

Anylinkin® ALK8266WIFI®模组

SPI 接口高速通信使用与集成

常见问题 FAQ

目 录

1 前言	4
1.1 简介	4
1.2 术语约定	4
1.3 通过书签方便阅读和快速检索定位	5
1.4 参考文档	5
1.5 联系我们	5
2 与 ALK8266WIFI®模组的基础使用有关的常见问题	6
2.1 怎么判断 ALK8266WIFI®模组是否正常地启动起来了?	6
2.2 ALK8266WIFI®模组复位的正常时序是怎样的?	6
2.3 ALK8266WIFI®模组上的两个 LED 灯各代表什么意思?	6
3 与 ALK8266WIFI®模组的集成开发使用有关的常见问题	7
3.1 与 ALK8266WIFI®模组的通信初始化过程是怎样的?	7
3.2 单片机主机为何要先将 nCS 片选拉低再对模组复位?	8
3.3 如何验证主机固件里 SPI 接口初始化和基本字节读写逻辑是正确的? ...	9
3.4 如何测试验证或获得在可靠通信时主机 SPI 走线能达到最大频率? ...	9
3.5 SPI 接口片选信号为何要用 GPIO 软件控制?	9
3.6 LED1 (绿灯) 闪烁或灭掉代表什么意思?	9
3.7 模组连接路由器时, 有哪些方法可以查看模组的 IP 地址?	10
4 与网络通信基础有关的常见问题	12
4.1 与 AP/STA、TCP/UDP 基本概念有关的基础问题	12
4.1.1 AP/STA 服务器/客户端有何区别? AP 可做 TCP 客户端吗?	12
4.1.2 如何缩短 STA 连接 AP 或路由器的时间?	13
4.1.3 如何用两个 ALK8266WIFI®模组组成一个对等的 UDP 通信?	15
4.1.4 为何使用 TCP 通信往往比 UDP 要慢?	15
4.1.5 路由器对 TCP 通信速度为何往往会有较大的 (负面) 影响?	16
4.1.6 网络通信时为何建议一个 TCP 包的大小不要超过 1460 或类似的某个数值?	16
4.1.6 建立 TCP 客户端服务, 为何建议随机产生本地端口而非指定? ..	17
4.1.7 UDP 广播时, 速度为什么很慢不超过 100KBytes/s?	17
4.1.8 地址冲突是什么意思? 地址冲突时为何 TCP 客户端会连不上 TCP 服务器?	17
4.1.9 发长包的有效速度为何比发短包快?	18
4.2 与天线相关的基础常识	20
4.2.1 什么是天线的增益? 为何增益越大, 对方向的对准要求越高? ..	20
4.2.2 为何高增益的天线有可能提高传输速度?	20
4.3 WIFI 的传输距离与那些因素有关? 模组的传输距离有多远?	22
4.4 如何减少射频的同频干扰, 例如产品上同时有 2.4G 蓝牙和 WIFI 模组?	23
5 与高速通信应用有关的常见问题	25
5.1 单片机通过 WIFI 模组上传文件, 对方是一个 FTP 服务器, 需要在单片机	

上移植一个完整的 FTP 客户端吗?	25
5.2 音视频的录-传-播同步, 需要注意什么?	25
5.3 采用例程测试速度有 1MBytes/s, 但加上高速采集后系统速度降低一半, 这是为什么? 该如何优化?	26
5.4 怎么通过这个模块获取互联网时间?	26
5.5 模组通过路由器联网做 TCP 服务器, 不知其 IP 地址怎么办?	28
5.6 如何让 WIFI 模块通过网页实名认证的网关上网?	29
6 与单片机基础性开发调试有关的常见问题	31
6.1 为何跑飞进入 HardFault_Handler 死循环?	31
7 如有不清之处, 可随时通过 QQ 或微信等手段和我们进行交流	32

1 前言

1.1 简介

Anylinkin® ALK8266WIFI® 是一款灵活、功能强大、高性能、精简小尺寸、绿色环保、在高性能前提下的高性价比的 802.11 b/g/n 无线模组。它包含有 (1) 高性能且高度集成的无线片上系统芯片, 提供智能高效的无线接入; (2) 标准 2.0mm 间距的排针全孔半孔 (复合邮票孔), 提供高速通信 SPI 从机接口; (3) 串口数据排针半孔 (邮票孔) 接口, 提供 UART 串行通信接口; (4) 同时, 还带有一些 IO 外设接口和 LED 灯, 可用于用户扩展。

通过 ALK8266WIFI® 模组所提供的 SPI 半孔整孔复合(邮票孔)接口, MCU 系统 (1) 可以实现和远端 TCP/UDP 服务节点实现高速通信, 最大波特率可达 40Mbps, 实测有效吞吐量可以超过 M 字节每秒, 适用于高速采集、语音、图片以及视频传输等场合; (2) 通过我们提供的基本的 SPI 控制协议, 直接通过 SPI 总线接口就可以对模组及其片上资源进行设置、查询和控制, 无需 UART 串口介入, 以便节约主机的串口线用作其他功能。

本文是在使用 ALK8266WIFI®模组进行单片机 WIFI 通信时的一些常见疑问和回答, 供参考。所牵涉范围不限于模组本身, 也包含许多单片机 WIFI 网络通信上的一些基础实用知识和技巧, 作为《ALK8266WIFI 模组 SPI 接口高速通信使用与集成—主机集成说明》辅助参考资料。随着对越来越多用户的支持和探讨的继续, 本文中的常见问题也会不断添加和更新。

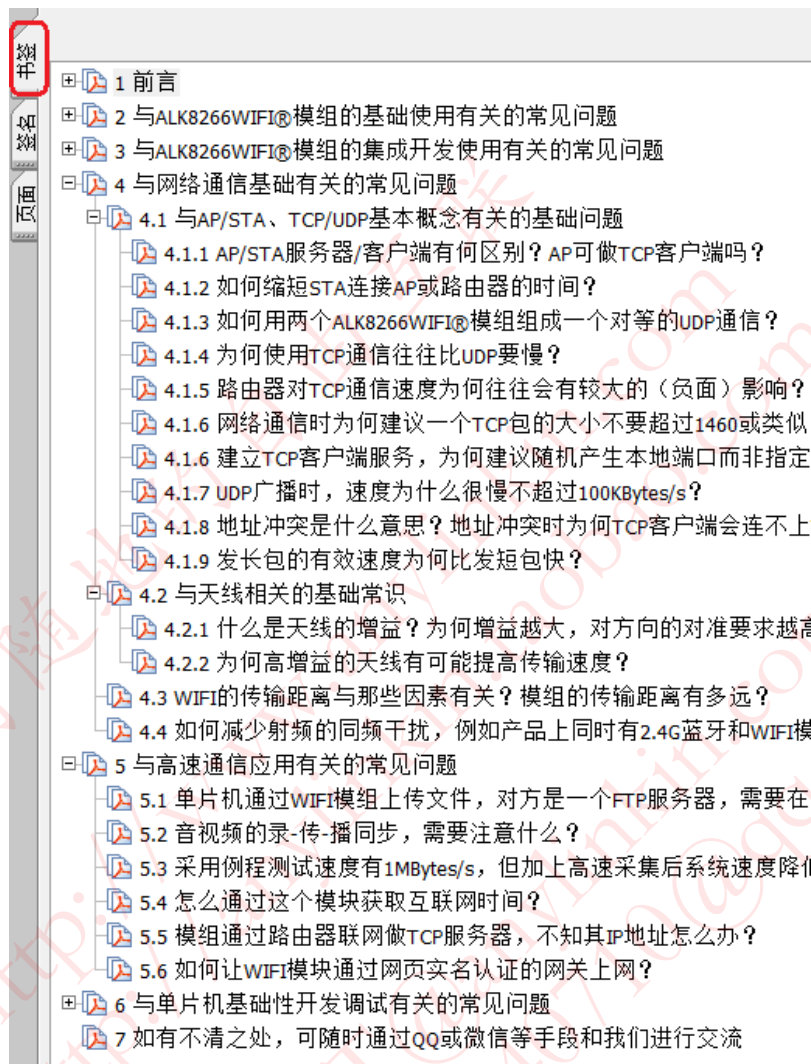
文中如有疏漏或认识不足之处, 欢迎大家积极指出、建议或批评斧正。谢谢!

1.2 术语约定

术语	解释
连接	一个广义的概念, 包括硬件电路上两个网表节点之间的连接、WIFI 工作站 (STA) 连接热点 AP 或路由器、网络层的 TCP 或 UDP 连接。在本文中, 为了避免混淆, 尽量使用接入和链接, 来区分后两种“连接”。如非特别声明, “连接”一般指硬件上的“逻辑互联”。
逻辑互联	特指硬件电路的两个网表节点之间, 用 PCB 布线、导线飞线等方式连接起来。
接入	特指 WIFI 工作站(STA)使用 SSID 和密码来连接热点 AP 或路由器, 对应英文单词 ACCESS。
链接	特指网络 TCP/IP 层的 TCP 或 UDP 套接字链路连接, 对应于模组上建立的一个服务 (Service)。
服务	一个网络节点开启的功能, 使用 IP 地址和端口标识, 通过套接字将两个节点的服务关联起来, 实现通信, 对应英文单词 Service。例如我们常说的 TCP 服务、UDP 服务等等。

1.3 通过书签方便阅读和快速检索定位

本文档包含相关书签, 请以方便查阅和快速检索, 如下图所示



1.4 参考文档

- 1、ALK8266WIFI 模组数据手册
- 2、ALK8266WIFI 模组 SPI 接口高速通信使用与集成—主机集成说明
- 3、ALK8266WIFI 模组 SPI 接口高速通信使用与集成—主机驱动 API 函数
- 4、常见的 WIFI 模组配网方式简介与对比暨 ANYLINKIN 8266WIFI 模组配网方式和操作说明

1.5 联系我们

网址: <http://www.anylinkin.com>

淘宝: <http://anylinkin.taobao.com>

电邮: IoT@anylinkin.com, 1521340710@qq.com

如有技术咨询、探讨、或疑惑, 欢迎和我们联系, 提出您的宝贵意见或建议。谢谢支持!

