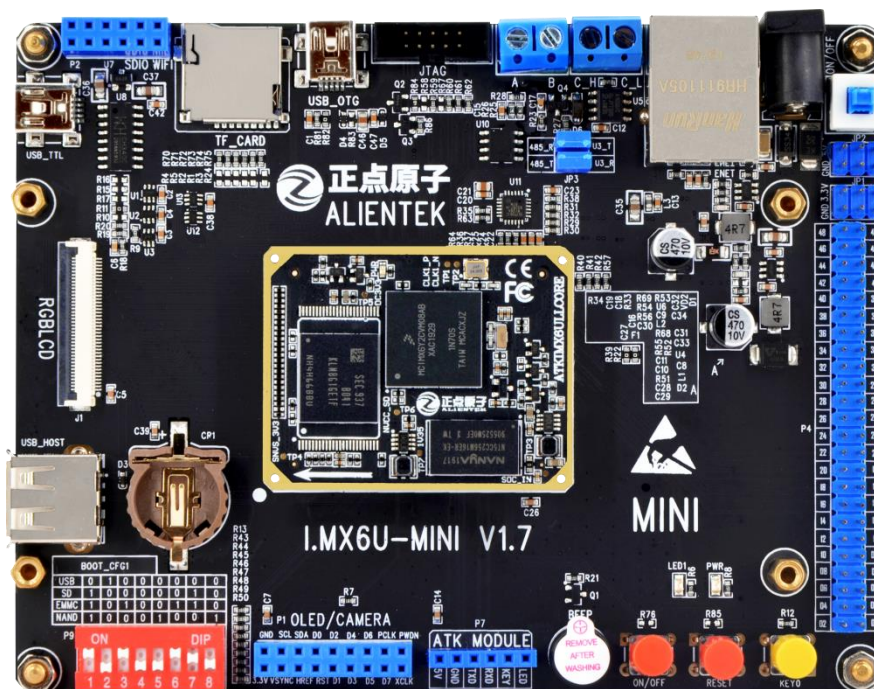
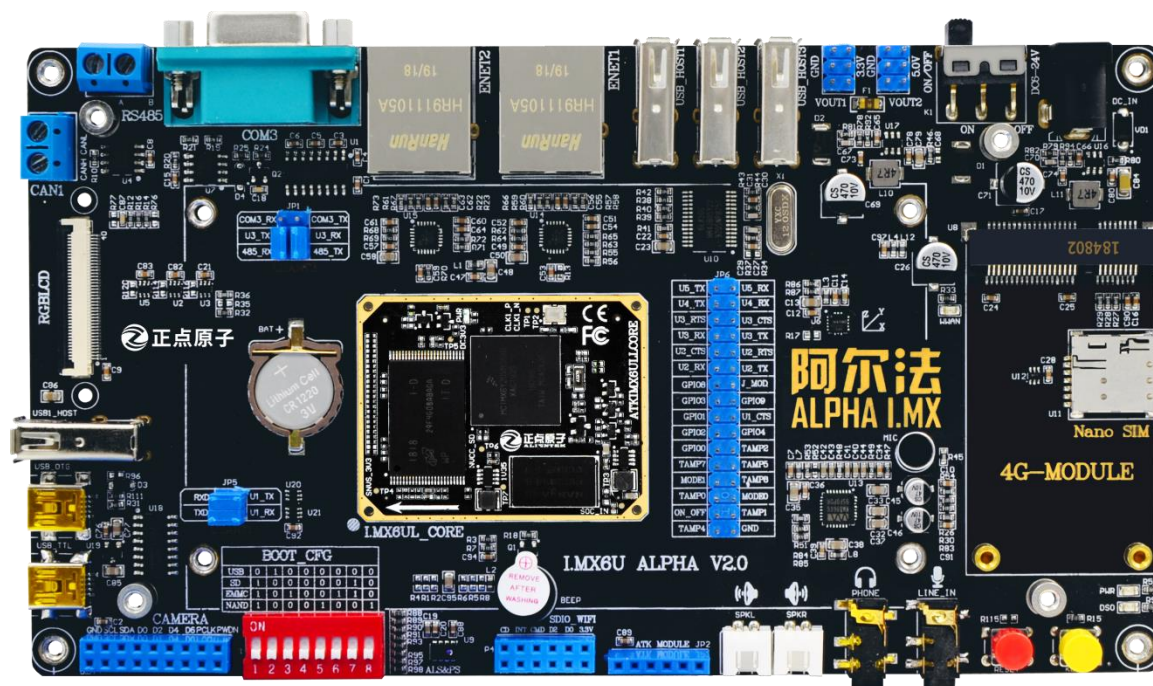


I.MX6U 网络环境 TFTP&NFS 搭建手册 V1.3.1





正点原子公司名称 : 广州市星翼电子科技有限公司

原子哥在线教学平台 : www.yuanzige.com

开源电子网 / 论坛 : <http://www.openedv.com/forum.php>

正点原子淘宝店铺 : <https://openedv.taobao.com>

正点原子官方网站 : www.alientek.com

正点原子 B 站视频 : <https://space.bilibili.com/394620890>

电话: 020-38271790 传真: 020-36773971

请关注正点原子公众号, 资料发布更新我们会通知。

请下载原子哥 APP, 数千讲视频免费学习, 更快更流畅。



扫码关注正点原子公众号



扫码下载“原子哥”APP

文档更新说明

版本	版本更新说明	负责人	校审	发布日期
V1.0	初稿:	正点原子 linux 团队	正点原子 linux 团队	2020.11.23
V1.1	1、添加 ping 失败示例。 2、添加 1.2~1.4 章节的小节汇总。 3、修改了一些笔误。	正点原子 linux 团队	正点原子 linux 团队	2020.12.01
V1.2	1、在前言中添加笔误联系方式。 2、在第二章、第三章的开头添加用户使用基础。 3、修改 4.2 小节 bootz 地址错误。	正点原子 linux 团队	正点原子 linux 团队	2020.12.28
V1.3	1、修改 2.2.3 小节的/etc/xinetd.d/tftp 文件表述 2、文档 4.3 小节最后添加 NFS 常见问题的解决思路。 3、添加第五章中虚拟网卡不存在的参考解决思路。 4、优化 TFTP、NFS 挂载指令排版	正点原子 linux 团队	正点原子 linux 团队	2021.3.31
V1.3.1	1、修复 1.4.3 小节 DNS 问题 2、修改/etc/xinetd.d/tftp 与驱动开发指南同步。 3、添加双网卡虚拟机无法上网问题。	正点原子 linux 团队	正点原子 linux 团队	2021.6.21

目录

前言	6
第一章 I.MX6U 开发板网络环境搭建	7
1.1 环境分析	7
1.1.1 开发工具	7
1.1.2 网络拓扑结构	7
1.1.3 同一网段	7
1.1.4 ping 测试概念	7
1.1.5 关闭 Ubuntu 和 Windows 防火墙	8
1.2 电脑和开发板直连同个路由器	10
1.2.1 准备工作	10
1.2.2 VMware 设置	10
1.2.3 Ubuntu 设置	11
1.2.4 Windows 设置	13
1.2.5 开发板设置	14
1.2.6 ping 测试	14
1.2.7 小结	17
1.3 电脑和开发板直连同个交换机	17
1.3.1 准备工作	17
1.3.2 VMware 设置	18
1.3.3 Ubuntu 设置	19
1.3.4 Windows 设置	21
1.3.5 开发板设置	22
1.3.6 ping 测试	22
1.3.7 小结	24
1.4 电脑 WiFi 上网, 开发板和电脑直连	25
1.4.1 准备工作	25
1.4.2 VMware 设置	25
1.4.3 Ubuntu 设置	33
1.4.4 Windows 设置	40
1.4.5 开发板设置	42
1.4.6 ping 测试	43
1.4.7 小结	44
1.5 电脑只有无线网卡	45
1.5.1 准备工作	45
1.5.2 VMware 设置	46
第二章 搭建 TFTP 环境	54
2.1 TFTP 简介	54
2.2 搭建 TFTP	54
2.2.1 安装和配置 xinetd	54
2.2.2 TFTP 目录	55
2.2.3 tftp-hpa 和 tftpd-hpa 服务程序	55

2.2.4 网络环境.....	57
2.2.5 TFTP 测试	57
第三章 搭建 NFS 环境	59
3.1 NFS 简介	59
3.2 安装 NFS 服务	59
3.3 NFS 共享目录	59
3.4 配置 NFS 服务	60
3.5 网络环境.....	61
3.6 NFS 测试	61
第四章 挂载教程系统.....	63
4.1 准备工作.....	63
4.1.1 网络环境.....	63
4.1.2 内核文件准备.....	63
4.1.3 文件系统准备.....	63
4.2 TFTP 挂载内核、设备树	64
4.3 NFS 挂载文件系统	65
第五章 附录	68
5.1 VMware 虚拟网卡问题	68
5.1.1 不存在虚拟网卡 VMnet1 和 VMnet8	68
5.2 双网卡虚拟机无法上网.....	68
5.2.1 只保留 NAT 网卡就可以上网.....	68
5.2.2 只保留 NAT 网卡也无法上网.....	69

前言

本文档使用的环境是 VMware Workstation Pro 15.5 虚拟机软件和 Ubuntu16.04, 主机系统是 Windows10 系统。开发板所使用的是正点原子阿尔法开发板和 MINI 开发板。若环境不一样可能会有所差别, 请酌情参考。

同理, 每个人的网络拓扑情况也是不一样, 这份文档列举了大部分用户的网络拓扑, 请通读一遍 [1.1 小节](#)后, 再根据自身环境情况选择对应的小节进行搭建:

开发板和电脑直连同个路由器的请看 1.2 小节;

开发板和电脑直连同个交换机的请看 1.3 小节;

电脑 WiFi 上网、开发板直连电脑的请看 1.4 小节;

电脑没有网口的请看 1.5 小节。

编者水平有限, 如果发现文档中有笔误或者歧义的地方, 恳请 QQ 联系编者 2765961790 进行更正。

第一章 I.MX6U 开发板网络环境搭建

1.1 环境分析

正点原子粉丝众多, 每个人所用的开发工具和网络环境都不太一样, 这里列举一些常见的网络环境供大家参考。

注意, 一些保密性较好的学校、机构、公司的网络环境因不可抗力因素是可能无法按照文档搭建完成的。

1.1.1 开发工具

首先看下我们所用到基础设备及工具: 电脑、开发板、直通网线。

这三个是本文档最基本的工具, 这里的电脑指的是我们用的 Windows 系统的电脑主机和 Ubuntu 系统的虚拟机、Ubuntu 系统的电脑主机, 苹果电脑这里不做说明。

除了这三个基础设备外, 有些用户可能还会用到路由器、交换机、USB 转网口拓展线、USB 转 WIFI 接口等工具, 遇到具体的情况具体分析。

路由器和交换机还是有所区别的。如果是使用交换机, 则接入网线生成的 IP 是和原宽带的 IP 是在同一网段的。如果使用的是路由器, 宽带接入 WLAN 口, 开发板网线和电脑网线接在 LAN 口, 则开发板网段和电脑网段处于同一网段, 但和宽带不在同一网段。也就是路由器会生成一个新的网段供开发板和电脑使用。

具体工具的使用在以下对应的网络拓扑中。

1.1.2 网络拓扑结构

根据百度百科的记录, 网络拓扑(Network Topology)结构是指用传输介质互连各种设备的物理布局。指构成网络的成员间特定的物理的即真实的、或者逻辑的即虚拟的排列方式。本文档的理解就是开发板与电脑的局域网组合形式。

各种设备指的是电脑主机、虚拟机、开发板、路由器/交换机等。

传输介质指的是直通网线、WIFI 等。

1.1.3 同一网段

同一网段: 本文档所用的是 C 类的 IP, IP 地址范围为 192.0.0.0-223.255.255.255。C 类 IP 前三个数表示不同的网络, 最后一个数表示同个网段中的不同主机。例如 192.168.10.0 和 192.168.18.0 是不同的网络(例如局域网和无线网络), 192.168.10.50 和 192.168.10.100 是在同一网段(都在局域网)。只有在同一网段中的主机才能 ping 通。

1.1.4 ping 测试概念

ping: ping 是一种网络诊断工具, 格式为:

ping IP 地址

以下是本文档常用的三种 ping 测试成功示例, 具体使用要结合后面的网络环境搭建。

```
C:\Users\ALIEN TEK>ping 192.168.3.7

正在 Ping 192.168.3.7 具有 32 字节的数据:
来自 192.168.3.7 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.3.7 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.3.7 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.3.7 的回复: 字节=32 时间<1ms TTL=64

192.168.3.7 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 0ms, 最长 = 0ms, 平均 = 0ms
```

