里先选定
self.combobox_data.current(0) # 默认选中第0个
# self.combobox_data.place(x=105, y=125) # 显示

# 停止位
self.stopvalue = tkinter.StringVar() # 窗体中自带的文本, 创建一个值
self.combobox_stop = ttk.Combobox(self.mainwin, textvariable=self.stopvalue, width=10, font=("宋体", 13))

# 输入选定内容
self.combobox_stop["value"] = ["1", "0"] # 这里先选定
self.combobox_stop.current(0) # 默认选中第0个

```

```

# self.combobox_stop.place(x=105, y=165) # 显示

# 按键显示, 打开串口
self.button_OK = tkinter.Button(self.mainwin, text="打开串口",
                                command=self.button_OK_click, font = ("宋体",13),
                                width = 10,height = 1)
self.button_OK.place(x = 5,y = 210) # 显示控件

# 关闭串口
self.button_Cancel = tkinter.Button(self.mainwin, text="关闭串口", # 显示文本
                                    command=self.button_Cancel_click, font = ("宋体",13),
                                    width=10, height=1)
self.button_Cancel.place(x = 120,y = 210) # 显示控件

# 清除发送数据
self.button_Cancel = tkinter.Button(self.mainwin, text="清除发送数据", # 显示文本
                                    command=self.button_clcSend_click, font=("宋体", 13),
                                    width=13, height=1)
self.button_Cancel.place(x=400, y=2) # 显示控件

# 清除接收数据
self.button_Cancel = tkinter.Button(self.mainwin, text="清除接收数据", # 显示文本
                                    command=self.button_clcRece_click, font=("宋体", 13),
                                    width=13, height=1)
self.button_Cancel.place(x=400, y=197) # 显示控件

# 发送按键
self.button_Send = tkinter.Button(self.mainwin, text="发送", # 显示文本
                                   command=self.button_Send_click, font=("宋体", 13),
                                   width=6, height=1)
self.button_Send.place(x=5, y=255) # 显示控件

# 接收按键
self.button_Send = tkinter.Button(self.mainwin, text="接收", # 显示文本
                                   command=self.button_Rece_click, font=("宋体", 13),
                                   width=6, height=1)
self.button_Send.place(x=5, y=310) # 显示控件

```

```

# 显示框
# 实现记事本的功能组件
self.SendDataView = tkinter.Text(self.mainwin,width = 40,height = 9,
                                  font = ("宋体",13)) # text 实际上是一个文本编辑
器
self.SendDataView.place(x = 230,y = 35) # 显示

self.ReceDataView = tkinter.Text(self.mainwin, width=40, height=9,
                                  font=("宋体", 13)) # text 实际上是一个文本编辑器
self.ReceDataView.place(x=230, y=230) # 显示

# 发送的内容
test_str = tkinter.StringVar(value="物联网工程")
self.entrySend = tkinter.Entry(self.mainwin, width=13,textvariable = test_str,font = ("宋体",15))
self.entrySend.place(x = 80,y = 260) # 显示

# 获取文件路径
test_str = tkinter.StringVar(value="物联网工程")
self.entrySend = tkinter.Entry(self.mainwin, width=13, textvariable=test_str, font=("宋体", 15))
self.entrySend.place(x=80, y=260) # 显示

# 获取界面的参数
self.band = self.combobox_band.get()
self.check = self.combobox_check.get()
self.data = self.combobox_data.get()
self.stop = self.combobox_stop.get()
print("波特率: "+self.band)
self.myserial = SerialAchieve(int(self.band),self.check,self.data,self.stop)

# 处理串口值
self.port_list = self.myserial.get_port()
port_str_list = [] # 用来存储切割好的串口号
for i in range(len(self.port_list)):
    # 将串口号切割出来
    lines = str(self.port_list[i])
    str_list = lines.split(" ")
    port_str_list.append(str_list[0])

self.combobox_port["value"] = port_str_list
self.combobox_port.current(0) # 默认选中第0个

```

```

def show(self):
    self.mainwin.mainloop()

def button_OK_click(self):
    ...