2 与 ALK8266WIFI®模组的基础使用有关的常见问题

2.1 怎么判断 ALK8266WIFI®模组是否正常地启动起来了?

Q: 怎么判断 ALK8266WIFI® 模组是否正常地启动起来了?

A: 要 ALK8266WIFI® 模组能正常使用, 需要模组正常启动。正常启动的末期, 模组上的两个灯会走马交替闪烁几次。

如果看到了这两个灯交替闪烁, 则表明模组正常地启动起来了, 参看《[ALK8266WIFI 模组 SPI 接口高速通信使用与集成—主机集成说明](#)》中“4.12.1 启动时两个 LED 灯先交替闪烁”章节的相关说明。

此外, 串口也会输出模组的启动信息, 并最后看到“ready”输出, 这也可以用来判断模组是否正常地启动了。

2.2 ALK8266WIFI®模组复位的正常时序是怎样的?

Q: 为什么我将模组接上单片机但还没有跑程序, 通电后模组依然可能不会启动?

A: 和其他模组比较, ALK8266WIFI® 模组几乎不会遇到启动不起来的情形。但是如果使用方法不对, 也可能会遇到不能正常启动的问题。

当使用 SPI 接口时, 需要确保模组能按照正常的复位时序进行启动, 需要确保在上电复位或 nRST 管脚复位阶段, 片选信号管脚处于悬空或拉低状态, 不得处于高电平状态。

可参看《[ALK8266WIFI 模组 SPI 接口高速通信使用与集成—主机集成说明](#)》中“4.2 模组的硬复位”章节中的相关说明以及本文 [3.2 主机固件程序里为何要先将 nCS 片选拉低再对模组复位?](#) 获取进一步的详细说明。

2.3 ALK8266WIFI®模组上的两个 LED 灯各代表什么意思?

Q: ALK8266WIFI® 模组上的两个 LED 灯各代表什么意思?

A: 参看《[ALK8266WIFI 模组 SPI 接口高速通信使用与集成—主机集成说明](#)》中“4.12 模组板载 LED 灯的诊断显示”章节中的说明。

3 与 ALK8266WIFI®模组的集成开发使用有关的常见问题

3.1 与 ALK8266WIFI®模组的通信初始化过程是怎样的?

Q: 对 ALK8266WIFI® 模组的通信初始化过程是怎样的?

A: 步骤如下:

ALK8266WIFI® 模组的通信初始化全部位于 M8266WIFI_ops.c 中的函数:

ALK8266WIFI_Module_Init_Via_SPI()

该函数的核心操作步骤描述如下:

1) 复位模组 -- M8266WIFI_Module_Hardware_Reset()

目的: 复位模组的目的是确保模组有个正确的起点, 包括确保能正确地加载程序等

要求: 必须项

2) 指定模组所使用的 SPI 接口参数 --M8266HostIf_SPI_Select()

目的: 向驱动传输主机所使用的 SPI 和频率

要求: 必须项

3) 配网与连网 – SPI API 函数配网、AT 指令配网、WEB 网页配网、自动配网

目的: 让模块接入路由器

要求: 有条件必须项

如果使用模组的 AP 模式

其他节点来连接这个模组热点所建立的局域网, 那么只需要通过 SPI 接口 API 函数、串口 AT 指令、或 WEB 网页, 来设置该模组的热点 SSID 和密码, 以便其他节点可以接入这个模组 AP 组成的局域网。一旦配置成 AP 模式, 本模块就无须配网。

如果使用模组的 STA 模式

如果以前成功地连接过期望的热点, 且在配网时选择了“保存”, 那么上电复位或掉线后, ALK8266WIFI® 模组会自动连接热点, 所以, 不必重新配网, 这一步可以省略。

否则, 可以通过 SPI 接口 API 函数、串口 AT 指令、或 WEB 网页进行手动配网, 或者自动配网方式 (参看我司提供的相关文档《[常见的 WIFI 模组配网方式简介与对比暨 ANYLINKIN 8266WIFI 模组配网方式和操作说明](#)》)。

4) 确保连网成功 -- M8266WIFI_SPI_wait_sta_connecting_to_ap_and_get_ip()

目的: 确保模块联网成功, 以便后续建立 TCP 或 UDP 链接

要求: 有条件必须项

如果使用模组的 AP 模式

不需要本步, 因为工作在 AP 模式下的模组, 使用自身提供的 IP 地址, 一般缺省为 192.168.4.1。

如果使用模组的 STA 模式

需要本步, 以便在后续建立 TCP 或 UDP 链接的之前, 模块已经成功地获取到了 IP 地址。

5) 建立链接 -- M8266WIFI_SPI_Setup_Connection()

完成了上述初始化后, 就可以开始建立服务或与远端节点 TCP/UDP 链接:

M8266WIFI_SPI_Setup_Connection()

然后就可以开始利用套接字链接开始高速收发数据了

对于 TCP 方式的连接, 目前 SPI 高速通信, 支持 ALK8266WIFI® 模组作为 TCP 客户端进行连接远端 TCP 服务器, 也支持 ALK8266WIFI® 模组作为 TCP 服务器。

对于 UDP 方式的连接, 因为是对等连接, 所以无所谓服务器和客户端, 两个节点之间的 UDP 服务可以任意地互联通信。UDP 模式下, 支持单播、组播, 以及广播。

3.2 单片机主机为何要先将 nCS 片选拉低再对模组复位?

Q: 在对 ALK8266WIFI® 模组复位时, 单片机主机为什么需要先将 nCS 片选拉低再复位?

A: ALK8266WIFI® 模组主芯片有多种启动模式, 通过启动时芯片的几个 IO 管脚上的上下拉电压进行选择, 这几个 IO 管脚也称之为启动模式选择管脚。要让 ALK8266WIFI® 模组能正常启动和正常工作, 需要确保模组在启动期间, nCS (即 IO15) 管脚处于低电平状态, 而 IO0 和 IO2 处于高电平状态。

在 ALK8266WIFI® 模组上, nCS (即 IO15) 管脚已通过一个 10Kohm 的电阻下拉, IO0 通过 100Kohm 和一个 LED 上拉, 而 IO2 内部上拉。所以, 如果只接电源 VCC 和地 GND 管脚, 上电后, 模组就会正常启动和正常工作。

但是, 大多数的主机主板在进行设计时, 会将 nCS 信号接一个电阻上拉, 或者启动时主板上可能会存在驱动竞争 (即在复位期间, nCS 可能会因为悬空或单片机的随机输出而被意外地驱动为高电平), 从而导致模组启动失败 (而到其他非正常工作的启动模式) 了。

因此, 为了确保模组能正常启动开始正常工作, 在对模组复位之前, 单片机固件里会有意先输出低电平来确保模组的 nCS 管脚处于低电平状态。

3.3 如何验证主机固件里 SPI 接口初始化和基本字节读写逻辑是正确的?

Q: 如何验证主机固件里的 SPI 接口初始化是正确的?

A: 单片机完成主机初始化并通过 M8266WIFI_Module_Hardware_Reset()对模组完成复位时序后, 第一步需要调用的函数为 M8266HostIf_SPI_Select()。若该函数返回成功, 则表明单片机主机对 SPI 接口的初始化已经成功, 和 WIFI 模组之间的基本字节读写逻辑都已经正确。

3.4 如何测试验证或获得在可靠通信时主机 SPI 走线所能达到最大频率?

Q: 如何测试验证或获得主机的 SPI 接口在可靠通信时, SPI 走线所能达到的最大频率?

A: 参看《[ALK8266WIFI 模组 SPI 接口高速通信使用与集成—主机集成说明](#)》中的“5.4.5 验证/获取主机板 SPI 接口稳定可靠通信的最大时钟频率”章节。

3.5 SPI 接口片选信号为何要用 GPIO 软件控制?

Q: SPI 接口的片选信号为何要用 GPIO 软件控制? 主机 SPI 片选是封装一体的怎么办?