图 1.1.4-1 Windows 命令终端 ping 示例

```
alientek@ubuntu16:~$ ping 192.168.3.9
PING 192.168.3.9 (192.168.3.9) 56(84) bytes of data.
64 bytes from 192.168.3.9: icmp_seq=1 ttl=64 time=0.660 ms
64 bytes from 192.168.3.9: icmp_seq=2 ttl=64 time=0.557 ms
64 bytes from 192.168.3.9: icmp_seq=3 ttl=64 time=0.633 ms
```

图 1.1.4-2 Ubuntu 终端 ping 示例

```
alientek@ubuntu16:~$ ping www.baidu.com
PING www.baidu.com (14.215.177.38) 56(84) bytes of data.
64 bytes from 14.215.177.38: icmp_seq=1 ttl=54 time=6.58 ms
64 bytes from 14.215.177.38: icmp_seq=2 ttl=54 time=7.09 ms
64 bytes from 14.215.177.38: icmp_seq=3 ttl=54 time=6.85 ms
64 bytes from 14.215.177.38: icmp_seq=4 ttl=54 time=8.83 ms
64 bytes from 14.215.177.38: icmp_seq=5 ttl=54 time=7.41 ms
```

图 1.1.4-3 ping 网站示例

以下是常见的 ping 失败示例。

```
alientek@ubuntu16:~$ ping 192.168.10.50
PING 192.168.10.50 (192.168.10.50) 56(84) bytes of data.
From 192.168.10.100 icmp_seq=1 Destination Host Unreachable
From 192.168.10.100 icmp_seq=2 Destination Host Unreachable
From 192.168.10.100 icmp_seq=3 Destination Host Unreachable
```

图 1.1.4-4 Destination Host Unreachable

这种一般是没有设置 IP 在同一网段。其他的 ping 失败可以结合反馈信息，参考下面的博客链接分析原因。

ping 失败的结果分析博客地址: <https://www.cnblogs.com/pinganzi/p/6812281.html>

执行 `ctrl + Z` 即可停止测试。如果不是以上示例的反馈信息，则 ping 测试失败，需要检查网络或防火墙。

1.1.5 关闭 Ubuntu 和 Windows 防火墙

在 Ubuntu 和 Windows 互相通信之前，需要关闭防火墙。在 Windows 主机的控制面板\系统和安全\Windows Defender 防火墙中选择关闭防火墙。



图 1.1.5-1 打开防火墙设置

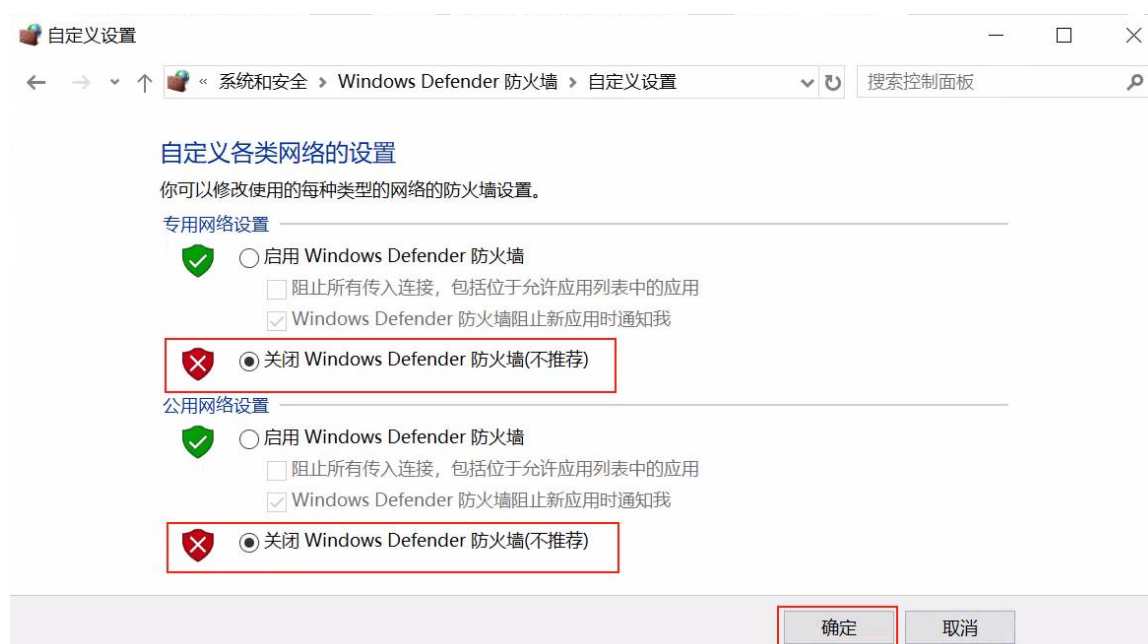


图 1.1.5-2 关掉 Windows 防火墙

打开 Ubuntu 终端, 使用以下指令关闭 Ubuntu 防火墙。

```
sudo ufw disable
```

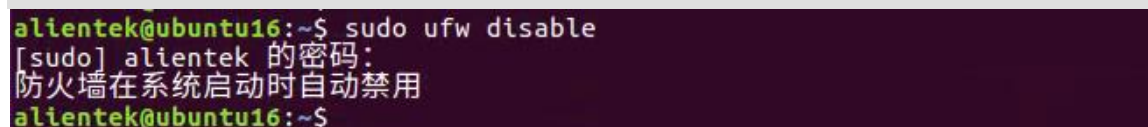


图 1.1.5-3 关闭 Ubuntu 防火墙

至此, 我们对网络的一些概念和操作有了一点了解, 现在就请根据自身网络拓扑情况, 选择对应的章节进行操作。

开发板和电脑直连同个路由器的请看 [1.2 小节](#);

开发板和电脑直连同个交换机的请看 [1.3 小节](#);

电脑 WiFi 上网、开发板直连电脑的请看 [1.4 小节](#);

电脑没有网口的请看 [1.5 小节](#)。

1.2 电脑和开发板直连同个路由器

1.2.1 准备工作

使用场景: 电脑和路由器/交换机比较近, 有两条网线。这种容易搭建。

设备: 电脑 (Windows)、正点原子 Linux 开发板 (出厂系统)、路由器、直通网线 (2 条)。

网络拓扑结构: 电脑网口通过直通网线连接到路由器/交换机的 LAN 口, 开发板网口通过直通网线连接到路由器/交换机的 LAN 口, 路由器/交换机的 LAN 口接到外部宽带, 从而实现上网。

