通信协议理解手册

**目录**

**[一、](#bookmark1)** [通](#bookmark1)  **篇导读** 2

二**、协议分类及理解** 2

**[三、IIC](#bookmark3)****[4](#bookmark3)**

**[四、](#bookmark4)**[S P I](#bookmark4) **[5](#bookmark4)**

**[五、](#bookmark5)**[U A R T](#bookmark5) **[5](#bookmark5)**

**[六、](#bookmark6)** [C A N](#bookmark6) **[6](#bookmark6)**

**[七、](#bookmark7)** [W I F I](#bookmark7) **[7](#bookmark7)**

**[八、](#bookmark8)****[USB](#bookmark8)****[7](#bookmark8)**

九**、时序图参考** 10

**[十、](#bookmark10)****[储存数据及连接电脑使用](#bookmark10)****[10](#bookmark10)**

**[十一、](#bookmark11)****[安全注意事项](#bookmark11)****[11](#bookmark11)**

**[十二、](#bookmark12)****[数据接口](#bookmark12)****[12](#bookmark12)**

**[十三、](#bookmark13)****[外形及安装尺寸图](#bookmark13)****[12](#bookmark13)**

**[十四、](#bookmark14)****[随机附件明细表](#bookmark14)****[13](#bookmark14)**

**一、通篇导读**

通信协议广泛应用于生活，但是我们对其中神秘的信号起始，数据解析还有种种之类的机制仿佛不是非常了解，下面来让我为大家解析一下信号的底层，做到知其然也知其所以然。

**二、协议分类及理解**

**目前我所接触了解到的常用通信协议大致分为有线通信协议和无线通信协议，两者各有优缺点，有线一般意味着稳定可靠高速率，无线意味着方便传输距离远。关于有线协议的理解目前我是通过具体的IIC或者SPI协议的通信方式反推进行理解的，包含片选引脚，信号引脚，数据引脚，一般有线协议的数据传输是通过电信号的方式进行传输的，而传输的数据底层样式其实就是二进制，我们的电平信号通过设置高低就可以表示0和1进行二进制数据的传输，整个的流程大体如下：[通过SCLK上下升沿的方式或结束一个字节的传输]**

**拉低 CS（片选）引脚：主设备将从设备的 CS 引脚拉低，表示即将开始与该从设备通信。**

**生成 SCLK 信号：主设备生成时钟信号 SCLK，为后续数据传输提供同步。**

**3. 数据传输**

**数据传输按位进行，每次 SCLK 周期传输一位数据。以下是一个字节（8 位）数据传输的详细步骤：**

**发送第 1 位数据：**

**主设备在 SCLK 的第一个有效边沿（由 CPHA 决定）将第 1 位数据放到 MOSI 上。同时，从设备在 MISO 上放置第 1 位要发送给主设备的数据。**

**在 SCLK 的下一个有效边沿，主设备采样 MISO 上的数据，从设备采样 MOSI 上的数据。**

**发送第 2 位到第 8 位数据：**

**主设备继续按照 SCLK 周期，将每一位数据放到 MOSI 上，接收从设备通过 MISO 发来的数据。**

**从设备则在每个 SCLK 周期采样主设备的 MOSI 数据，同时发送自身的下一位数据。**

**通过 8 个 SCLK 周期完成一个字节的数据传输。**

**完成字节传输：**

**当所有位（例如 8 位或 16 位）发送完成时，主设备和从设备完成一个数据帧的发送。**

**需要发送更多数据时，主设备继续生成时钟并将下一字节的数据放入 MOSI。**

**4. 数据接收和校验**

**主设备和从设备在 SCLK 的每个有效边沿采样数据，完成数据接收。**

**通常主设备在接收完数据后会进行校验（如根据协议要求），确认数据的完整性。**

**5. 通信结束**

**拉高 CS 引脚：主设备将 CS 拉高，表示本次通信结束，从设备停止发送数据。**

**停止 SCLK 信号：主设备停止生成时钟信号，通信完成。**

**6. 多从设备场景（可选）**

**在多个从设备的 SPI 通信中，主设备会使用多个 CS 引脚分别连接各个从设备：**

**需要与某个从设备通信时，主设备仅拉低对应的 CS 引脚，其他从设备的 CS 引脚保持高电平。**

**每次通信结束后，主设备将所有 CS 引脚拉高，确保没有其他从设备处于选中状态。**

### SPI 通信的完整时序流程总结

| **步骤** | **主设备操作** | **从设备操作** | **时序引脚变化** |
| --- | --- | --- | --- |
| **初始化** | 设置 SCLK、MOSI、MISO、CS 引脚 | 配置 SPI 模式 |  |
| **通信开始** | 拉低 CS 引脚 | 检测到 CS 拉低，准备通信 | CS 拉低，SCLK 开始 |
| **数据传输** | 按 SCLK 的节奏传输 MOSI 数据 | 按 SCLK 的节奏传输 MISO 数据 | 在 SCLK 的边沿，MOSI 和 MISO 传输数据 |
| **数据接收** | 按 SCLK 采样 MISO 数据 | 按 SCLK 采样 MOSI 数据 | 每个 SCLK 周期采样一次 |
| **通信结束** | 拉高 CS 引脚，停止 SCLK 生成 | 检测到 CS 拉高，结束通信 | CS 拉高，SCLK 停止 |