A: 使用主机的 GPIO 来输出片选 (即软件方式控制片选), 是因为在一个片选有效期间, 可以支持多字节连续读和写 SPI 数据。如果采用主机的硬件接口自动配置片选信号输出, 就不够灵活了。所以, 绝大多数单片机主机, 都支持由 GPIO 来软件控制 SPI 片选。

当然, 也可能有极个别非常特殊的单片机主机, 它的片选信号是封装一体的, 或者因为平台对外公开的库做了封装, 导致主机在读写 SPI 时, 只能是读写一个字节, 同时自动就一个片选周期, 很不灵活 (我们一个客户使用第三方平台做二次开发就遇到了这种情形)。

这个时候, 可以采用一种技巧, 同样可以实现片选信号的 GPIO 软件控制, 即, 将主机或库封装一体所对应的片选信号废弃不用 (悬空), 不要去接上模组的片选, 重新挑选一个主机 GPIO 作为片选, 并按照《[ALK8266WIFI 模组 SPI 接口高速通信使用与集成—主机驱动 API 函数](#)》中的章节“3.1 M8266HostIf_GPIO_CS_RESET_Init”和章节“3.5 M8266HostIf_Set_SPI_nCS_Pin”实现相关初始化个高低电平输出的实现。这样, 在该单片机上执行 SPI 读写或者调用封装一体的库函数时, 尽管该单片机或封装库, 会在原来封死指定的管脚上输出 SPI 片选, 但是由于该管脚被悬空没有接上模组, 所以, 对模组没有影响, 而真正的模组 SPI 接口的片选, 则由另外指定的这个 GPIO 给出。

3.6 LED1 (绿灯) 闪烁或灭掉代表什么意思?

Q: 接收数据采用查询方式, 单片机大循环固件功能复杂增多膨胀时, 会看到 LED1 等开始闪烁或灭掉, 这代表什么意思?

A: 表明 ALK8266WIFI® 模组接收到了数据, 但是单片机主机没有或者没有及时地读取走数据。此时, 一般需要优化单片机固件大循环中执行时间太长的环节, 避免查询阻塞, 或者改用中断接收方式。参看《[ALK8266WIFI 模组 SPI 接口高速通信使用与集成—主机集成说明](#)》中的“4.10.2 利用数据中断 IO 管脚来辅助诊断和优化单片机主程序的技巧”。

3.7 模组连接路由器时, 有哪些方法可以查看模组的 IP 地址?

Q: 模组连接路由器时, 有哪些方法可以查看模组的 IP 地址?

A: ALK8266WIFI® 模组的功能非常全面和灵活, 有多种方法可以查看模组从路由器上获取到的 IP 地址。

(1) SPI API 函数查询, 如果使用 SPI 接口和模组交互

u8 M8266WIFI_SPI_Get_STA_IP_Addr(char* sta_ip , u16* status);

```
/* *****  
 * M8266WIFI_SPI_Get_STA_IP_Addr  
 * .Description:  
 *   To get ip address of M8266WIFI STA via SPI  
 * .Parameter(s)  
 *   1. sta_ip : the sta ip address returned if successful  
 *               "0.0.0.0" returned if in AP-only mode or ip not achieved  
 *   2. status : pointer to return errcode(LSB) and status(MSB) upon error  
 *               Use NULL if you don't expect them returned  
 * .Return value:  
 *   =1, success  
 *   =0, has error(s)  
 * ***** */  
u8 M8266WIFI_SPI_Get_STA_IP_Addr(char* sta_ip , u16* status);
```

(2) AT 指令查询, 如果使用 UART 串口和模组交互

AT+CIFSR\r\n

(3) WEB 网页查询

如果模组的工作模式包含有 AP 模式, 例如 STA+AP 模式, 那么可以通过打开模组上的 WEB 网页, 来查看模组当前所连接的路由器信息, 包括获取到的 IP 地址。

WIFI连接信息

连接状态	CONNECTED
当前SSID	aa7
信道	9
IP地址	192.168.43.89
MAC地址	
信号强度	-37dBm

刷新

连接路由器

SSID	aa7
密码	

连接成功!

(4) 路由器直接查看

我们的每个模组都支持路由器测显示对应的 HostName, 每个 HostName 都有 Anylinkin! 的特殊标志, 并且和其序列号存在一一对应关系, 所以便于和路由器上

其他接入的节点区分。

客户端列表

本页面用于查看当前DHCP服务器的客户端信息。

ID	客户端名	MAC地址	IP地址	有效时间
1				22:22:01
2				19:28:50
3				16:54:12
4				22:09:48
5	Anylinkin! WIFI-001			23:59:28

刷新

这里的显示数值可以通过M8266WIFI_SPI_Set_STA_Hostname()等API函数修改

4 与网络通信基础有关的常见问题

4.1 与 AP/STA、TCP/UDP 基本概念有关的基础问题

4.1.1 AP/STA 服务器/客户端有何区别? AP 可做 TCP 客户端吗?

Q: AP/STA TCP/UDP 服务器和客户端有什么区别? 热点 AP 可以做 TCP 客户端吗?

A: AP (Access Point, 热点) 和 STA (Station, 基站) 是与基础网络的建立(组网联网)有关的一个层次上的概念; TCP 和 UDP 是另外一个层次上与数据包的传输控制有关的同层次概念, 再套接字通信上, 服务器和客户端是建立了 TCP 服务之后与 TCP 有关的一个同层次概念。进一步的理解这里概念的差别, 可通过 www.bing.com 相搜索关键字熟悉。

如果要建立 WIFI 局域网, 必须有一个热点 AP (或包含接入功能的路由器), 其他的节点作为 STA 接入这个热点所组成的局域网络。AP 和 STA 都是这个局域网内的节点, 都有唯一的 IP 地址。

TCP 和 UDP 是网络数据传输的控制方式/协议。TCP/UDP 需要先建立相应的服务 SERVICE, 一个服务一般包括 IP 地址和端口。任意的局域网节点, 都可以建立起 TCP 或 UDP 服务, 使用唯一的 (IP 地址, 端口号) 来标识。

SERVICE(服务)有服务器端 SERVER 和客户端 CLIENT 的概念。但并非所有的 SERVICE 都有 SERVER (服务器端) 和 CLIENT (客户端) 的概念, 例如 UDP 就没有 SERVER (服务器端) 和 CLIENT (客户端) 的概念之分。

TCP 服务必须有一个 TCP SERVER (服务器端), 其他的都是 TCP CLIENT (客户端)。每个 TCP 客户端只能和 TCP SERVER (服务器端) 建立连接和通信, 且 TCP CLIENT (客户端) 在连接 TCP SERVER (服务器) 时, TCP SERVER (服务器) 必须已经建立起来了服务器服务 (SERVER SERVICE), 监控客户端的连接。所以 TCP 连接一般属于非对等连接。TCP 通信底层包含有握手和重发机制。

而 UDP 连接则是对等的, UDP 本身没有 SERVER (服务器端) 和 CLIENT (客户端) 的概念。任何一个节点, 只要在其上启动一个 UDP 服务, 网络内的其他节点就可以向其对应的(ip 地址, 端口号)发送数据包。只要发送方在 UDP 数据包中所包含的(ip 地址, 端口号)对应上目标节点, 这个节点就可以接收到该数据。UDP 通信底层没有握手和重发机制, 相当于发送方只负责发送出去, 接收方负责根据自己的(ip 地址, 端口号)过滤接收数据。

IP 地址可以是单播地址、也可以是组播地址、也可以是 255.255.255.255 这样的广播地址。所以, 当发送一个数据包里包含有广播地址时, 所有的端口号对应上的节点都可以收到数据。只有 UDP 才可以组播或广播。

因此, 完全可以在热点 AP 上建立起一个 TCP 客户端服务。

例如, 让模组 A 工作 AP 模式, 然后让 PC 去连接这个热点获取到 IP 地址。然后在 PC 上建立一个 TCP 服务器, 然后让模组作为 TCP 客户端去连接 PC 上的 TCP 服务器。

4.1.2 如何缩短 STA 连接 AP 或路由器的时间?

Q: 使用 WIFI (甚至包括有线以太网) 连接路由器时, 会发现连上网的时间有时候长, 有时候很短, 这是为什么? 如何尽可能缩短这个时间?

A: 粗略分析 STA 连接 AP 或路由器的过程, 有助于我们针对性地优化接入的时间。

(以下粗略的原理说明适用于 WIFI 连接, 不局限于 WIFI 模组)

STA 连接 AP/路由器的过程, 粗略可包括 2 个大致环节:

(1) STA 向 AP/路由器发送密码进行接入认证, 往往对应的提示是“正在连接....”;

(2) AP/路由器给 STA 分配/认可 IP 地址, 往往对应的提示是“正在获取 IP 地址....”。

这两个环节会牵涉到一系列的 STA 和 AP/路由器之间的数据处理和交互, 连接的时间也主要由这两部分组成。而连接总时间中, 环节 (2) 往往占据了主要比例, 所以, 当我们看连网过程提示时, 大多数时候看到的都是“正在获取 IP 地址.....”。

1、影响“STA 向 AP/路由器发送密码进行接入认证”过程时间的因素:

主要就是 (1) 路由器的认证处理较慢, 或 (2) 信号太弱或周围强干扰。

一些老旧路由器、或者某些路由器存在缺陷开机太久累积了垃圾卡壳效应、或者同时接入的 STA 太多, 都可能导致对 STA 的认证请求的处理较慢, 从而影响整体的联网成功的时间。

此时, 更换更好一些的路由器, 或者简单的复位一下路由器, 往往发现连网的速度就快了不少。

此外, 当信号太弱或周围存在强干扰时, 底层数据交换的可靠性降低, 会带来底层交互的重发次数增多, 也会导致连接时间拉长。

提高信号水平和抗干扰能力, 例如外接天线、靠近距离、屏蔽或避开干扰, 有可能加快连网的速度。

2、影响“AP/路由器给 STA 分配/认可 IP 地址”过程时间的因素:

AP 或路由器对 STA 的地址进行分配, 往往符合下述流程或算法:

(1) 接入认证时, 是否指定了静态 IP 地址?