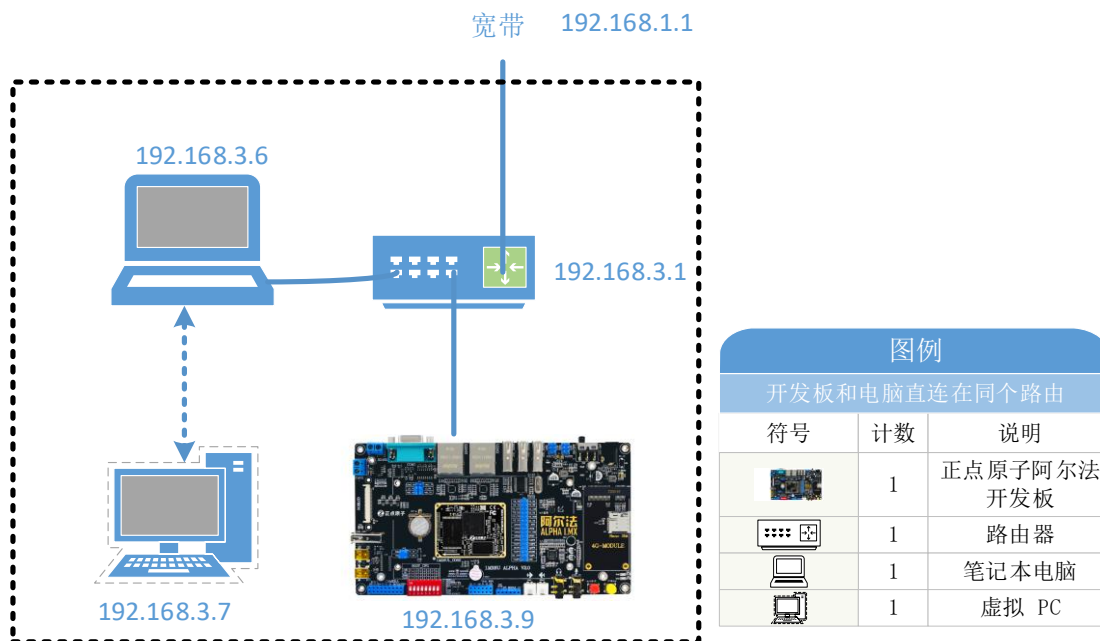


图 1.2.1-1 笔记本和开发板接在同个路由器

1.2.2 VMware 设置

Ubuntu 设置: 打开 VMware Workstation Pro 的 虚拟机 -> 设置

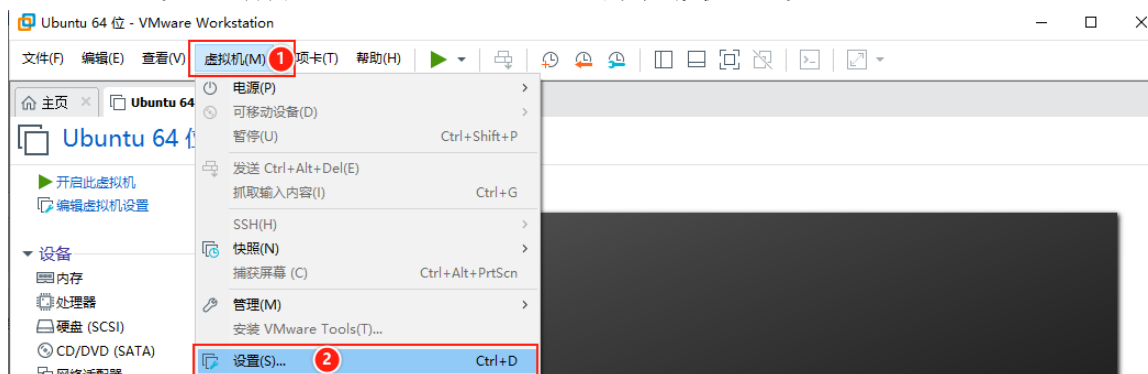


图 1.2.2-1 打开虚拟机设置

在虚拟机设置中, 将 网络适配器 修改为 桥接模式, 点击确定。

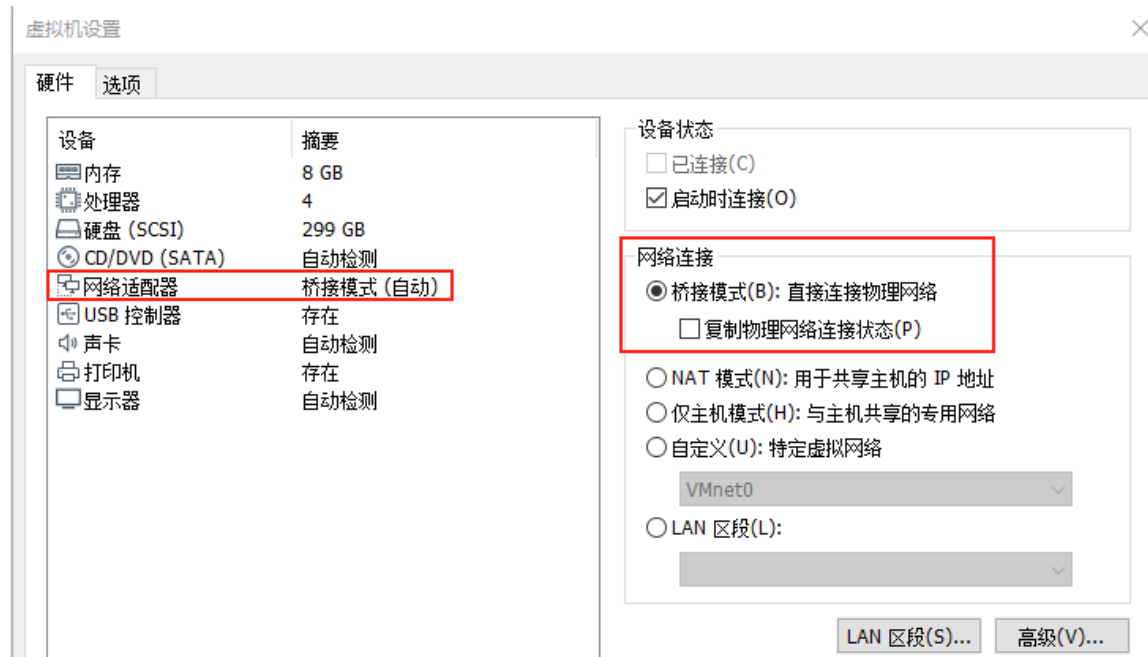


图 1.2.2-2 将网络适配器修改为桥接模式

1.2.3 Ubuntu 设置

打开 Ubuntu16 系统, 点击任务栏的设置 -> 网络。

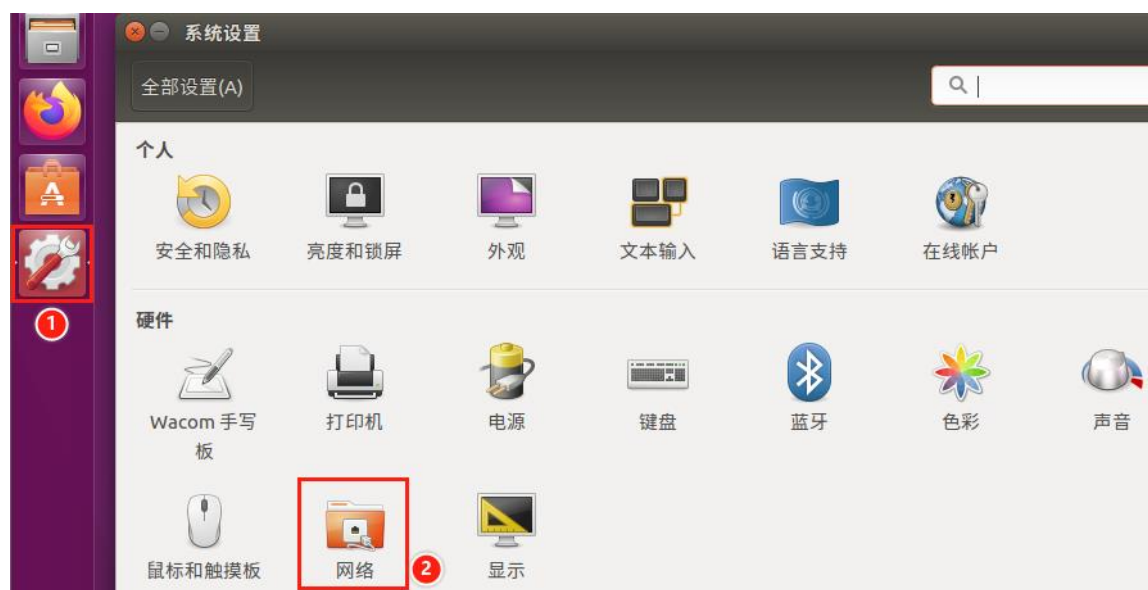


图 1.2.3-1 打开 Ubuntu 网络设置

在 Ubuntu 的网络设置这里可以看到 Ubuntu 系统的 IP 和网关、DNS 等信息。如果没有的话就打开右下角的 选项 按钮。



图 1.2.3-2 网络信息

打开 选项 按键后,就打开了编译网络设置的界面,在这里可以选择 IPv4 设置,选择使用自动 (DHCP) 的方式来获取路由器分配的 IP, 设置完成点击右下角的保存。

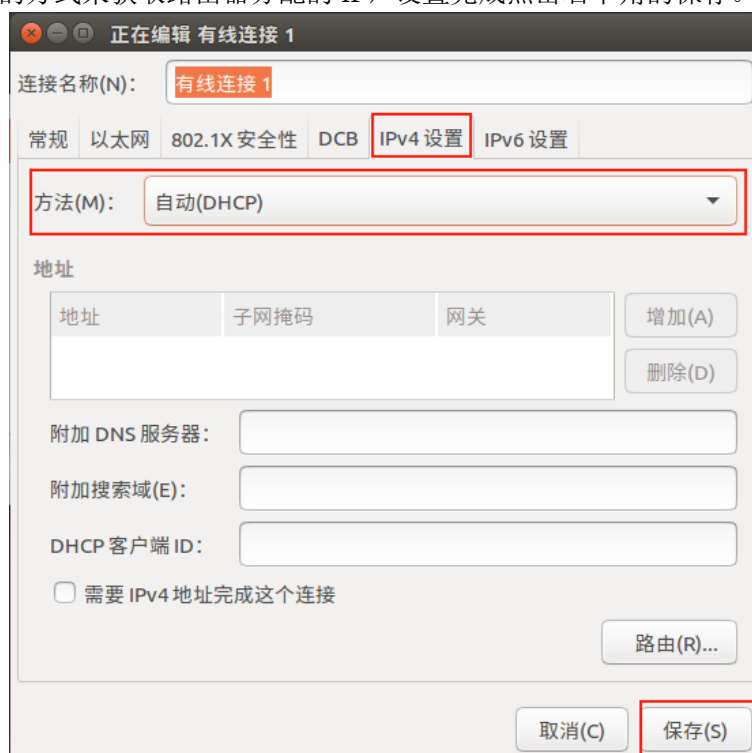


图 1.2.3-3 设置自动获取 IP

设置完成后,并不会更新网络信息,会返回到刚刚的 网络 界面,点击下图中的开关按键,关闭网络后重新打开,就会更新网络信息。如果不进行此操作就不会更新设置。



图 1.2.3-4 重新打开网络更新设置

1.2.4 Windows 设置

查看 Windows 主机的 IP: 在 Windows 端按下 win + R 快捷键打开运行栏, 输入 cmd 指令后按下回车。

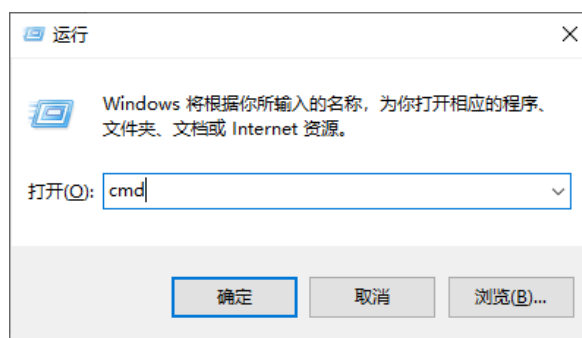


图 1.2.4-1 执行 cmd 指令

在打开的终端中输入 ipconfig 指令, 查看电脑的 IP。

```
ipconfig
```

```
CA: 选择C:\WINDOWS\system32\cmd.exe
Microsoft Windows [版本 10.0.19041.572]
(c) 2020 Microsoft Corporation. 保留所有权利。

C:\Users\ALIENTEK>ipconfig

Windows IP 配置

以太网适配器 以太网:

    连接特定的 DNS 后缀 . . . . . :
    本地链接 IPv6 地址. . . . . : fe80::69c0:3c0:45c2:5d19%33
    IPv4 地址 . . . . . : 192.168.3.6
    子网掩码 . . . . . : 255.255.255.0
    默认网关. . . . . : 192.168.3.1
```

图 1.2.4-2 查看电脑主机 IP

1.2.5 开发板设置

开发板设置: 直通网线的一端接正点原子 Linux 开发板的网口, 如果是阿尔法开发板就接 ENET2, 另一端接路由器的 LAN 网口。在串口终端输入 ifconfig 指令来查询开发板是否获取到路由器分配到的 IP。

```
ifconfig

root@ATK-IMX6U:~# ifconfig
eth0    Link encap:Ethernet  HWaddr 22:17:93:36:76:f1
        inet addr:192.168.3.9 Bcast:192.168.3.255 Mask:255.255.255.0
        inet6 addr: fe80::2017:93ff:fe36:76f1/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:185 errors:0 dropped:0 overruns:0 frame:0
        TX packets:64 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:20432 (19.9 KiB)  TX bytes:11855 (11.5 KiB)
```

图 1.2.5-1 开发板获取到 IP

1.2.6 ping 测试

在 ping 测试之前要关闭 Windows 和 Ubuntu 的防火墙。

综合前面的设置, 这里笔者的 IP 如下。

Windows 主机 IP: 192.168.3.6

Ubuntu 虚拟机 IP: 192.168.3.7

开发板 IP: 192.168.3.9

测试开发板和 Ubuntu、Windows 互 ping。

开发板 ping 虚拟机 Ubuntu 的 IP:

ping 192.168.3.7

```
root@ATK-IMX6U:~# ping 192.168.3.7
PING 192.168.3.7 (192.168.3.7) 56(84) bytes of data.
64 bytes from 192.168.3.7: icmp_seq=1 ttl=64 time=1.11 ms
64 bytes from 192.168.3.7: icmp_seq=2 ttl=64 time=0.610 ms
64 bytes from 192.168.3.7: icmp_seq=3 ttl=64 time=0.587 ms
```

图 1.2.6-1 开发板 ping 虚拟机

验证 ping 通后, 按下 `ctrl + Z` 快捷键即可停止 ping。以下停止 ping 测试的方法同理。

Ubuntu 虚拟机 ping 开发板 IP

ping 192.168.3.9

```
alientek@ubuntu16:~$ ping 192.168.3.9
PING 192.168.3.9 (192.168.3.9) 56(84) bytes of data.
64 bytes from 192.168.3.9: icmp_seq=1 ttl=64 time=0.660 ms
64 bytes from 192.168.3.9: icmp_seq=2 ttl=64 time=0.557 ms
64 bytes from 192.168.3.9: icmp_seq=3 ttl=64 time=0.633 ms
```

图 1.2.6-2 虚拟机 ping 开发板

Ubuntu 虚拟机 ping 外网。

ping www.baidu.com

```
alientek@ubuntu16:~$ ping www.baidu.com
PING www.baidu.com (14.215.177.38) 56(84) bytes of data.
64 bytes from 14.215.177.38: icmp_seq=1 ttl=54 time=6.58 ms
64 bytes from 14.215.177.38: icmp_seq=2 ttl=54 time=7.09 ms
64 bytes from 14.215.177.38: icmp_seq=3 ttl=54 time=6.85 ms
64 bytes from 14.215.177.38: icmp_seq=4 ttl=54 time=8.83 ms
64 bytes from 14.215.177.38: icmp_seq=5 ttl=54 time=7.41 ms
```

图 1.2.6-3 虚拟机 ping 外网

开发板 ping 外网。

ping www.baidu.com

```
root@ATK-IMX6U:~# ping www.baidu.com
PING www.baidu.com (14.215.177.38) 56(84) bytes of data.
64 bytes from 14.215.177.38: icmp_seq=1 ttl=54 time=6.51 ms
64 bytes from 14.215.177.38: icmp_seq=2 ttl=54 time=6.13 ms
64 bytes from 14.215.177.38: icmp_seq=3 ttl=54 time=6.39 ms
64 bytes from 14.215.177.38: icmp_seq=4 ttl=54 time=7.07 ms
^C64 bytes from 14.215.177.38: icmp_seq=5 ttl=54 time=6.09 ms
```

图 1.2.6-4 开发板 ping 外网

Windows 和 Ubuntu 虚拟机互 ping。

Windows 主机 ping 虚拟机 Ubuntu。

ping 192.168.3.7

```
C:\Users\ALIEN TEK>ping 192.168.3.7

正在 Ping 192.168.3.7 具有 32 字节的数据:
来自 192.168.3.7 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.3.7 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.3.7 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.3.7 的回复: 字节=32 时间<1ms TTL=64

192.168.3.7 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 0ms, 最长 = 0ms, 平均 = 0ms
```

图 1.2.6-5 Windows 主机 ping 虚拟机 ubuntu

Ubuntu 虚拟机 ping 主机 Windows。

在 Ubuntu 终端执行以下指令 ping 主机 Windows 的 IP。

```
alien tek@ubuntu16:~$ ping 192.168.3.6
PING 192.168.3.6 (192.168.3.6) 56(84) bytes of data.
64 bytes from 192.168.3.6: icmp_seq=1 ttl=64 time=0.256 ms
64 bytes from 192.168.3.6: icmp_seq=2 ttl=64 time=0.234 ms
64 bytes from 192.168.3.6: icmp_seq=3 ttl=64 time=0.331 ms
^C
--- 192.168.3.6 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2034ms
rtt min/avg/max/mdev = 0.234/0.273/0.331/0.045 ms
```

图 1.2.6-6 Ubuntu 虚拟机 ping 主机 Windows

以上就是开发板、Ubuntu 和 Windows 三者互 ping 测试，在开发板 **uboot 命令行** 设置完开发板 IP 相关的信息后，也能 ping 虚拟机 Ubuntu。如果了解 uboot 命令行和网络操作指令，请先看《【正点原子】I.MX6U 嵌入式 Linux 驱动开发指南》的 30.3 小节和 30.4.4 小节。

```
setenv ipaddr 192.168.3.9
setenv ethaddr 00:04:9f:04:d2:35
setenv gatewayip 192.168.3.1
setenv netmask 255.255.255.0
setenv serverip 192.168.3.7
saveenv
ping 192.168.3.7
```

```
Hit any key to stop autoboot: 0
=>
=>
=>
=> setenv ipaddr 192.168.3.9
=> setenv ethaddr 00:04:9f:04:d2:35
=> setenv gatewayip 192.168.3.1
=> setenv netmask 255.255.255.0
=> setenv serverip 192.168.3.7
=> saveenv
Saving Environment to MMC...
Writing to MMC(1)... done
=> ping 192.168.3.7
FEC1 Waiting for PHY auto negotiation to complete... done
Using FEC1 device
host 192.168.3.7 is alive
```