    @ 串口打开函数
    :return:
    ...

    if self.port == None or self.port.isOpen() == False:
        self.myserial.open_port(self.combobox_port.get())
        print("打开串口成功")
    else:
        pass

def button_Cancel_click(self):
    self.myserial.delete_port()
    print("关闭串口成功")

def button_clcSend_click(self):
    self.SendDataView.delete("1.0", "end")

def button_clcRece_click(self):
    self.ReceDataView.delete("1.0", "end")

def button_Send_click(self):
    send_str1 = "hello"
    self.myserial.Write_data(send_str1)
    try:
        if self.myserial.port.isOpen() == True:
            print("开始发送数据")
            send_str1 = self.entrySend.get()
            self.myserial.Write_data(send_str1)
            self.SendDataView.insert(tkinter.INSERT, send_str1+" ")
            self.store = send_str1
            print("发送数据成功")
        else:
            print("串口没有打开")
    except:
        pass

def button_Rece_click(self):
    try:

        readstr = self.myserial.Read_data()
        self.ReceDataView.insert(tkinter.INSERT, self.store+ ' ' + readstr + " ")

```

```
        self.store = None
    except:
        pass
if __name__ == '__main__':
    my_ser1 = MainSerial()
    my_ser1.show()
```

六、实验总结与感悟

这次实验如同一场激动人心的探索之旅，让我对 LoRa 通信和无线传感器网络（WSN）的奥秘有了更深的理解，同时也让我体会到了理论知识在实际应用中的无穷魅力。实验的过程既是对知识的检验，更是对自我能力的一次次挑战。

在实验中，我学会了如何在 Keil4 开发环境中编写代码，并通过 J-LINK 工具将其下载到硬件设备中。看着代码在 STM32 芯片上运行，主从节点通过 LoRa 无线通信实现数据的传输，我心中充满了成就感。尤其是当看到上位机发送的数据成功通过 LoRa 链路回传至主机，整个过程如同魔法般令人惊叹。通过 Python 实现的串口通信和 GUI 界面显示，更是让整个实验变得生动具体，仿佛将抽象的理论世界具象化在眼前。

实验中遇到的问题仿佛一道道关卡，但每次攻克这些难关时的喜悦，更坚定了我对物联网技术的热爱。数据乱码和通信故障曾让我一度困惑，但通过调整频率和修改代码，这些问题最终迎刃而解。每一次的失败和成功都让我对 LoRa 通信的实现原理有了更深刻的理解，增强了我的问题解决能力。

更为重要的是，这次实验让我深刻体会到了团队协作的力量。与组员们一起讨论、解决问题的过程，是我最珍贵的收获之一。每一次的交流和合作，不仅让我们更快地找到解决方案，也让我明白了团队合作的重要性和美妙之处。那种共同攻克难关后的喜悦，远比单打独斗来的更加绚丽。

动手实践能力的提高是这次实验的另一大收获。通过亲自操作和调试，我对 LoRa 通信的每一个细节都了然于心。这不仅锻炼了我的实际操作能力，也让我对物联网技术有了更直观的认识和体验。每一次成功的数据传输，每一个顺利运行的程序，都像是在探索未知领域时发现的新大陆，令人心潮澎湃。

这次实验不仅巩固了我对课堂知识的掌握，还拓展了我的知识面。理论与实践的完美结合，使我对物联网的未来充满了无限憧憬。每一个细节，每一个步骤，都在我心中留下了深刻的印记，激发了我不断探索和学习的热情。

所以总的来说，这次实验是一次充满挑战和收获的旅程。它不仅让我在知识和技能上得到了提升，更让我在心灵上感受到了探索和发现的快乐。我深知，这只是物联网世界的一小步，但对于我个人成长而言，却是一次重要的飞跃。我期待着在未来的学习和研究中，继续探索物联网技术的广阔天地，掌握更多的知识和技能，为实现自己的梦想而不懈努力。