如果是, 则校验地址的合法性。若合法, 则记录该 IP 地址和 MAC 的映射关系; 否则, 标记地址冲突。并结束分配/认可 IP 地址的过程。

如果否, 则进入下一步。

(2) 查看 AP/路由器上的绑定地址列表, 是否存在该 STA 的 MAC 对应的绑定 IP?

如果是, 则分配该 IP 地址, 反馈给 STA。并结束分配/认可 IP 地址的过程。

如果否, 则进入下一步。

- (3) 查看 AP/路由器上的缓冲地址列表, 是否存在对该 STA 之前分配过的 IP 地址?

如果是, 则分配该 IP 地址, 反馈给 STA。并结束分配/认可 IP 地址的过程。

如果否, 则进入下一步。

- (4) DHCP “复杂” 算法和处理, 计算出一个 IP 地址。分配该 IP 地址, 反馈给 STA。如果计算失败或卡壳超时, 则不返回, STA 端或提示获取 IP 地址失败。

上述 (1) 对应的就是我们常说的“设置静态 IP 地址”。当我们采用这种方式时, 很少出现连网很长提示正在获取 IP 地址的情形 (但是可能会提示“正在连接”, 因此此时依然会存在前面的接入认证过程), 这是因为在第 (1) 步就结束了, 不必进行后续的 (2) - (4) 的耗时步骤, 所以耗时往往最短。

但是, 设置静态 IP 地址有一个弊端, 就是“地址冲突”, 所以适合一些可以认为预先分配好 IP 地址的场合。如果运营部署支持指定静态 IP 地址, 并做好“地址冲突”的处理, 那么这会是缩短连网时间的一个好方式。

此外, 一般建议在 STA 发起连网认证请求之前就先设置好静态 IP 地址。这是因为, STA 设置静态 IP 后并不总是立刻将 IP 地址信息上传给 AP/路由器, 一般会在两种情形下上传: (1) STA 在连网认证的同时; (2) AP/路由器广播查询, 例如某节点要访问局域网内某 IP, AP/路由器查询发现自己的映射列表里没有这个 IP 地址, 就会广播询问这个 IP 地址的设备的 MAC 地址, 此时, STA 如果发现自己的 IP 地址和广播询问的相同, 也会上报。这也是为何, 如果我们后设 IP 地址, 出现“地址冲突”时的提示, 往往不在连网认证后马上出现的原因。所以, 为了节省网络资源, 同时及时地知悉地址冲突, 一般先设置好静态 IP 地址再发起联网。

上述 (2) 对应的绑定地址列表我们也经常用到--我们经常通过在路由器里来配置固定某个 STA 的 IP 地址。固定 IP 地址的本意可能是“锁定”某 STA 的 IP 地址不要变, 其带来的一个附带好处是, 大大缩短了分配 IP 地址的时间。这个地址列表一般是保存在路由器的 FLASH ROM 区的, 也就是掉电依然存在, 除非恢复出厂设置。

其优势是不存在 (1) 分配静态 IP 地址所导致地址冲突, 不足是需要去配置路由器, 在多数场景 (客户场景) 可能会存在一些运营上的困难。

上述 (3) 对应的缓冲地址列表是路由器等常见的一种机制, 就是一旦分配过的 IP 地址, 会临时缓存到某个缓冲区, 以后只要该 STA 来申请 IP 地址时, 就直接分配该 IP 地址。当然, 当新连接的数据缓冲记录较多之后, 这个缓冲区内较“老”而长期为使用的历史记录, 也可能被删除。这个机制可以解释我们常常见到的一些现象: (1) “某 STA 多次连接路由器, 发现分配的 IP 地址经常保持不变”、(2) “STA 设备首次连接路由器时, 时间可能会较长, 但是以后连接时, 就或很快了”、(3) “复位路由器后, 可能会导致 STA 设备重新首次连接路由器的时间加长”这是因为是“缓冲区”。

因此, 如无必要, 尽量别经常重启路由器、或让许多 STA 设备来反复重连路由器, 尽量避免使用太过老旧的路由器。有助于确保绝大多数时候的连网速度。

上述 (4) 提到的对应的 **DHCP “复杂” 算法和处理** 就是多数时候连网分配 IP 地址较慢时, 所达到的步骤。路由器的处理性能、当然前连接的设备数量和列表数量、信号的强弱, 等等, 都可能导致这里的时间拉长且比较显著, 尤其是在一些逻辑算法欠优化的路由器上, 表现更加明显。所以, 缩短连网时间的许多策略, 都是尽量避免进入这一步, 或者优化这里要素, 比如选择更可靠的路由器。

4.1.3 如何用两个 ALK8266WIFI®模组组成一个对等的 UDP 通信?

Q: 如何用两个模组组成一个对等的 UDP 通信?

A: 先让这两个模组之间建立网络连接, 然后建立各自的网络服务 SERVICE, 将目标的 IP 地址和端口分别指定为对方的。

第一步, 先建立网络连接。

如果这两个网络节点都通过路由器接入, 那么这两个网络节点本身就是连接的。

如果没有路由器或第三方提供的局域网, 可以将其中一个模组 A 作为 AP 建立一个局域网, 而让另外一个模组 B 来接入它并获取到 IP 地址。作为 AP 的模组 A 本身具有一个 IP 地址, 假设为 192.168.4.1。假设另外一个模组获取到的 IP 地址为 192.168.4.2。

第二步, 建立 UDP 服务连接。

假设准备指定各自的本地端口分别为 1234 和 4321

示意图如下

	模组 A	模组 B
配置的模式	AP	STA
IP 地址	192.168.4.1	192.168.4.2
本地 UDP 端口	1234	4321

MCU 主机配置函数

A: `ALK8266WIFI_Config_Connection_via_SPI(0, 1234, 192.168.4.2, 4321, 0);`

B: `ALK8266WIFI_Config_Connection_via_SPI(0, 1234, 192.168.4.1, 1234, 0);`

函数调用中最后一个 0 表示使用模组的链接号 (通道号), 最多有 4 个, 取值为 0~3, 这里举例用 0。

然后在各自的 MCU 主机程序里调用各自的发送和接收函数就可以了。

4.1.4 为何使用 TCP 通信往往比 UDP 要慢?

Q: 在做通信测速时, 同样条件下使用 TCP 通信的速度会什么会比 UDP 慢?

A: 这是因为 TCP 和 UDP 的底层机制有区别。UDP 底层只管发, 没有 ACK 握手和重发机制(参看 [4.1.1 AP/STA 服务器/客户端有何区别? AP 可做 TCP 客户端吗?](#)), 因此其他消耗对带宽的占用少, 所以常常会发现, 使用 UDP 通信的效率比 TCP 高。

4.1.5 路由器对 TCP 通信速度为何往往会有较大的(负面)影响?

Q: 在做通信测速时, 即使通信速度远远低于路由器的标称带宽, 但是为什么路由器对 TCP 通信速度, 相对于 UDP 通信, 依然可能会有较大的(负面)影响?

A: 这是因为影响路由器的实际“转发”性能的指标, 除了常见的表征路由器带宽 MBPS 参数之外, 还有一个很重要但是经常被大家忽略的、表征路由器处理能力的指标---MPPS (Mega Packet Per Seconds, 每秒的兆包数), 它表示了路由器每秒能处理转发的 IP 包的数量。

如果将路由器的带宽 MBPS 数值比作邮局每天进出的车辆的运载能力(单位吨), 那么, 包转发率 MPPS 可以类比邮局的分包拣包能力。一般地, 邮局每天处理的包的数量, 受限于人手(包括智能分包拣包), 所以都会有一个最大的限制。而每个包有多重(或多大尺寸), 对每天能处理的包的数量的影响反而不大; 只要每天处理的包的数量 \times 包的重量, 不超过车辆的运载能力就成。假设一个邮局车辆每天的运载能力是 1000 公斤(类比路由器的带宽 MBPS), 每天处理的包的数量最大不能超过 1 万个(类比路由器的包转发率 MPPS), 那么:

- (1) 如果平均每个包裹的重量远远小于 100g(类比网络上存在大量的小包数据), 那么, 车辆每天的运载能力会有富余。对应于在网络传输中, 很难达到路由器的极限带宽, 往往远远小于路由器的极限带宽。
- (2) 如果平均每个包裹的重量都接近 100g(对应于网络上存在大量的大数据包), 那么就会接近车辆的运输能力(逼近路由器的最大带宽)。
- (3) 如果邮局通过内部优化, 提高了分包拣包的能力, 从每天 1 万个包提升到 10 万个包(相当于换了一个包转发率 MPPS 更大的路由器), 那么也有助于发挥路由器的最大带宽性能(即使大量存在小包)。

在通过路由器进行 TCP 通信时, 因为中间存在大量的 ACK 小包, 所以, 当路由器的能力(MPPS)存在瓶颈时(尤其是对于某些便宜的路由器), 这些 ACK 包的存在, 会限制 TCP 通信的实际速度。

4.1.6 网络通信时为何建议一个 TCP 包的大小不要超过 1460 或类似的某个数值?