图 1.2.6-7 开发板 uboot 阶段 ping 虚拟机

uboot 阶段时, Ubuntu 虚拟机 ping 开发板是没有反馈的, 所以不用进行这一步。按照上述完成测试后, 就可以使用 TFTP 和 NFS 来挂载内核、设备树和文件系统。搭建 TFTP 请看[第二章](#)的内容, 搭建 NFS 请看第三章的内容。

1.2.7 小结

防火墙设置: 关闭电脑和虚拟机的防火墙。

网络连接: 开发板网口和电脑网口都通过网线直接连到同个路由器。

VMware 设置: 设置网络适配器桥接模式, 虚拟网络编辑器还原默认设置。

Ubuntu 设置: 设置网络连接模式为自动 (DHCP)。

Windows 设置: 设置以太网 TCP/IPv4 属性为自动获取 IP 地址。

开发板设置: 出厂系统默认会自动获取 IP, 没有这个功能的话可以执行以下指令获取 IP。

```
ifconfig eth0 up
```

```
udhcpc -i eth0
```

1.3 电脑和开发板直连同个交换机

1.3.1 准备工作

使用场景: 宽带可以直连交换机, 有两条网线, 没有路由器, 没有 WiFi 或者无线网卡。

设备: 电脑 (Windows)、正点原子 Linux 开发板 (出厂系统)、交换机、直通网线 (2 条)。

网络拓扑结构:

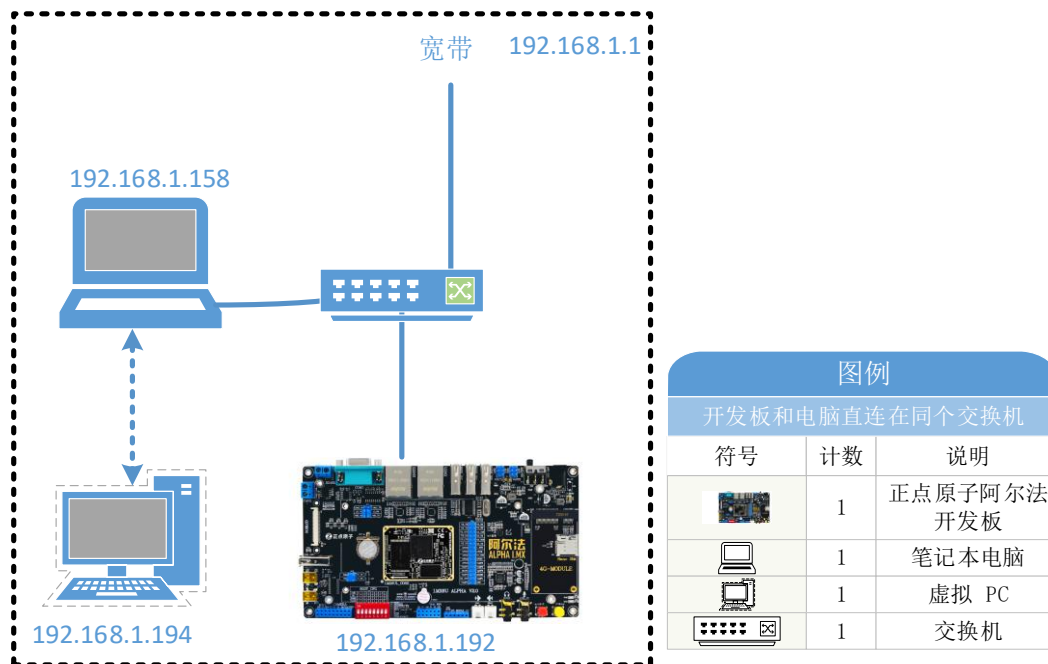


图 1.3.1-1 开发板和电脑直连同个交换机

直连路由器和直连交换机的效果是差不多的，只不过路由器会一个单独的网段，而交换机的网段和宽带分配的网段是一样的。也就是一个是以路由器为中心的局域网，一个是以宽带为中心的局域网。

1.3.2 VMware 设置

打开 VMware Workstation Pro 的 虚拟机 -> 设置

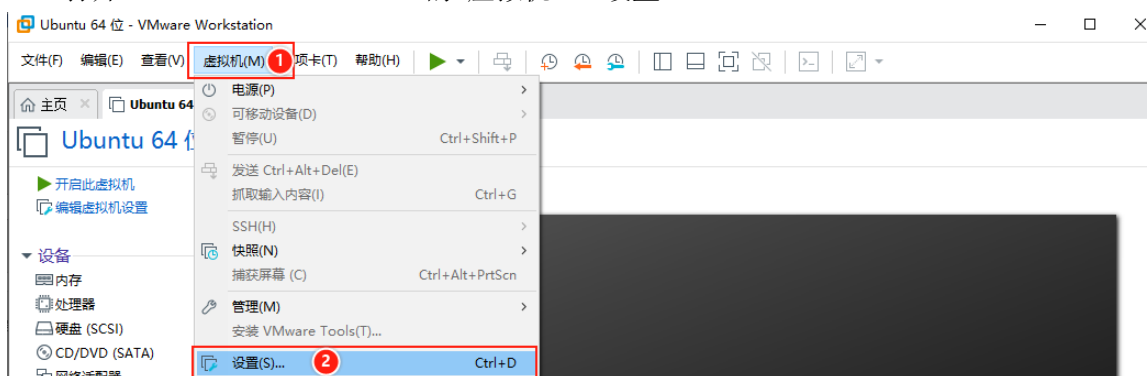


图 1.3.2-1 打开虚拟机设置

在虚拟机设置中，将 网络适配器 修改为 桥接模式，点击确定。

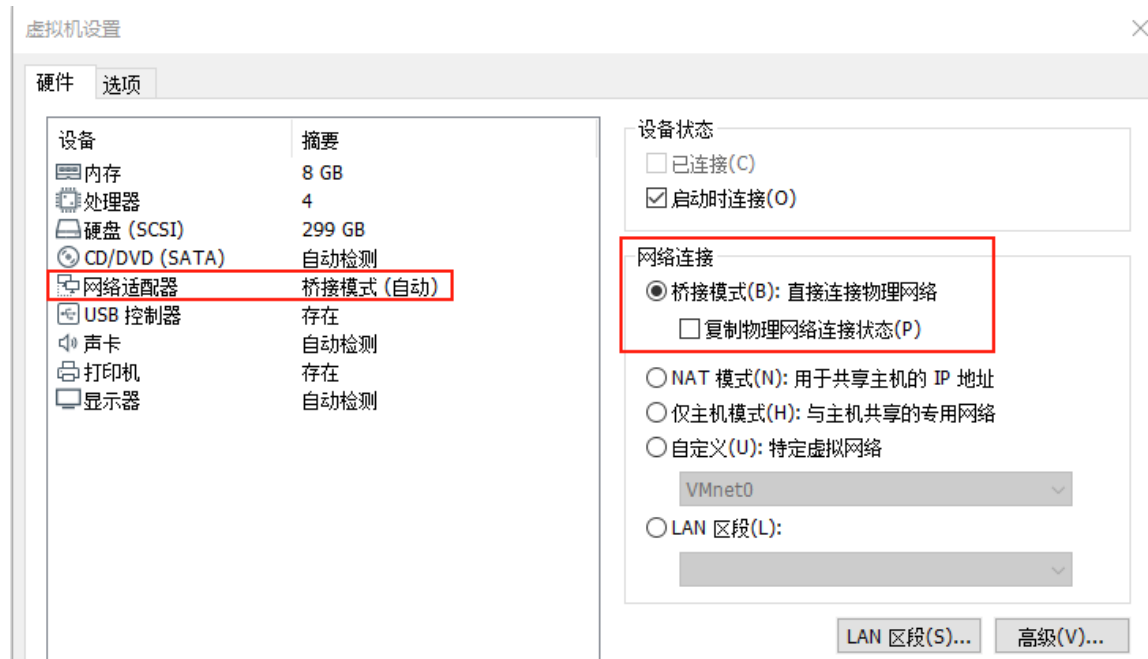


图 1.3.2-2 将网络适配器修改为桥接模式

1.3.3 Ubuntu 设置

打开 Ubuntu16 系统, 点击任务栏的设置 -> 网络。

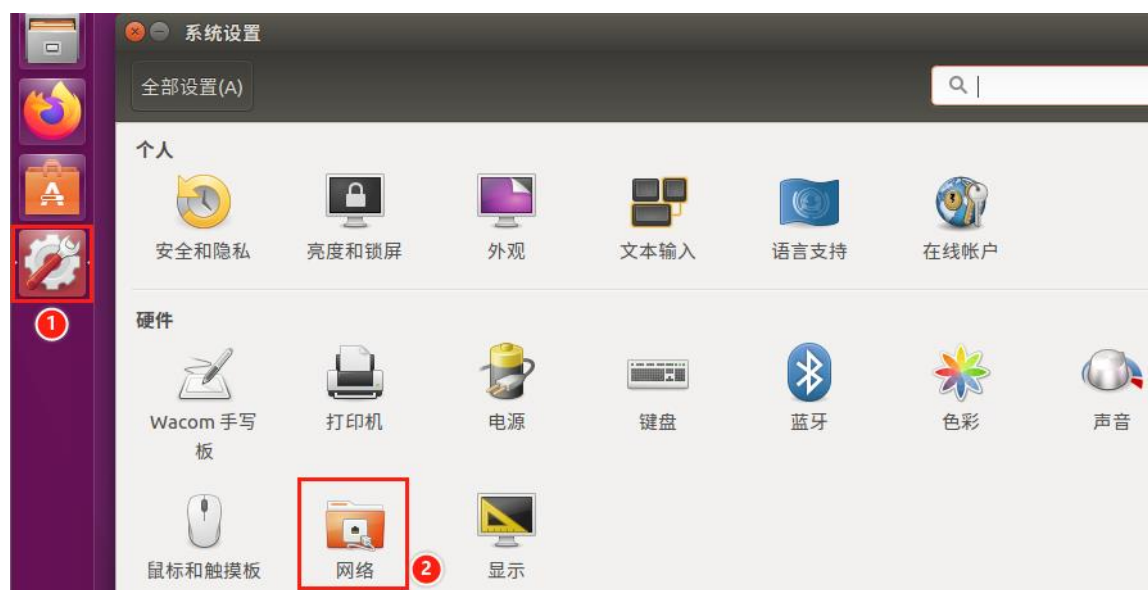


图 1.3.3-1 打开 Ubuntu 网络设置

在 Ubuntu 的网络设置这里可以看到 Ubuntu 系统的 IP 和网关、DNS 等信息。如果没有的话就打开右下角的 选项 按钮。



图 1.3.3-2 网络信息

打开 选项 按键后,就打开了编译网络设置的界面,在这里可以选择 IPv4 设置,选择使用自动 (DHCP) 的方式来获取路由器分配的 IP,设置完成点击右下角的保存。