**无线通信协议：**

**上面我们以SPI举例，具体阐述了他的通信过程，下面我们来介绍无线通信的相关流程：**

**WIFI**

**A. 扫描信道和找到网络**

**扫描：设备先打开无线电功能，扫描附近的信道，寻找可以连接的 Wi-Fi 网络（通常显示为网络名称，SSID）。**

**找到信道：设备找到可以使用的信道，并选择要连接的 Wi-Fi 网络。**

**B. 建立连接和身份认证**

**发送连接请求：设备选择好网络后，向 Wi-Fi 路由器发送一个“我想连接”的请求。**

**认证：如果 Wi-Fi 设置了密码，设备和路由器之间会交换密钥（加密信息），确保只有你知道密码的设备可以连接。**

**C. 数据传输过程**

**发送数据：连接成功后，设备会通过无线电信号将数据发送到路由器，数据被分成很多“小包”（数据包）发送。**

**确认收到：每当路由器接收到一个数据包时，会发回一个“收到”的信号。如果设备没有收到确认，它会再次发送这个数据包，直到收到确认。**

**多设备使用信道：如果有多个设备连接到同一个路由器，路由器会安排它们“轮流”发送数据，以减少干扰。这类似于多辆车交替通过一个窄桥，确保不会发生冲突。**

**D. 断开连接**

**主动断开：当设备需要断开连接时，它会发送一个“断开请求”给路由器。**

**释放信道：路由器收到请求后，会释放信道，这样其他设备可以使用。**

**4. Wi-Fi 使用的调制和加密方式**

**调制：为了让数据通过无线电波传输，Wi-Fi 使用了调制技术。你可以理解为“编码”，将数据转换成适合无线传输的信号。**

**OFDM（正交频分复用）：这是 Wi-Fi 常用的调制方式，它将数据分成多个“小块”并通过多个小的频率传输，使得信号更抗干扰，传输速度更快。**

**加密：为了确保数据安全，Wi-Fi 网络通常会加密传输的数据，只有知道密码的设备才能解密。最常见的加密方式是 WPA2 加密，它通过多次“握手”确保数据只在设备和路由器之间流通，防止中间被拦截。**

**上面的流程了解即可，用我们自己的话来理解阐述:首先，WIFI信号其实就是无线电波信号（电磁波信号），其中的参数2.4GHZ和5GHZ实质上就是表示振荡次数（GHZ:每秒十亿次），这些信号可以穿透物体和障碍物进行传输，而信道则可以想象成车运输的通道，也就是公路，一般2.4GHZ和5GHZ分别有13/23个信道，在不同的信道工作就可以减少因为频率数据传输相同相撞的可能性，当我们开启WIFI功能实质上就是开启一个无线电功能，扫描信道，寻找SSID（WIFI标识符），然后选择连接，就会向路由器发送一个我想连接的信号，此时的路由器就会产生认证的过程，也就是你的设备和路由器进行密钥交换，匹配即可，连接上之后就是数据层的事情了，为啥多人连接WIFI会卡呢，其实就是因为堵车机制，WIFI会采用轮流发送的形式减少干扰，避免多辆车一起走一个道引起堵塞，关于为啥只有2.4G和5G呢，其实是国际上规定了民用的频率只有这两个，低频率意味着穿透力强，传输范围广，但是传输量和传输效率低下，高频率意味着信道多和带宽更大，可携带的数据更多。而信道其实就是中心频率和带宽的综合作用结果，2.4GHZ中，2.401+-5MHZ，2.410+-5MHZ，可以理解为两个不重合的频段。还有就是类似与有线通信的数据层格式，其实也是使用调制，使用频段的物理层形式将数字信号（0/1）转化成模拟信号（无线信号）。**

**讲完有线和无线，再来讲解一下整体通信协议框架：**

**物理层（Physical Layer）**

**负责传输原始的比特流，即 0 和 1 的电信号或光信号。**

**定义了物理连接的标准，如电缆类型、接口、信号电压等。**

**数据链路层（Data Link Layer）**

**提供可靠的节点间数据传输，主要负责帧同步、流量控制、错误检测。**

**分为两个子层：逻辑链路控制（LLC）和介质访问控制（MAC），用于决定设备如何在共享介质上发送数据。**

**网络层（Network Layer）**

**负责路由选择，通过路由器将数据从源端传输到目的端。**

**提供逻辑地址（如 IP 地址），用于在不同的网络之间找到路径。**

**传输层（Transport Layer）**

**提供端到端的通信服务，确保数据的完整性和顺序。**

**使用传输协议如 TCP（可靠连接）和 UDP（无连接）来控制数据传输的可靠性。**

**会话层（Session Layer）**

**负责建立、管理和终止会话（通信会话）。**

**处理会话控制、对话管理和同步，确保两个设备可以维持连接。**

**表示层（Presentation Layer）**

**负责数据格式的转换，确保数据在不同系统之间的兼容性。**

**包括数据加密、解密、压缩、解压缩等功能。**

**应用层（Application Layer）**

**直接面向用户，提供网络服务的接口，例如 HTTP、FTP、SMTP 等协议。**

**处理应用程序之间的通信，是用户与网络交互的接口。**

**数据传输在七层中的流程**

**数据在发送时，从应用层开始向下传输到物理层，每一层在传输过程中添加必要的协议头或尾，完成通信需求。接收端则从物理层向上逐层解析，还原数据直到应用层，使得通信完成。**

**用自己的话来阐述就是：OSI七层模型，我们的嵌入式通信主要关注的是物理层和数据链路层，其他的则是有纯软的人来负责，大致流程就是将应用层从上往下解析到物理层进行发送，接收则是由下往上解析，我们这方面的功夫可以放在如何理解整个流程和手写底层上面（因为有些场景可以简化部分通信协议来进行通信，这一点STM32做的挺好，基本上都是自己来搓底层，我们一般使用Arduino来简化开发，Arduino上有许多大佬的开源库，可以参考他们的思路来进行自己的定制改造）**

补：后续进行陆续补充，USB和CAN暂时没用到，用到再实时补充好了