Q: 网络通信时为何建议一个 TCP 包的大小不要超过 1460 或类似的某个数值?

A: 这是因为网络通信上有一个“最大传输单元(MTU)”或“最大报文段长度(MSS)”的概念。每次传输的包的大小和 MTU 或 MSS 不相适应时, 可能会带来拆包、粘包(UDP 没有粘包现象)、以及降低通信效率或可靠性等一系列现象。可以 www.bing.com 搜索一下了解更多详情。

尤其是在一些高速实时性通信的场合, 拆包形成的小短包可能造成发送的效率降低; 同时, 因为拆包可能阵发包的数量连贯倍增, 有可能造成接收方来处理不及出现 ACK 丢弃而造成发送方因为重发次数增多而降低通信效率, 甚至可能因为重发次数出现越界而出现套接字状态复位。所以, 对于连续高速通信, 一般建议 TCP 包长度不要超过 MSS 个字节甚至更少以保持一定富裕量。对于当前网络条件下的 TCP 链接的 MSS 数值, ALK8266WIFI®模组支持通过下述 API 进行查询: `M8266WIFI_SPI_Query_Tcp_Mss()`, 以支持客户充分利用当前网络条件的最大效率。

4.1.6 建立 TCP 客户端服务, 为何建议随机产生本地端口而非指定?

Q: 为何建议随机产生 TCP 客户端的本地端口?

A: 随机生成的端口, 每次会不一样, 这样可以避免客户端连续使用相同的 IP 地址和本地端口去链接服务器时, 出现被拒绝链接的情形。

TCP 客户端在和服务器建立连接时, 会将自己的 IP 地址和端口传递给 TCP 服务器, TCP 服务器会维护这个 IP 地址和端口, 作为“存在这个客户端链接”的一种标志。而一旦客户端异常断开链接, 例如调试过程中的直接复位, 这种客户端侧的异常断开链接, 可能并不会通知服务器, 于是服务器将继续“维持该链接的记录”, 直到超时后删除。

如果客户端在这个超时时间段重新连接该服务器, 并使用了和上一次连接相同的本地端口, 那么在服务器那边, 就会出现和之前依然存在的 TCP 客户端链接完全一样的信息, 绝大多数服务器的处理方式, 会将其当成冲突而拒绝这个新的链接请求。在客户端这边就表现为, 异常断开后再次重连服务器, 会出现失败, 而需要等待较长时间后, 才可以继续使用这个端口再次去链接服务器。

为了避免出现这种情形, 建议本地的端口, 由客户端自行随机产生, 随机产生的端口往往每次会不一样, 所以可以避免出现被服务器拒绝链接的情形。

4.1.7 UDP 广播时, 速度为什么很慢不超过 100KBytes/s?

Q: UDP 单播测试速度有 1Mbytes/s, 但是改成广播后, 速度立刻降低到不到 100KBytes/s, 这是为什么?

A: 为了兼容各种不同的节点设备, 在 802.11 无线系统上, 广播帧(甚至也包括组播帧、以及一些管理帧), 都会以缺省最低的速率传播, 这个数值一般是 1Mbps, 大约相当于有效速度不到 100KBytes/s。当然, 在某些系统上, 这个缺省的最小值近来有所提高, 但是基本上都会为了兼容各种不同的节点设备, 广播的速度依然会采用一个较小的速率, 而非适配器本身所支持的速率。

4.1.8 地址冲突是什么意思? 地址冲突时为何 TCP 客户端会连不上 TCP 服务器?

Q: 模组 A 作为 AP+TCP 服务器, 模组 B 工作在 AP+STA 模式, 模组 B 链接模组 A 的热点可以连上并获取到 IP 地址, 此时模组 B 作为 TCP 客户端为何连不上模组 A 上的 TCP

服务器? 但是用电脑作为 STA 链接上模组 A 的 AP 后却可以连上模组 A 上的 TCP 服务器。

A: 有可能模组 B 的 AP 和模组 A 的 AP 所分配的网段出现重叠, 导致模组 B 试图去链接某个 IP 地址 (比如 192.168.4.1) 的 TCP 服务器时, 不知道对应的是模组 A 上的 IP 地址, 还是模组 B 自身的 IP 地址。

出厂缺省的模组 A 的 AP 的 IP 地址为 192.168.4.1。但是如果模组 B 也包含有 AP, 其 AP 下的缺省地址也是 192.168.4.1, 和模组 A 的 192.168.4.1 出现的重复。如果此时我们试图让模组 B 作为 TCP 客户端去链接 192.168.4.1 这个 TCP 服务器时, 模组 B 会面对两个 192.168.4.1 的 IP 地址, 一个是模组 A 上的, 一个是自己 AP 的。于是, 模组 B 可能会认为试图去链接自己这个 AP 对应的节点, 而不是去链接模组 A 的那个节点, 从而导致链接 TCP 服务器失败。

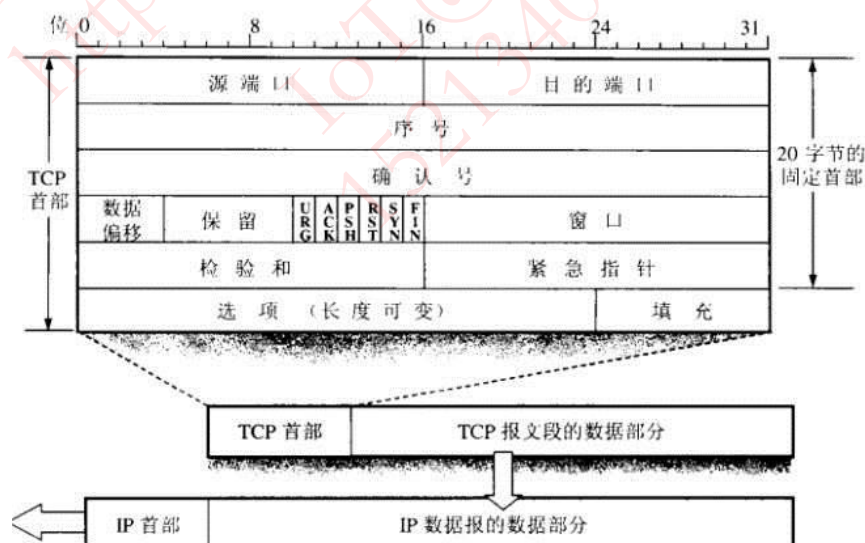
此时的解决办法有两个: (1) 修改模组 B 的工作模式, 从 AP+STA 模式改成 STA Only 模式, 那么此时模组 B 上不会有 AP 也就不会 192.168.4.1 这个 IP 地址, 也就不会和模组 A 上的 IP 地址冲突了。(2) 修改模组 A 或模组 B 的 AP 模式下的 IP 地址, 让两个处于不同的网段, 比如, 将模组 B 的 AP 的 IP 地址从缺省的 192.168.4.1 改成 192.168.5.1。这样, 当模组 B 试图链接模组 192.168.4.1 这个 IP 地址所对应的 TCP 服务器时, 只会找到模组 A 而不会出现地址冲突。

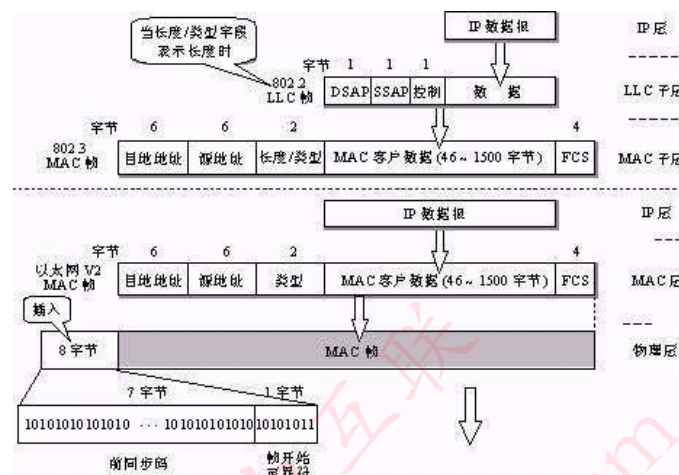
4.1.9 发长包的有效速度为何比发短包快?

Q: 连续发送测试时, 发小包显示的有效速度为何比发长包慢?

A: 这里有三个原因导致发长包所显示的速度比发小短包的速度快:

- (1) 在发送 IP 或 MAC 帧时, 报文段数据有效载荷 payload 的只占总包长或总帧长的一部分。payload 之外的部分长度基本固定 (至少 50 个字节以上), 因此, payload 越长, 则 payload 的占比越高, 发送的效率也就越高。





- (2) TCP 通信存在 ACK 占用带宽资源, 对于同样的数据量, 按照小包发送意味着更多的发送包, 也意味着会有更多的 ACK 占用资源, 因此有效通信效率也会降低。
- (3) TCP 通信启动之初, 一般都会有一个速度适配从低速开始逐步提升的过程。在某些小短包发送的场合, 尤其是发送间歇较大的场合, 大部分小短包的即时速度会处于提升的初期, 也就是会以较低的速度通信。

4.2 与天线相关的基础常识

4.2.1 什么是天线的增益? 为何增益越大, 对方向的对准要求越高?