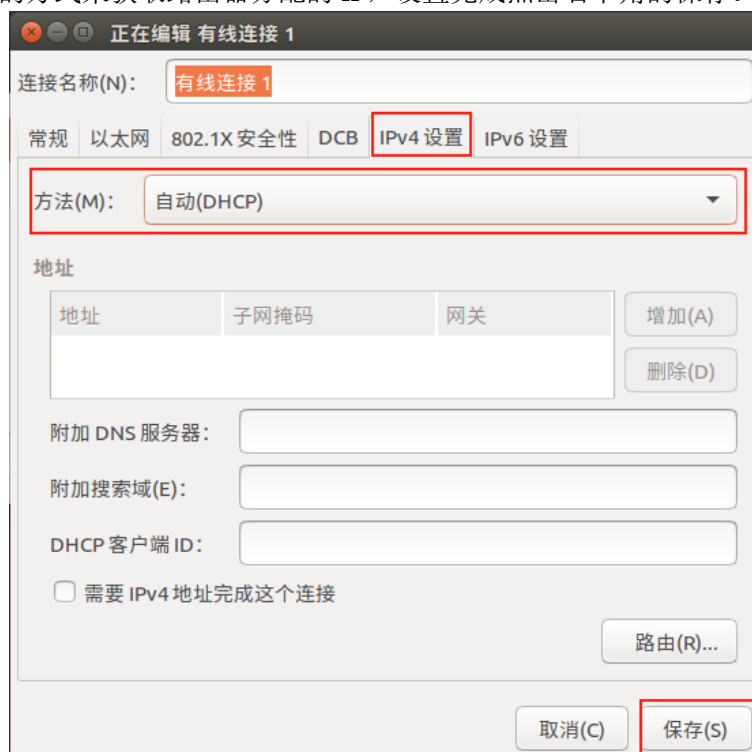


图 1.3.3-3 设置自动获取 IP

设置完成后,并不会更新网络信息,会返回到刚刚的 网络 界面,点击下图中的开关按键,关闭网络后重新打开,就会更新网络信息。如果不进行此操作就不会更新设置。



图 1.3.3-4 重新打开网络更新设置

1.3.4 Windows 设置

查看 Windows 主机的 IP: 在 Windows 端按下 win + R 快捷键打开运行栏, 输入 cmd 指令后按下回车。

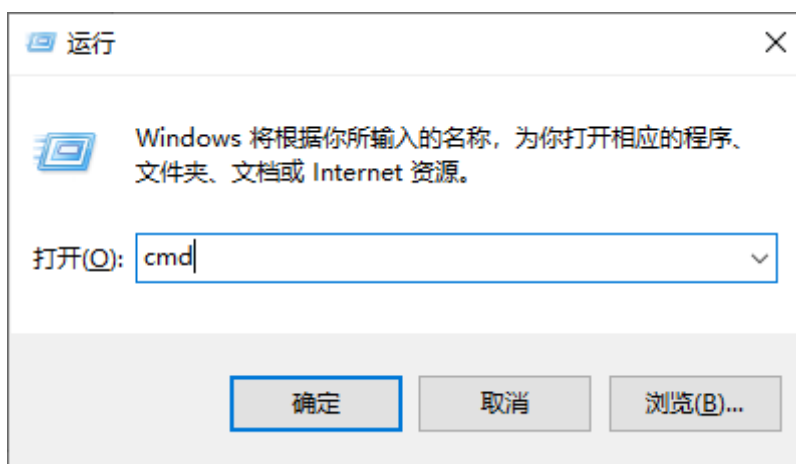


图 1.3.4-1 执行 cmd 指令

在打开的终端中输入 ipconfig 指令, 查看电脑的 IP。

```
ipconfig
```

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [版本 10.0.19041.572]
(c) 2020 Microsoft Corporation. 保留所有权利。

C:\Users\ALIEN TEK>ipconfig

Windows IP 配置

以太网适配器 以太网:

    连接特定的 DNS 后缀 . . . . . : 
    本地链接 IPv6 地址. . . . . : 
    IPv4 地址 . . . . . : 192.168.1.158
    子网掩码 . . . . . : 255.255.255.0
    默认网关. . . . . : 192.168.1.1
```

图 1.3.4-2 查看电脑主机 IP

1.3.5 开发板设置

开发板设置: 直通网线的一端接正点原子 Linux 开发板的网口, 如果是阿尔法开发板就接 ENET2, 另一端接电脑的以太网网口。在串口终端输入 `ifconfig` 指令来查询开发板是否获取到路由器分配到的 IP。

```
ifconfig
root@ATK-IMX6U:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 3a:b5:c1:60:b4:81
          inet addr:192.168.1.192  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::38b5:c1ff:fe60:b481/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:105 errors:0 dropped:0 overruns:0 frame:0
          TX packets:56 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:12102 (11.8 KiB)  TX bytes:9880 (9.6 KiB)
```

图 1.3.5-1 查看开发板获取到的 IP

1.3.6 ping 测试

在 ping 测试之前, 要关闭 Windows 和 Ubuntu 的防火墙。

综合上述, 这里笔者的 IP 如下。

Windows 主机 IP: 192.168.1.158

Ubuntu 虚拟机 IP: 192.168.1.194

开发板 IP: 192.168.1.192

测试开发板和 Ubuntu、Windows 互 ping。

开发板 ping 虚拟机 Ubuntu 的 IP:

```
ping 192.168.1.194
```

```
root@ATK-IMX6U:~# ping 192.168.1.194
PING 192.168.1.194 (192.168.1.194) 56(84) bytes of data.
64 bytes from 192.168.1.194: icmp_seq=1 ttl=64 time=0.604 ms
64 bytes from 192.168.1.194: icmp_seq=2 ttl=64 time=0.384 ms
64 bytes from 192.168.1.194: icmp_seq=3 ttl=64 time=0.412 ms
64 bytes from 192.168.1.194: icmp_seq=4 ttl=64 time=0.440 ms
```

图 1.3.6-1 开发板 ping 虚拟机 Ubuntu

ping 测试成功后, 按下 `ctrl + Z` 组合键即可停止。

Ubuntu 虚拟机 ping 开发板。

ping 192.168.1.192

```
alientek@ubuntu16: ~
alientek@ubuntu16:~$ ping 192.168.1.192
PING 192.168.1.192 (192.168.1.192) 56(84) bytes of data.
64 bytes from 192.168.1.192: icmp_seq=1 ttl=64 time=0.367 ms
64 bytes from 192.168.1.192: icmp_seq=2 ttl=64 time=0.358 ms
64 bytes from 192.168.1.192: icmp_seq=3 ttl=64 time=0.370 ms
64 bytes from 192.168.1.192: icmp_seq=4 ttl=64 time=0.378 ms
```

图 1.3.6-2 Ubuntu 虚拟机 ping 开发板

开发板 ping 主机 Windows。

ping 192.168.1.158

```
root@ATK-IMX6U:~# ping 192.168.1.158
PING 192.168.1.158 (192.168.1.158) 56(84) bytes of data.
64 bytes from 192.168.1.158: icmp_seq=1 ttl=64 time=0.488 ms
64 bytes from 192.168.1.158: icmp_seq=2 ttl=64 time=0.323 ms
64 bytes from 192.168.1.158: icmp_seq=3 ttl=64 time=0.290 ms
64 bytes from 192.168.1.158: icmp_seq=4 ttl=64 time=0.291 ms
```

图 1.3.6-3 开发板 ping 主机 Windows

Windows 主机 ping 开发板。

ping 192.168.1.192

```
C:\Users\ALIENTEK>ping 192.168.1.192

正在 Ping 192.168.1.192 具有 32 字节的数据:
来自 192.168.1.192 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.1.192 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.1.192 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.1.192 的回复: 字节=32 时间<1ms TTL=64

192.168.1.192 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
    往返行程的估计时间(以毫秒为单位):
        最短 = 0ms, 最长 = 0ms, 平均 = 0ms
```

图 1.3.6-4 Windows 主机 ping 开发板

Ubuntu 虚拟机 ping 主机 Windows。

ping 192.168.1.158

```
alien@ubuntu16:~$ ping 192.168.1.158
PING 192.168.1.158 (192.168.1.158) 56(84) bytes of data:
64 bytes from 192.168.1.158: icmp_seq=1 ttl=64 time=0.292 ms
64 bytes from 192.168.1.158: icmp_seq=2 ttl=64 time=0.241 ms
64 bytes from 192.168.1.158: icmp_seq=3 ttl=64 time=0.193 ms
64 bytes from 192.168.1.158: icmp_seq=4 ttl=64 time=0.220 ms
64 bytes from 192.168.1.158: icmp_seq=5 ttl=64 time=0.199 ms
```

图 1.3.6-5 Ubuntu 虚拟机 ping 主机 Windows

以上就是开发板、Ubuntu 和 Windows 三者互 ping 测试，在开发板 **uboot 命令行** 设置完开发板 IP 相关的信息后，也能 ping 虚拟机 Ubuntu。如果了解 uboot 命令行和网络操作指令，请先看《【正点原子】I.MX6U 嵌入式 Linux 驱动开发指南》的 30.3 小节和 30.4.4 小节。

```
setenv ipaddr 192.168.1.192
setenv ethaddr 00:04:9f:04:d2:35
setenv gatewayip 192.168.1.1
setenv netmask 255.255.255.0
setenv serverip 192.168.1.194
saveenv
ping 192.168.1.194
```

```
Hit any key to stop autoboot: 0
=>
=>
=> setenv ipaddr 192.168.1.192
=> setenv ethaddr 00:04:9f:04:d2:35
=> setenv gatewayip 192.168.1.1
=> setenv netmask 255.255.255.0
=> setenv serverip 192.168.1.194
=> saveenv
Saving Environment to MMC...
Writing to MMC(1)... done
=> ping 192.168.1.194
FEC1 Waiting for PHY auto negotiation to complete.... done
Using FEC1 device
host 192.168.1.194 is alive
```

图 1.3.6-6 开发板 uboot 阶段 ping 虚拟机

uboot 阶段时，Ubuntu 虚拟机 ping 开发板是没有反馈的，所以不用进行这一步。按照上述完成测试后，就可以使用 TFTP 和 NFS 来挂载内核、设备树和文件系统。搭建 TFTP 请看[第二章](#)的内容，搭建 NFS 请看[第三章](#)的内容。

1.3.7 小结

防火墙设置：关闭电脑和虚拟机的防火墙。

网络连接：开发板网口和电脑网口都通过网线直接连到同个路由器。

VMware 设置：设置网络适配器桥接模式，虚拟网络编辑器还原默认设置。

Ubuntu 设置：设置网络连接模式为自动（DHCP）。

Windows 设置：设置以太网 TCP/IPv4 属性为自动获取 IP 地址。

开发板设置：出厂系统默认会自动获取 IP，没有这个功能的话可以执行以下指令获取 IP。

```
ifconfig eth0 up
udhcpc -i eth0
```

1.4 电脑 WiFi 上网，开发板和电脑直连

1.4.1 准备工作

使用场景：路由器离电脑比较远，只有一条网线。

设备：电脑（Windows）、正点原子 Linux 开发板（出厂系统）、路由器/交换机、直通网线。如果是台式机没有无线网卡的话，可以使用正点原子的 USB 转 WIFI 模块来让电脑连接到 WiFi。

网络拓扑结构：

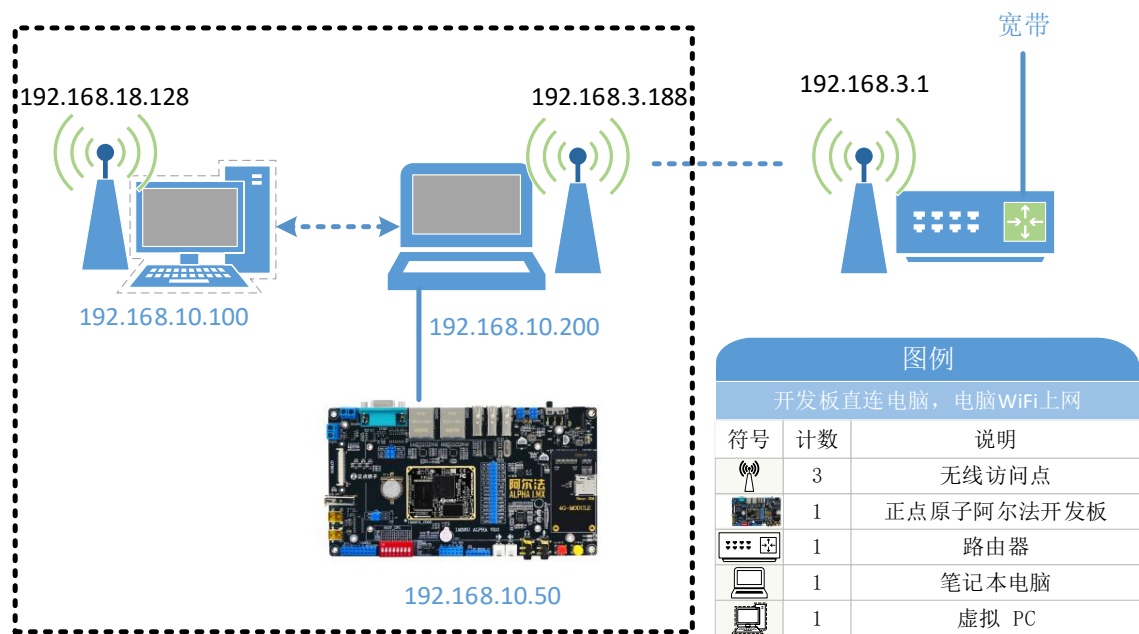


图 1.4.1-1 开发板直连电脑，电脑 WiFi 上网

1.4.2 VMware 设置

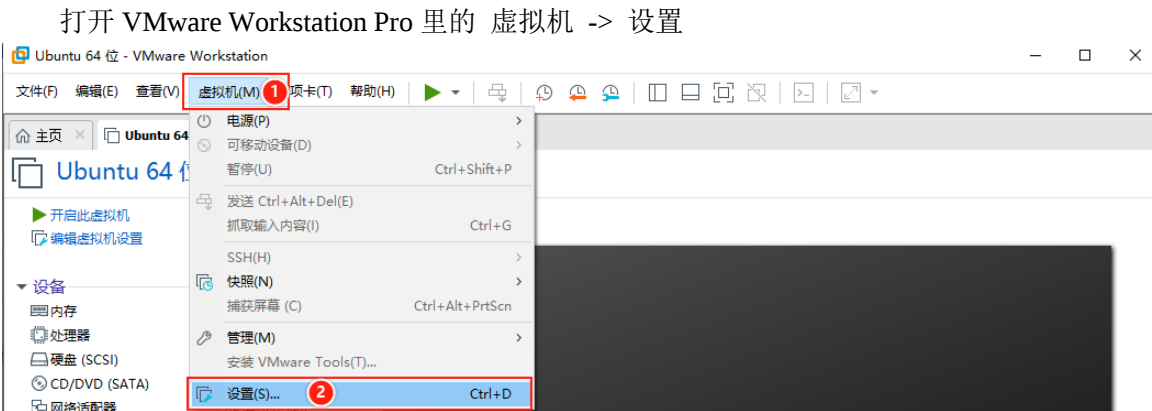


图 1.4.2-1 打开虚拟机设置

设置网络适配器为桥接模式。



图 1.4.2-2 设置桥接模式

因为电脑是 WiFi 上网,所以需要添加一个网络适配器并设置成 NAT 模式,供虚拟机上网。具体操作如下图所示。

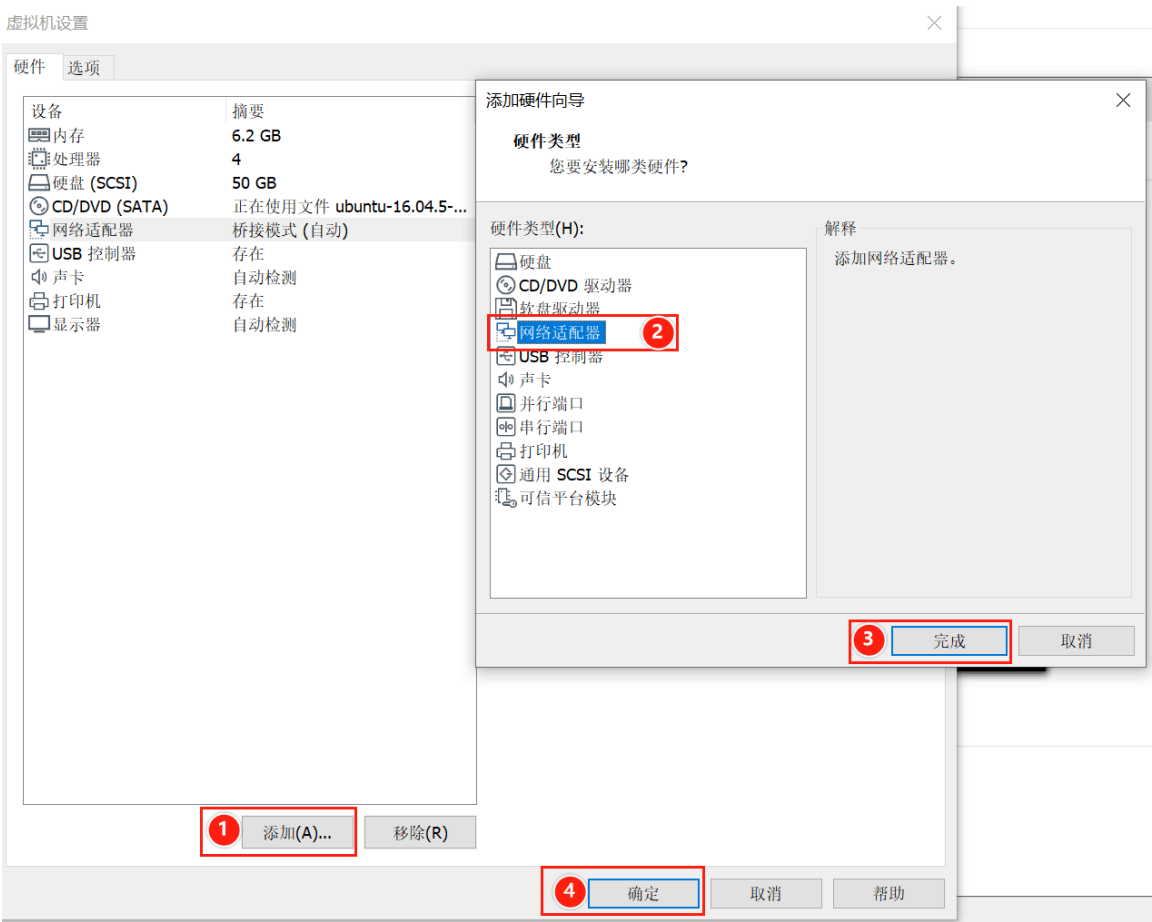


图 1.4.2-3 添加网络适配器

默认添加的网络适配器是 NAT 模式的, 如果不是 NAT 模式则要手动设置成 NAT 模式。

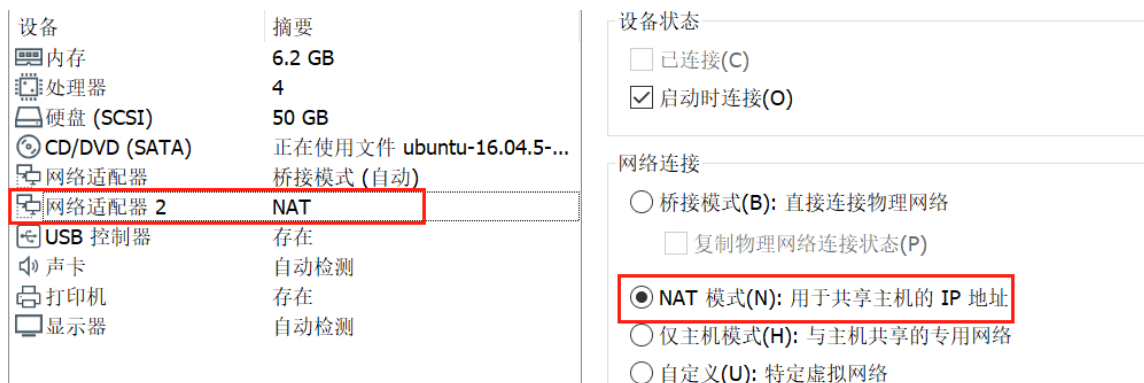


图 1.4.2-4 设置网络适配器 2 为 NAT 模式

打开菜单栏的 编辑 -> 虚拟网络编辑器。

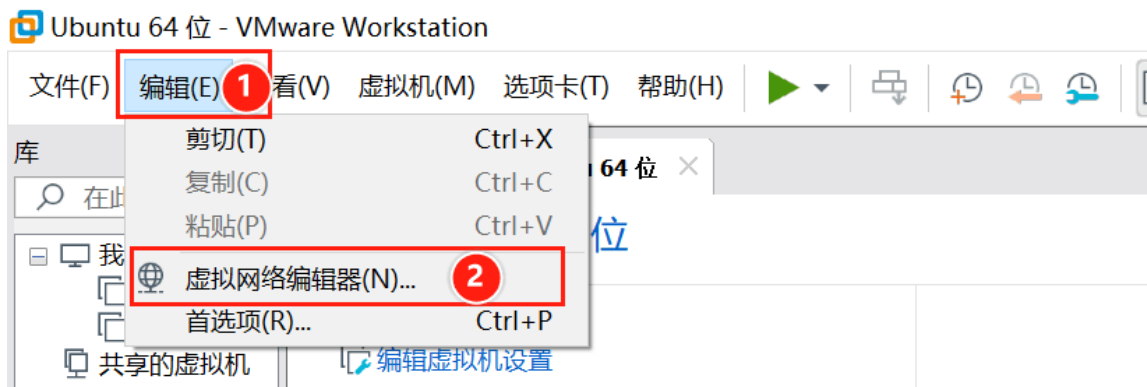


图 1.4.2-5 打开虚拟网络编辑器

点击虚拟网络编辑器的 更改设置 选项。



图 1.4.2-6 更改设置

这个需要电脑管理员权限，如果有提示用户账户控制的提示框，点击 是 就可以了。

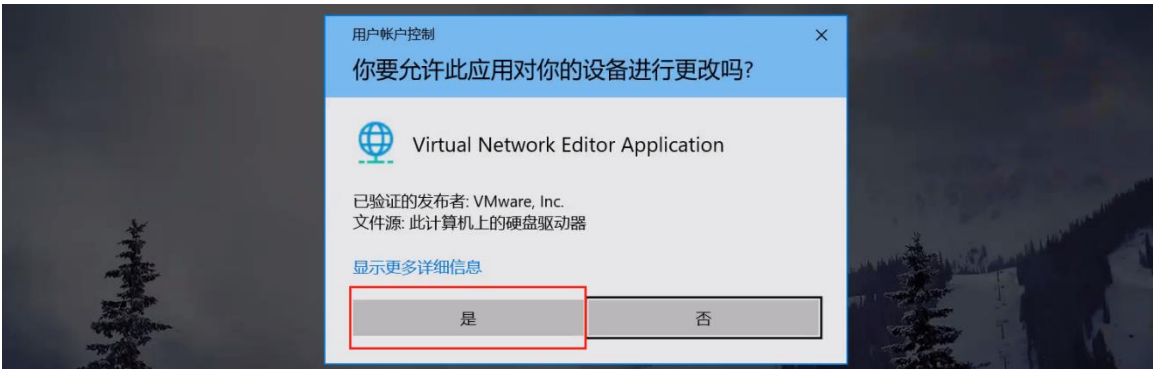


图 1.4.2-7 赋予修改权限

重新打开虚拟网络编辑器，可以看到如下界面。



图 1.4.2-8 赋予权限后的虚拟网络编辑器界面

如果此时没有 VMnet0，可以手动添加下。

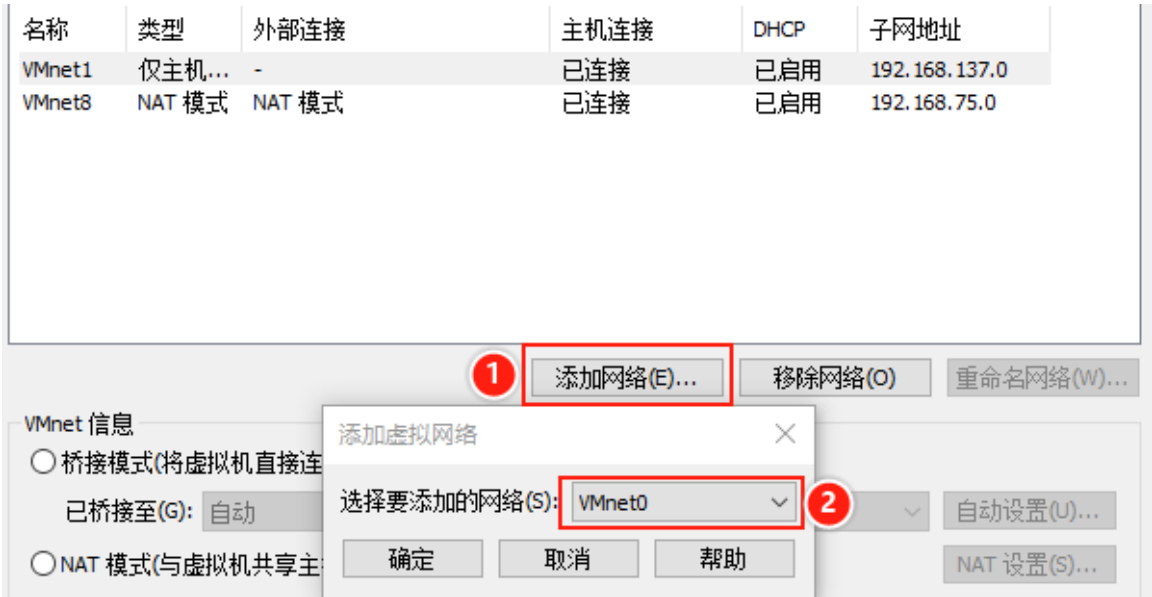


图 1.4.2-9 手动添加 VMnet0

开发板是直连到电脑的网口的,所以需要虚拟网络编辑器里的网络适配器 1 (即 VMnet0) 桥接到有线网卡上,比如文档这里是 Realtek PCIe,如下图所示。



图 1.4.2-10 查看电脑网口类型

我们需要把网络桥接到这个网卡上,以下处理的都是 VMnet0,用于桥接网络。

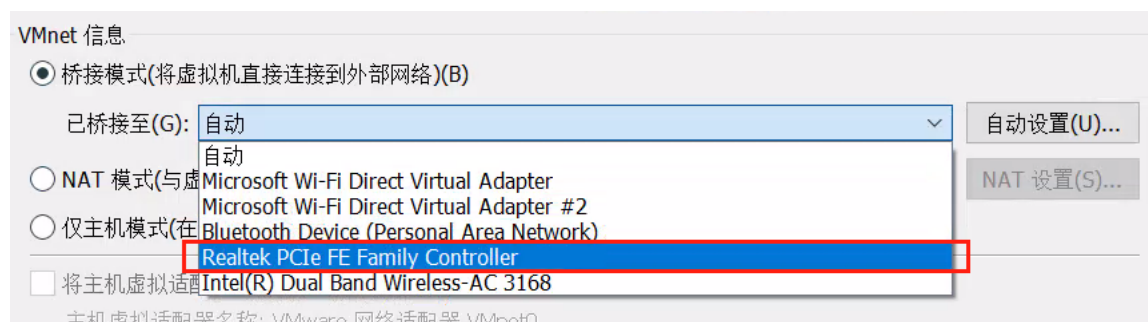


图 1.4.2-11 设置网络适配器 1 桥接到有线网卡

设置完成后点击确定,再点击应用即可。



图 1.4.2-12 完成设置

如果虚拟机之前设置过网络的相关信息,自己却搞混了,不清楚现在虚拟机网络的设置情况(虚拟网络情况复杂),可以点击虚拟网络编辑器左下方的还原默认设置,这步需要先关闭虚拟机再进行。



图 1.4.2-13 还原默认设置