Q: 天线增益是个什么概念? 为何天线的增益越大, 对方向的对准要求越高?

A: 天线本质上是一个将电磁波对外辐射的无源器件。如果天线只是空间里的一个孤立点, 不难理解, 其辐射的模式会是一个球体, 均匀地向四周辐射, 每个球面上的点所接收到的信号强度是一样的。而天线增益的本质, 是将向空间球面辐射的信号, 集中压缩到某个方向(平面上、圆柱圆锥、或甚至直线上), 从而增强某个接收方向上的信号强度。这种增强后的信号强度, 与没有增强之前自然球面辐射所得到的信号强度之比, 就是这个天线的增益。所以, 这种增益, 是以牺牲其他方向上的信号传播为代价, 来增强有用/期望的方向上的信号。

打个比方, 当我们用手做出喇叭状, 对着我们想说话的人时, 那个人听到的声音将会增强, 但是其他方向上的人听到的声音就会减小甚至可能听不见了。

因为天线本质上是将其他方向上的信号辐射, 压缩集中到某个方向上, 且增益越高的天线, 一般压缩的集中度越高, 方向方位越窄, 因此对于方向的对准要求或难度也就越高, 也可能会相应增加部署的难度。

购买天线时, 尤其是高增益天线, 一般需要参考其方向图, 找到其对应的最大方向, 确保通信方处于这个方向上。否则, 天线可能反而起到反效果。

4.2.2 为何高增益的天线有可能提高传输速度?

Q: 为何高增益的天线有助于提高传输速度(即使不接天线时的信号强度已经够强了)?

A: 和有线通信相比, 无线通信更容易受到周围环境的同频干扰, 接收方所处的环境, 不仅有来自所期望的发送方的信号, 同时, 在空间还存在着无数的其他同频信号。这些同等强度甚至更高强度的信号的干扰, 会导致接收方的接收效率降低, 从而表现出来的系统速度降低。

当发送方发送过来的信号以足够的强度到达接收方(接收方足够感知得到)时, 接收方不仅感受到这个所期望的信号, 也会同时感知到周围空间存在的其他信号。而接收方是否可以成功接收, 还取决于接收方对这些信号的处理/过滤能力, 或者说抗干扰的能力。对其他信号的过滤处理, 也会影响有用双方的通信效率/速度。

这个场景, 可以类比于两个人相距一段距离互相说话。如果周围没有其他人说话, 那么他们的交流会比较清晰顺畅。但是如果周围有其他人用相似的声音强度同时说话, 就会对这两个人的交流产生干扰, 导致一方无法听清楚或听完整另外一方的讲话, 结果就会是, 听话的一方只能听到部分而丢失了部分讲话, 或者说话的一方不得不多次重述, 于是交流的效率(速度)就降低了。即使原本两个人交流的声音足够强, 只要和周围其他人说话的声音没有形成明显的落差, 依然也免不了出现这种情形。

如果此时这两人,说话的一方用手做出一个喇叭状对准听话的一方,听话的一方会感觉到其声音明显压过周围其他人的声音,此时交流又会回归清晰顺畅,这是因为增强声音后,听话的一方就很容易将说话人所说的话,从其他周围的声音中区分开来。

在无线通信的场合,一般大量存在着许多足够强度同频信号。外接增益天线相当于就起到了上述类比中喇叭的作用,从而让接收方所接收到的信号强度明显超过周围的其他信号,减少接收方处理效率降低、接收丢包等因素导致的发送方不得重发等现象所带来的(表现)速度降低的情形。

从上面的描述也可以看出,增益天线对速度的贡献,有一个上限,这个上限是通信系统固有的最高速度。到了这个速度之后,即使提高天线增益,也不会再提高通信系统的速度。所以,外加天线增益只需要确保到达接收方的信号强度和周围的噪声有显著落差就可以了,没有必要过分强调增益,在周围很干净没有其他同频噪声干扰的场合,增益天线对速度的正面作用可能也不大。

4.3 WIFI 的传输距离与那些因素有关? 模组的传输距离有多远?

Q: WIFI 的传输距离与那些因素有关? 模组的传输距离有多远?

A: WIFI 的可接收的传输距离大体上主要受以下几个要素的影响:

- 1) 发射方达到接收方时的信号水平强度, 以及接收方的信号敏感度, 以及
- 2) 系统所期望或接受的速度要求

等因素所决定。

1 发射方达到接收方时的信号强度

(或者, 更严谨地说, 是发送方的信号作为有用信号, 相对于周围信号作为噪声, 所得到的“信噪比”, 能超过接收方的感知下限)

理解这个概念, 可以参照我们经常在电脑上扫描某个信号强度对应的信号数值。

这个要素, 往往包括以下几点:

(1) 发射方的发射功率。

发射功率越大, 传输到接收方的信号自然越强, 或者说, 在保持同样接收信号强度的前提下, 传输的距离可以更远。不过发射功率越大, 意味着系统的功耗也将越大。

(2) 发射方天线的增益

参考 [4.6.1 天线增益是个什么概念? 为何天线的增益越大, 对方向的对准要求越高?](#), 天线增益有助于将信号压缩集中到传输放上, 当然也就可以提高到达接收方的信号强度。

(3) 周围环境的衰减

例如墙壁或者潮湿的空气等等, 都可能会对信号的传输形成显著的衰减。

上述各个因素都不是完全由 WIFI 模组唯一决定的。

2 系统所期望或接受的速度要求

当到达接收方的速度较弱时, 系统的传输速度会因此而降低 ([4.6.2 为何高增益的天线有助于提高传输速度\(即使不接天线时的信号强度已经够强\)?](#)), 而较远的传输距离, 往往意味着到达接收方的信号会减弱。

因此, 如果对速度的要求不是很严格, 那么传输距离可以适当拉长。而如果希望传输速度接近通信系统的最大效能, 则传输距离不宜较远。

所以, 笼统地说 WIFI 模块的传输距离有多大, 其实是不严谨的, 尽管基本上说一个大致的范围并没有错。

3 本模组的传输距离

本模组实测结果显示, 在不外接天线时, 普通室外晴天环境, 视线距离 100 米左右, 依然可以保证速度不低于近距离速度的 70% (例如, 近距离测试时, 如果速度为 1M 字节每秒, 视线距离 100 米左右时, 速度不会低于 700K 字节每秒)。而在外接增益天线时, 强增益天线可以确保 1KM 以上的传输距离, 而视线传输距离在 500 米以上, 速度可以确保速度依然维持在近距离速度的 70% 以上。

4.4 如何减少射频的同频干扰, 例如产品上同时有 2.4G 蓝牙和 WIFI 模组?

Q: 如何减少同频干扰, 例如产品上同时有 2.4G 蓝牙和 WIFI 模组?

A: 减少同频干扰, 一般采取如下策略:

- 1) 让同频设备或组件在物理位置上尽量远离

例如进行产品设计时, 多个同频模组尽量远离放置, 针对不同模组的天线, 尽量远离。

- 2) 尽量避免让同频设备或组件同时工作

例如开启蓝牙时, 若不需要 WIFI 工作, 可以考虑临时关闭 WIFI, 反之亦然。

- 3) 合理的进行频道分配, 尽量错开带宽重叠

IEEE802.11 为 ISM 频段通信定义了 14 个频道, 每个频道的带宽为 20MHz+2MHz 的带宽, 频道的中心点频率如下图所示。

频道编号	频率 (MHz)	频道编号	频率 (MHz)
1	2412	8	2447
2	2417	9	2452
3	2422	10	2457
4	2427	11	2462
5	2432	12	2467
6	2437	13	2472
7	2442	14	2484

ALK8266WIFI®模组完全符合上述频道的定义。

从上图可见, 某个频道可能会和两旁±4 个频道发生带宽重叠, 所以, 频道号相差 4 个以上, 则一般不会出现重叠。类似的, 任何多个同频设备都可以进行类似地分析处理, 包括蓝牙设备和 WIFI 设备共存的场景。

ALK8266WIFI®模组有巧妙的处理将频道重叠的影响降低到最低, 因此一般不必担心 ALK8266WIFI®模组的频道配置, 主要考虑其他 2.4G 设备的相应处理即可。但若从产品最佳化的角度考虑, 也尽量做到避免带宽重叠。

ALK8266WIFI®模组支持频道设置。

如果工作在 AP 模式, 可通过 *M8266WIFI_SPI_Config_AP* 或

M8266WIFI_SPI_Config_AP_Param(param_type=AP_PARAM_TYPE_CHANNEL)

进行配置。此时, 配置模组的 AP 频道即为配置模组的射频频道。

但是, 如果模组此时的工作模式包含 STA 且连接上了第三方 AP 热点或路由器, 则模组的射频频道将跟随 AP 热点或路由器所在的射频频道。因此, 如果希望模组工作在 STA 方式下连接路由器时设置到某个频道下, 需要通过所连接的第三方 AP 热点或路由器进行设置。

特别地, 有些用户的多射频器件或高功耗器件在一起同时工作时, 遇到的一些异常, 并非一定都是射频干扰所致, 相当一部分问题, 其实是主板本身的设计缺陷导致了多组件之间的基带信号的传导干扰或电源完整性的问题, 所以, 在分析问题, 也不要局限于射频方面而忽视了非射频因素导致的协作异常。

5 与高速通信应用有关的常见问题

5.1 单片机通过 WIFI 模组上传文件, 对方是一个 FTP 服务器, 需要在单片机上移植一个完整的 FTP 客户端吗?