这里会提示我们是否恢复到默认网络设置,点击 是 即可。

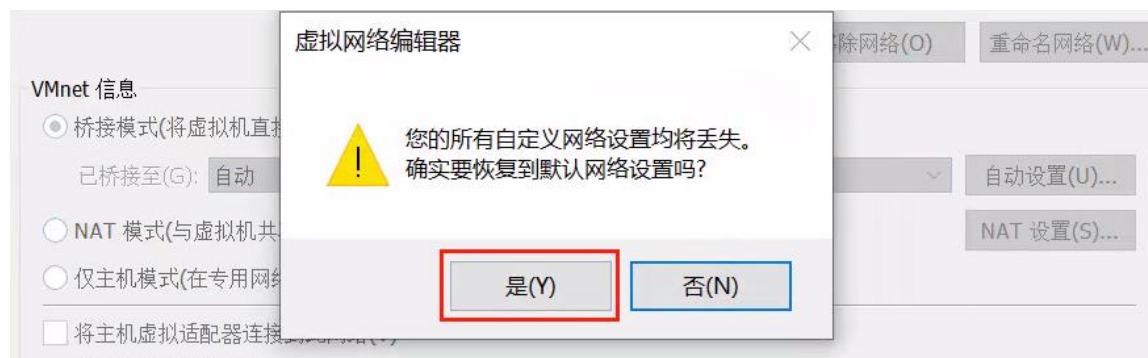


图 1.4.2-14 确认恢复到默认网络设置

恢复到默认网络设置后如下图所示,全部的 VMnet 子网地址会重新随机分配。



图 1.4.2-15 默认网络设置

这时候就可以按照之前的步骤来设置网络适配器桥接到有线网卡，最终设置成如下。



图 1.4.2-16 虚拟机桥接到有线网卡

1.4.3 Ubuntu 设置

启动虚拟机，打开 Ubuntu 的网络设置。



图 1.4.3-1 打开 Ubuntu 网络设置

在网络设置中有两个有线网络, 其中一个还没有设置 IP 的, 是我们的网络适配器 1, 用来桥接到开发板和 Windows 的。



图 1.4.3-2 还没设置的网络适配器

另外一个有线网络其实是 NAT 模式的网络适配器 2, 已经自动分配好 IP 了。



图 1.4.3-3 另一个已自动分配好的网络

注意, 此时这个 NAT 模式的网络适配器和笔者前面设置的 VMnet8 是在同一网段的 (192.168.18.X), 如果不是在同一网段, 可能是之前用户有设置过 Ubuntu 的 IP, 这里我们可以在这个网络的选项中将它设置为自动模式即可。



图 1.4.3-4 设置 NAT 网络适配器为自动获取 IP 模式

有了这个 NAT 模式的网络适配器 2, Ubuntu 就能上网且能和 Windows 互通。

先测下 Ubuntu 上网功能, ping 下百度。

ping www.baidu.com

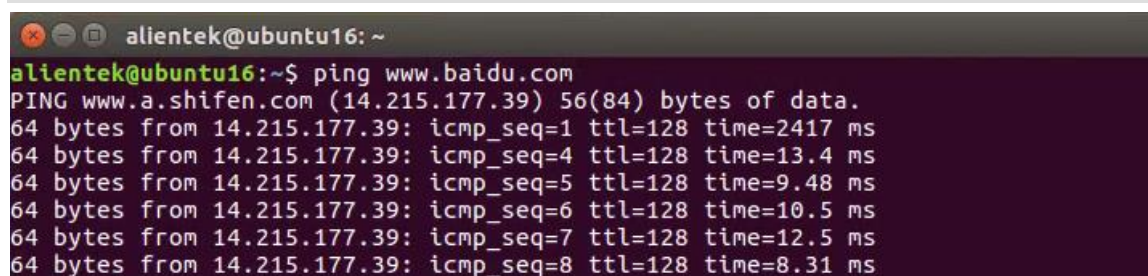


图 1.4.3-5 Ubuntu 上网测试

测试 Ubuntu 和 Windows 互传之前, 要关闭 Ubuntu 和 Windows 的防火墙。

使用 Filezilla 软件验证 Ubuntu 和 Windows 文件互传, 依次打开 文件 -> 站点管理器。

 FileZilla

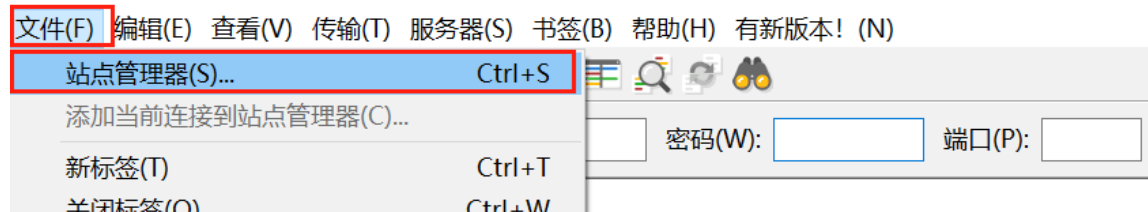


图 1.4.3-6 打开 Filezilla 软件站点管理器

根据虚拟机的信息, 设置站点, 如下图所示。

站点管理器



图 1.4.3-7 设置站点管理器

设置完后连接站点服务器，即可加载 Ubuntu 和 Windows 的目录，可以实现文件互传。

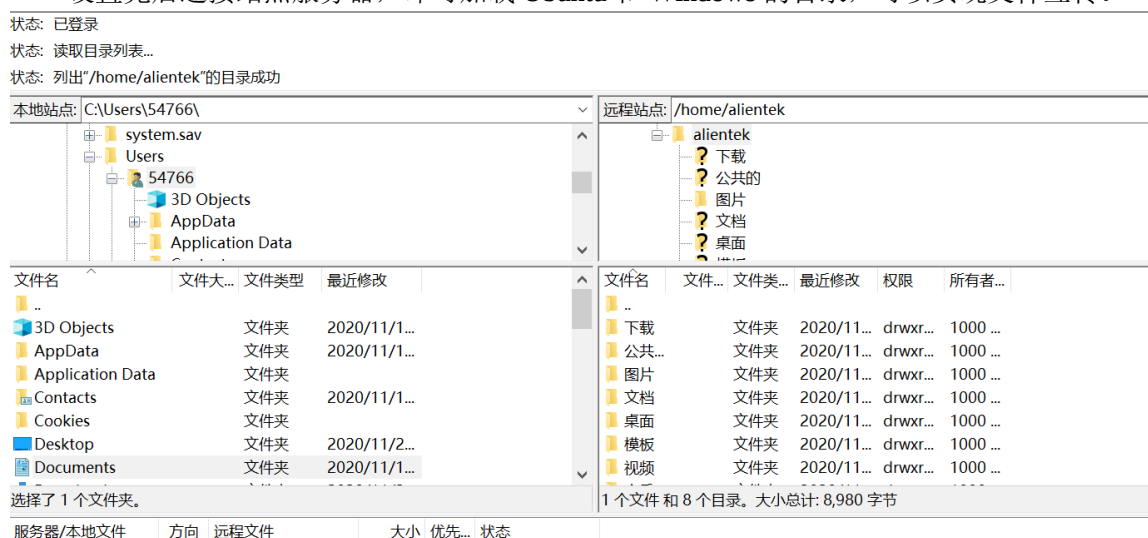


图 1.4.3-8 Ubuntu 和 Windows 目录

现在要做的就是设置桥接模式的网络适配器 1 的 IP 信息。网络适配器 1 用于和 Windows 和开发板通信，所以我们要先看下 Windows 主机的用于桥接网络的 IP。

查看 Windows 主机的 IP: 在 Windows 端按下 win + R 快捷键打开运行栏，输入 cmd 指令后按下回车。

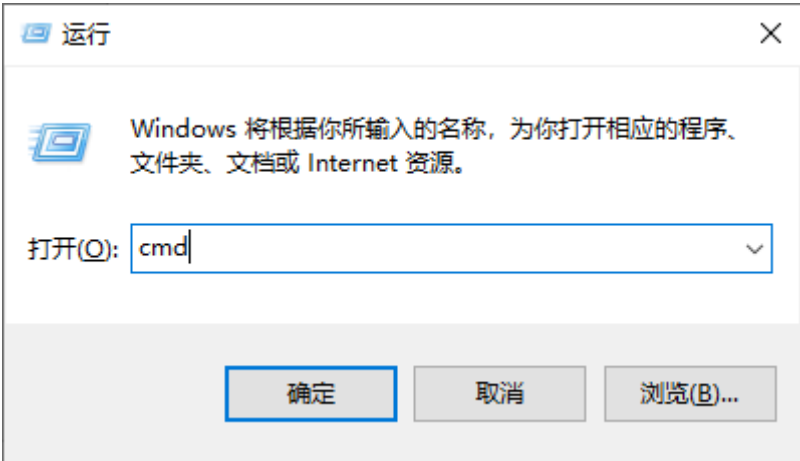


图 1.4.3-9 执行 cmd 指令