Q: 单片机通过 WIFI 模组上传文件, 对方是一个 FTP 服务器, 需要在单片机上集成一个完整的 FTP 客户端吗?

A: 没必要。

单片机是一个小型的智能系统, 其优势在于精简, 如果仅仅为了传输一个文件, 在单片机上集成一个完整的 FTP 客户端, 不仅浪费资源, 降低效率, 而且可能因为单片资源不够而无法实现。

所以, 我们完全可以只抽取其中所需要的部分功能, 实现和 FTP 服务器的正常通信和文件上传。具体做法, 我们称之为“模拟复制”实现方法, 具体说明如下:

- 1、找一个 FTP 客户端工具, 执行操作, 向 FTP 服务器上传一个文件, 不需要有其他的操作。

- 2、在 1 中操作的同时, 打开 Wireshark 工具, 对此过程进行抓包。

这个工具很强大, 还会帮助你分析各个包, 所以, 很容易看到 TCP 包的内容。

- 3、将 2 中得到的 TCP 包的内容, 在 TCP-USR-232 等工具里, 以 TCP 包数据的形式直接发送, 观察服务器端的反应, 进行验证。

- 4、将 3 中验证正确的 TCP 包数据, 转移到单片机中, 作为 TCP 包向 FTP 服务器发送并处理服务器的相关的必要反馈, 即实现在单片机上的简单的 FTP 文件上传。

5.2 音视频的录-传-播同步, 需要注意什么?

Q: 在单片机上测试 WIFI 模组的通信速度能到 1M 字节每秒, 而且不丢数据, 但是为什么用来传输视频播放时(需要的速度只有 500K 字节每秒)时, 依然出现视频卡顿的情形? 这个问题该如何解决?

A: 这是因为, 受网络环境因素等各种因素的影响, 网络通信的速度一般都是波动的, 很难保持恒速, 因为网络通信是一种流控制, 而非实时传输。但是视频的录或播, 往往是时间均匀的, 所以存在一种可能性, 就是在某个时刻, 网络的传输速度低于了视频播放所需要的速度, 而可能因为播放器没有数据输入播放而出现了卡顿。

恰当地使用先进先出缓存, 则可以很好地解决音视频录传播的同步问题。

缓存的作用可以理解成, 当网络的速度超过播放的速度时, 可以先预存一部分先进先出的数据, 以便在网络速度偶尔低于播放速度时, 可以有这部分被缓存的数据接力播放。

这个缓存的大小, 需要根据实际的情形做优化处理, 如果网络速度波动容易加大较长, 则缓存可能会需要较多。

我们看有一种体验, 许多在线播放器, 一般都一个缓冲区参数设置。当我们将这个参数设置得太小, 播放时就会经常出现卡顿, 即使我们的网速够快。但是一旦我们将这个缓存设计得合适了, 播放会非常流畅。而如果将这个参数设置得太大, 也是一种浪费。此外, 我们也会看到, 网络传输并非一直都在进行, 而是会根据缓存里的数量大小, 时而启动, 时而停止。这些, 都可以用来作为单片机做音视频同步录传播的思路参考。

所以, 做实时录传播, 不只是 WIFI 模组的速度越快越好, 而只要 WIFI 模组的平均速度能超过录和播的速度就可以了, 重要的是, 处理好先进先出的缓存机制。

5.3 采用例程测试速度有 1MBytes/s, 但加上高速采集后系统速度降低一半, 这是为什么? 该如何优化?

Q: 执行例程测试速度稳定在 1Mbytes/s, 加上高速采集模块代码后, 速度降低到 450KBytes/s, 这是为什么? 该如何优化? 采样速度为 800KB/s。

A: 这可能是因为单片机串行而非并行执行的结果, 导致“系统”的速度降低。

假设每次采集 800 个字节就发送一次, 那么, 在串行执行模式下, 以 800KBytes/s 采集速度, 采集 800 个字节需要占用 CPU 平均约 1ms 的时间, 然后以 1MBytes/s 的速度通过 WIFI 发出去, 则需要平均约 0.8ms。所以, 从整个系统的角度来看, 完成 800 个字节的采集和传输, 占用了 1.8ms 的时间, 和系统平均速度为 $800/1.8=444\text{KBytes/s}$ 。

在这里可以看到, 系统的速度不仅远远低于模块的及时通信速度 1Mbytes/s, 也远远低于高速采集的速度。这是因为, 串行执行的结果导致模块和采集器都未发挥其最大性能。

采用并行技术, 辅以 FIFO 缓存, 例如, 将高速采集改成 DMA 而非由单片机来执行处理, 相当于有“2 个 CPU”在流水线并行处理, 可以大大提高系统整体的吞吐速度, 最终的结果可以让系统的吞吐速度逼近 800KBytes/即采集的速度。

5.4 怎么通过这个模块获取互联网时间?

Q: 怎么通过这个模块获取互联网时间?

A: 通过 WIFI 模块获取互联网时间非常简单, 只要让模块接入互联网, 作为客户端去访问某个互联网服务器就成。

步骤 1: 确保模块接入了互联网, 比如, 链接 1 个可以接入互联网的路由器

步骤 2: 建议一个 TCP 链接, 模块作为 TCP 客户端, TCP 服务器的参数如下:

目标地址: www.beijing-time.org

目标端口: 80

步骤 3: 向这个服务器发送 GET 请求, 获取时间, 即将如下字符串作为 TCP 包的数据发送出去:

"GET /t/time.asp HTTP/1.1\r\nHost: www.beijing-time.org\r\nUser-Agent: Mozilla/5.0\r\n\r\n"

步骤 4: 从所接收到的数据中, 解析出互联网时间来。

接收到的数据样例如下 (或许会有细微的差别), 末尾就是互联网时间

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html
Date: Sun, 14 Feb 2021 06:31:16 GMT
Content-Length: 98
t0=new Date().getTime();
nyear=2021;
nmonth=2;
nday=14;
nwday=7;
nhrs=14;
nmin=31;
nsec=16;
```

小提示: 这个服务器后台有可能会改变获取时间的GET请求的具体格式, 也就是说不一定总是 “/t/time.asp” (例如2年前曾是/time15.asp)。可以先发送 “GET / HTTP/1.1” 根据返回提示进行分析, 得到目前获取时间的GET请求的具体格式, 如下所示:

```
HTTP/1.1 200 OK
Content-Type: text/html
Last-Modified: Sun, 24 Jan 2021 04:29:56 GMT
Accept-Ranges: bytes
Date: Sun, 14 Feb 2021 06:36:49 GMT
Content-Length: 8921
<!DOCTYPE html>
<html>
<head>
<title> </title>
<link rel="canonical" href="https://www.beijing-time.org/">
<link href="//www.beijing-time.org/css/css.css" rel="stylesheet" />
<script language="Javascript" src="//www.beijing-time.org/t/time.asp"></script>
<script language="Javascript" src="//www.beijing-time.org/js/beijingtimeorg.js"></script>
</head>
<body>
.....
```

这其实也是许多服务器常用的方式: 统一不变的URL入口 (本例子里是 “/”, 然后根据一些算法 (例如所处的地理位置、或特定的功能), 重定向到会返回目标URL (本例子里是 “/t/time.asp”), 而重定向到的URL可能会改变或更新。

当然, 如果有自己的服务器, 也可以自定义直接从该服务器上获取互联网时间。

5.5 模组通过路由器联网做 TCP 服务器, 不知其 IP 地址怎么办?

Q: 模组通过路由器联网, 每次都是 DHCP 获取到动态地址。所以客户端去链接它时如何知悉其 IP 地址?

A: 有两种方法可以解决这个问题: 方法一, 对 TCP 服务器模组设定静态 IP 地址; 方法二, 利用 UDP 的广播属性获取做 TCP 服务器的模组的 IP 地址。

方法一, 静态 IP 地址:

可以调用 API 函数 `M8266WIFI_SPI_Config_STA_StaticIpAddr()` 来对模组预设静态 IP 地址。

详情可参看《[ALK8266WIFI 模组 SPI 接口高速通信使用与集成—主机驱动 API 函数](#)》中的章节“4.2.5 M8266WIFI_SPI_Config_STA_StaticIpAddr”

方法二, 利用 UDP 的广播属性获取做 TCP 服务器的模组的 IP 地址

我们知道, UDP 相较于 TCP 的一个优势是, 支持广播帧 (目标地址为广播地址), 广播帧可以被网段内的所有节点接收到, 然后比较目标端口是否一致决定是否接受处理。我们可以利用这个特点来提取做 TCP 服务器的模块的 IP 地址和端口信息。

同时, ALK8266WIFI® 模组支持多链接, 可以方便的配合实现对 TCP 服务器的 IP 地址的提取。

具体说明如下 (为了方便说明, 我们以下称做 TCP 服务器的模组为节点 A, 而准备作为 TCP 客户端需要去链接 TCP 服务器的节点--可能是手机、电脑或者另外一个模组--称为节点 B):

- (1) 节点 A 上电后, 除了建立一个 TCP 服务器服务外, 同时也建立一个 UDP 服务, UDP 服务的本地端口预先指定已知。
- (2) 节点 B 在连接节点 A 上的 TCP 服务器之前, 先建立一个 UDP 服务, 来发送一个 UDP 广播包。这个广播包的目标地址为广播地址, 例如 255.255.255.255, 端口为节点 A 预先设定的 UDP 本地端口, 广播包的内容可以是自己定制的数据协议格式, 表示“询问某某 TCP 服务器的 IP 地址”。
- (3) 因为节点 B 发送的 UDP 广播包的目标端口是节点 A 上 UDP 服务的端口, 而目标地址是广播地址, 节点 A 的 UDP 服务会接收到这个包, 解析后知道是节点 A 在询问自己的 IP 地址。在接收这个 UDP 的同时, 节点 A 也知道了节点 B 的 UDP 服务的 IP 地址和 UDP 本地端口。

于是节点 A 可以以节点 B 的 IP 地址和 UDP 本地端口为自己的目标地址与目标端口, 返回一个 UDP 包 (此时就不需要是广播包了), 包的内容包含了节点 A 自己的 IP 地址。

- (4) 于是节点 B 就知道了节点 A 的 IP 地址了。于是就可以继续作为 TCP 客

户端建立对节点 A 的连接了。

对方法二的补充说明:

- (1) 上述方法不仅仅可以传递 TCP 服务器的 IP 地址, 也可以用来传递 TCP 服务器的端口等其他信息。只要定制好 (2) (3) 所有提到的数据协议格式就成。
- (2) 上述方法也可以用于接入认证等功能实现。比如在方法二的 (3) 中, 节点 A 同时会对数据的有效性进行分析, 合法的才返回自己的 IP 地址和 TCP 服务器的端口, 才允许被连接。

比较方法一和方法二:

- (1) 方法一, 直接设置静态 IP 地址, 实现简单。

但是可能存在地址冲突的现象, 即指定的静态 IP 地址可能已经被本网段其他的节点给占用了。此时可以在路由器里绑定 IP 地址 (需要现场的路由器支持这么做, 且不会给运营带来麻烦)。

- (2) 方法二, 实现灵活, 适应面广, 推荐使用这种方法。

但是实现起来可能不如方法一简单直接。其实也不难, 只是比方法一稍微多几个步骤。

5.6 如何让 WIFI 模块通过网页实名认证的网关上网?

Q: 如何让 WIFI 模块通过网页认证的网关上网? 就是在宾馆、机场、医院、或校园网那种场景下通过网页认证的方式实现上网。

A: 这个问题, 其实和本 WIFI 模组无关, 下面的解决办法适合所有的 WIFI 模块。

要让 WIFI 模块来通过网页认证来实现上网, 首先有必要先理解一下网页实名认证的认证机理。当一个手机或电脑 (有屏幕人机接口), 在在宾馆、机场、医院、或校园网里, 通过网页实名认证的过程, 一般是这样的:

- (1) 手机或电脑的网卡通过 WIFI 或有线以太网, 连接上本场景下的热点 AP 或路由器。
- (2) 手机或电脑“首次”打开浏览器打开某个网页后, 热点或路由器所对应的网关, 会首先拦截该网页访问请求, 并将该请求强制转移到一个指定的登录页面, 要求输入认证的用户名和密码。

手机或电脑打开网页, 有可能是基于强制入口模式自动弹出的网页, 也可能是用户手动打开的浏览器。总之, 最终会被转移到一个缺省的登录页面。

- (3) 用户在该页面输入认证的用户名和/或密码, 提交到网关。
- (4) 网关收到网页提交的请求后, 会进行认证, 如果认证通过, 网关会记录当前手机或电脑的 MAC 地址等信息, 加入允许列表。

以后这台手机或电脑, 就可以直接访问互联网, 而不会被网关拦截或转移, 从而实

现了互联网自由冲浪

了解了上述机理后,我们就明白了,关键的是步骤(1)中“连接局域网的热点 AP 或路由器”,以及步骤(3)中能“将用户名和/或密码提交到网关进行认证的接口”。至于(2)中打开一个网页,只是为了(3)中的提交,所以是否需要打开一个网页这一点并不重要;而(4)中网关的处理,由网关后台负责完成,不需要 WIFI 模块的太多介入和关注。

因此,让 WIFI 模组去模拟手机或电脑通过网页实名认证的做法,可以步骤如下:

- (1) 让 WIFI 按照普通的方法,首先连接上热点或路由器;
- (2) 单片机在 WIFI 上创建一个套接字,去连接网关所对应的服务器,并按照网页表单提交的格式(一般是 HTTP POST 方式,可参考 [5.4 怎么通过这个模块获取互联网时间?](#) 了解如何通过 WIFI 模组发送 HTTP 协议包),发送一个数据包(包含认证的用户名和密码),完成认证;
- (3) 然后网关不再拦截或转移,可以进行正常的上网访问。

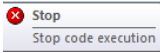
小窍门:

- (1) 上述步骤(2)中,如果不知道认证网关的服务器信息(含 IP 地址和端口)与提交表单的格式,可以在电脑上打开一个 Firefox 之类的浏览器(这类浏览器可以显示网页交互过程中的所有网页请求和相应),来捕获分析上述机制过程步骤中的相关网页信息包括 POST 等信息,可以找到其中的网关服务器信息和提交表单对应的 POST 格式信息。
- (2) 大多数场景的实名认证机制,都是明文方式的,所以都可以轻松实现。但是也有少数个别场景采用了 SSL 等加密访问方式,那么,就得联系这些场景的网关,得到加密访问的证书和密钥等信息,通过 SSL 方式访问实现。
- (3) ALK8266WIFI 模组支持多链接和 SSL 访问,所以可以支持上述需求的实现。

6 与单片机基础性开发调试有关的常见问题

6.1 为何跑飞进入 HardFault_Handler 死循环?

Q: 执行例程时一切正常, 速度很快很稳定。但移植到自己的工程后, 发现程序跑飞, 调试

方式下执行“停止运行”() 操作后, 发现程序进入 HardFault_Handler 里死循环。这是为什么?

A: “硬故障”这个问题, 其实是单片机的一个常见的基础性问题, 本身和 WIFI 模组及其驱动无关。

出现这类问题, 通常是因为系统栈设置相对小了, 导致了栈溢出。应适当加大系统栈。

当出现“硬故障”时, 单片机会进入这个故障的处理程序, 而这个故障处理程序常常缺省为一个死循环, 所以对外表现出来的是程序跑飞。尤其在单步调试时, 进入这个故障处理程序后, 单片机并不自动停止, 因此常会让人误以为是当前所执行函数跑飞进入了死循环。此时, 必须在调试界面点击“停止运行”后, 才会发现进入了这个故障的处理函数 HardFault_Handler 里了。

造成“硬故障”的原因往往比较复杂, 多数是与片内总线之间的访问有关, 最常见的是内部数据访问越界。有兴趣的朋友可自行 www.bing.com 关键字 hard fault 了解相关细节。

此处从简, 说一个使用我们模组时遇到此问题的用户的几乎是唯一的原因(实际也是其他许多单片机工程遇到此类问题的唯一原因): **栈溢出了**---如果在某硬件平台上执行例程一切正常, 移植到自己的工程后就出现“硬故障”, 往往基本都是自己的工程里定义了相对较小的栈(STACK)上限, 而实际需要用到的栈(STACK)较大, 导致栈溢出了。

通常, 临时变量都是保存在栈(STACK)里。当使用到的临时变量比较大(例如大数组, 在高速通信大块数据通信场景下使用了较大的数组临时变量, 来缓存需要发送或接收的数据), 意味着可能需要更大的栈占用, 如下图所示。

```
#define TEST_SEND_DATA_SIZE    2920
// If using large size of array, ensure system stack is large enough for the array variable.
// Or stack over-bound leakage might bring about the mcu into "hardware fault exception"
// (Chinese: 如果使用较大的数组, 记得确保有足够大的系统栈来容纳这个大数组变量.
//否则, 单片机程序可能会因为栈溢出越界而跳入"hardware fault"系统异常)
u8  snd_data[TEST_SEND_DATA_SIZE];
```

多数工程, 尤其是复制或自动化生成的工程, 初始栈的数值往往设置得比较小如 0x400 才 1024 个字节, 容纳不了上述数组, 所以, 运行时会导致栈溢出。

解决办法: **根据工程使用栈的实际情况, 适当调整加大系统栈就可以了**。如下图所示:

```
Stack_Size      EQU      0x00001000 ; changed from 0x00000400(=1024) originally
```

7 如有不清之处, 可随时通过 QQ 或微信等手段和我们进行交流

随时随地的自由互联
<http://www.anylinkin.com>
<http://anylinkin.taobao.com>
IoT@anylinkin.com
1521340710@qq.com

非公开声明

本文档及其相关资料(包括相关文档和参考例程等),仅授权购买我司高速 WIFI 模组 ALK8266WIFI® 的客户(公司)使用参考,其他人员不得使用或参考。

凡是经过我司正常渠道接收本文档及其相关资料的用户,即获得使用本文档及其相关资料的授权。

未经我司同意,授权客户不得对外公开、分享,或转让本文档及其相关资料的部分或全部。