```
在打开的终端中输入 ipconfig 指令，查看电脑的 IP。

ipconfig

以太网适配器 以太网:

    连接特定的 DNS 后缀 . . . . . : 
    本地链接 IPv6 地址. . . . . : fe80::fc78:e034:2dd9:6901%16
    自动配置 IPv4 地址 . . . . . : 169.254.105.1
    子网掩码 . . . . . : 255.255.0.0
    默认网关. . . . . : 

以太网适配器 VMware Network Adapter VMnet1:

    连接特定的 DNS 后缀 . . . . . : 
    本地链接 IPv6 地址. . . . . : fe80::70ad:fa36:49a7:2c83%17
    IPv4 地址 . . . . . : 192.168.232.1
    子网掩码 . . . . . : 255.255.255.0
    默认网关. . . . . : 

以太网适配器 VMware Network Adapter VMnet8:

    连接特定的 DNS 后缀 . . . . . : 
    本地链接 IPv6 地址. . . . . : fe80::1113:c1e5:ba92:57b%20
    IPv4 地址 . . . . . : 192.168.18.1
    子网掩码 . . . . . : 255.255.255.0
    默认网关. . . . . : 

无线局域网适配器 WLAN:

    连接特定的 DNS 后缀 . . . . . : wifi
    本地链接 IPv6 地址. . . . . : fe80::a450:6cab:cd78:3708%18
    IPv4 地址 . . . . . : 192.168.3.188
    子网掩码 . . . . . : 255.255.255.0
    默认网关. . . . . : 192.168.3.1
```

图 1.4.3-10 查看 Windows 主机的 IP 信息

可以看到 Windows 主机的网络比较复杂，这里我们只截取可能用到的信息。根据 Windows 的 ipconfig 指令和 Ubuntu 虚拟网络编辑器，可以对应到以下表格。

网络适配器名	IP 地址	备注
--------	-------	----

以太网适配器 以太网	169.254.105.1	以太网的 IP 信息, 默认有一个 IP, 需要手动修改。如果显示已断开, 请将电脑网口接到开发板网口上
以太网适配器 VMware Network Adapter VMnet1	192.168.232.1	在 host-only 模式下, 动态分配虚拟系统的 TCP/IP 配置信息, 本文档不涉及
以太网适配器 VMware Network Adapter VMnet8	192.168.18.1	提供 DHCP 服务, 用于主机和虚拟机互通, 虚拟机上网
无线局域网适配器 WLAN	192.168.3.188	用于 Windows 主机上网
以太网适配器 VMware Network Adapter VMnet0		不提供 DHCP, 用于桥接网络, 要在虚拟网络编辑器设置手动设置 IP

表 1.4.3-1 主机 IP 表

在 Windows 终端没有看到 VMnet0, 在虚拟网络编辑器中可以看到。这是我们用来桥接的网络适配器, 不会自动分配 IP, 需要手动设置。这里要注意, **VMnet0 设置的网段不能和 VMnet1、VMnet8 的网段一样**, 不然会有冲突。

我们需要手动设置 Ubuntu 中桥接网络的 IP。

回到 Ubuntu 中, 打开网络适配器 1 的选项。



图 1.4.3-11 网络适配器 1 的选项

参考如下设置, 这里我们将连接名称改为 **VMnet0**, 方便区别。将 IP 手动设置为 192.168.10.100, 子网掩码 255.255.255.0, 网关 192.168.10.1。

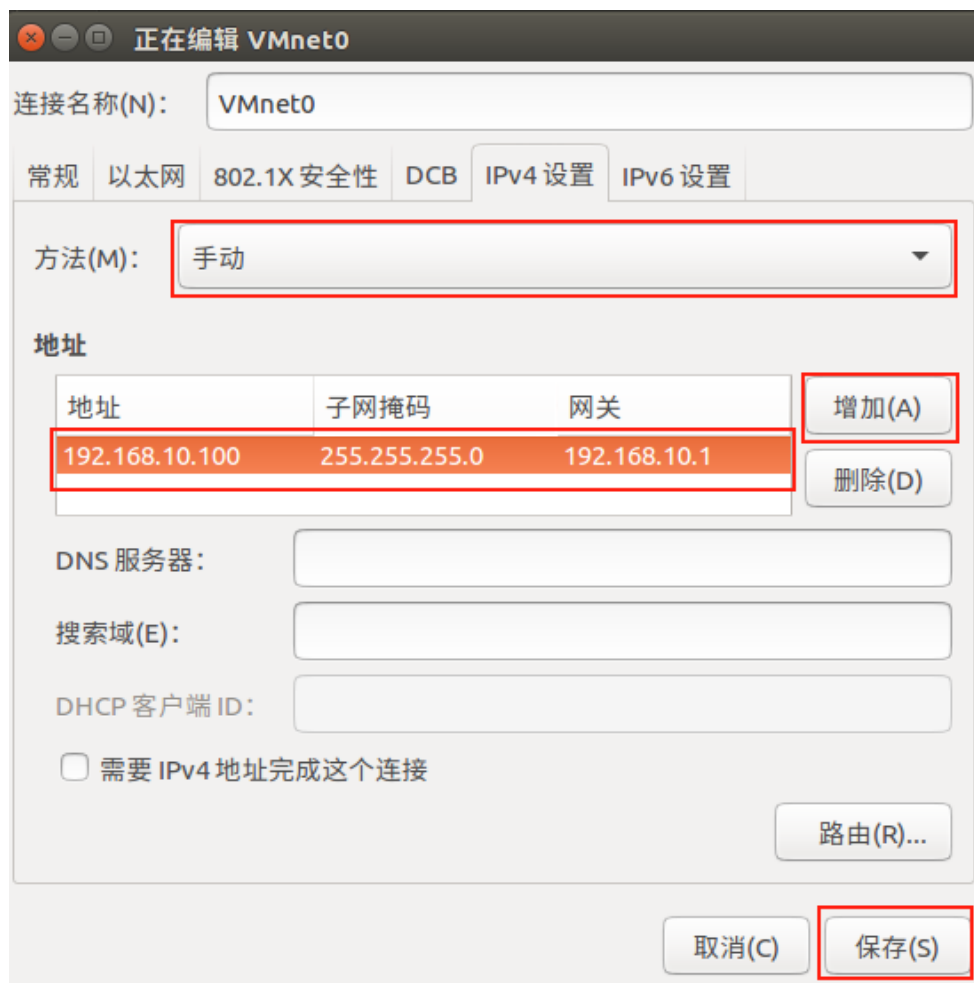


图 1.4.3-12 设置 VMnet0



图 1.4.3-13 重启网络服务来更新配置



图 1.4.3-14 Ubuntu 桥接网络设置完成

如果设置完成后, 虚拟机不能上网的话, 可以参考附录 5.2 小节的解决方案。

1.4.4 Windows 设置

设置完 Ubuntu 的 VMnet0 后, 再设置 Windows 的以太网 IP 的相关信息。

在 Windows 主机打开控制面板 -> 网络和 Internet -> 网络连接, 找到以太网。



图 1.4.4-1 网络连接

选择以太网, 右键打开菜单栏, 选择属性。在打开的网络设置中, 双击 Internet 协议版本 4 (TCP/IPv4), 如下图所示。

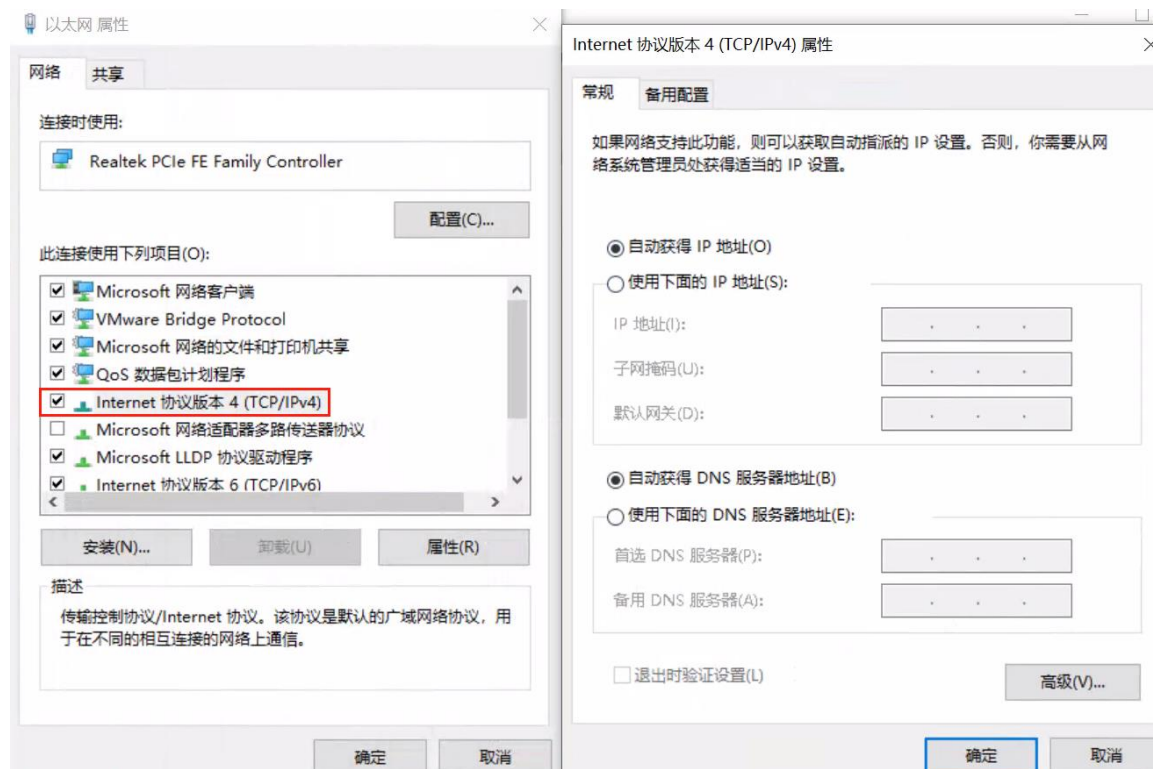


图 1.4.4-2 打开以太网 Internet 协议版本 4 (TCP/IPv4)

按照下图设置以太网的 IP 为 192.168.10.200, 要和虚拟机的 VMnet0 在同一网段。

常规

如果网络支持此功能, 则可以获取自动指派的 IP 设置。否则, 你需要从网络系统管理员处获得适当的 IP 设置。

☐ 自动获得 IP 地址(O)

☒ 使用下面的 IP 地址(S):

IP 地址(I): 192 . 168 . 10 . 200

子网掩码(U): 255 . 255 . 255 . 0

默认网关(D): 192 . 168 . 10 . 1

☐ 自动获得 DNS 服务器地址(B)

☒ 使用下面的 DNS 服务器地址(E):

首选 DNS 服务器(P): . . .

备用 DNS 服务器(A): . . .

☐ 退出时验证设置(L)

高级(V)...

确定 取消

图 1.4.4-3 设置以太网 IP 信息

1.4.5 开发板设置

接下来设置下开发板的 IP。这里我们将开发板 IP 设置为 192.168.10.50, 和 VMnet0 在同一网段即可。

进入开发板文件系统, 设置开发板 IP。

```
ifconfig eth0 up
ifconfig eth0 192.168.10.50
ifconfig
```

```
root@ATK-IMX6U:~# ifconfig eth0 up
root@ATK-IMX6U:~# ifconfig eth0 192.168.10.50
root@ATK-IMX6U:~#
root@ATK-IMX6U:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:04:9f:04:d2:35
          inet addr:169.254.44.233  Bcast:169.254.255.255  Mask:255.255.0.0
          inet6 addr: fe80::204:9fff:fe04:d235/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3131 errors:0 dropped:0 overruns:0 frame:0
          TX packets:934 errors:0 dropped:0 overruns:0 carrier:0
```

图 1.4.5-1 设置开发板 IP

1.4.6 ping 测试

开发板 ping 虚拟机 Ubuntu 的 IP,

```
ping 192.168.10.100
```

```
root@ATK-IMX6U:~# ping 192.168.10.100
PING 192.168.10.100 (192.168.10.100) 56(84) bytes of data.
64 bytes from 192.168.10.100: icmp_seq=1 ttl=64 time=9.88 ms
64 bytes from 192.168.10.100: icmp_seq=2 ttl=64 time=4.86 ms
64 bytes from 192.168.10.100: icmp_seq=3 ttl=64 time=5.18 ms
```

图 1.4.6-1 开发板 ping 虚拟机

开发板 ping 主机 Windows 的 IP。

```
ping 192.168.10.200
```

```
root@ATK-IMX6U:~# ping 192.168.10.200
PING 192.168.10.200 (192.168.10.200) 56(84) bytes of data.
64 bytes from 192.168.10.200: icmp_seq=1 ttl=128 time=9.79 ms
64 bytes from 192.168.10.200: icmp_seq=2 ttl=128 time=5.09 ms
64 bytes from 192.168.10.200: icmp_seq=3 ttl=128 time=4.93 ms
```

图 1.4.6-2 开发板 ping 主机

Ubuntu 虚拟机 ping 开发板的 IP。

```
ping 192.168.10.50
```

```
alientek@ubuntu16:~$ ping 192.168.10.50
PING 192.168.10.50 (192.168.10.50) 56(84) bytes of data.
64 bytes from 192.168.10.50: icmp_seq=1 ttl=64 time=7.89 ms
64 bytes from 192.168.10.50: icmp_seq=2 ttl=64 time=5.11 ms
64 bytes from 192.168.10.50: icmp_seq=3 ttl=64 time=5.08 ms
```

图 1.4.6-3 虚拟机 ping 开发板

Windows 主机 ping 开发板。

```
ping 192.168.10.50
```

```
C:\Users\ALIEN TEK>ping 192.168.10.50

正在 Ping 192.168.10.50 具有 32 字节的数据:
来自 192.168.10.50 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.10.50 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.10.50 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.10.50 的回复: 字节=32 时间<1ms TTL=64

192.168.10.50 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 0ms, 最长 = 0ms, 平均 = 0ms
```

图 1.4.6-4 主机 ping 开发板

以上就是开发板、Ubuntu 和 Windows 三者互 ping 测试, 在开发板 **uboot 命令行** 设置完开发板 IP 相关的信息后, 也能 ping 虚拟机 Ubuntu。如果了解 uboot 命令行和网络操作指令, 请先看《【正点原子】I.MX6U 嵌入式 Linux 驱动开发指南》的 30.3 小节和 30.4.4 小节。

```
setenv ipaddr 192.168.10.50
setenv ethaddr 00:04:9f:04:d2:35
setenv gatewayip 192.168.10.1
setenv netmask 255.255.255.0
setenv serverip 192.168.10.100
saveenv

ping 192.168.10.100

Hit any key to stop autoboot: 0
=>
=> setenv ipaddr 192.168.10.50
=> setenv ethaddr 00:04:9f:04:d2:35
=> setenv gatewayip 192.168.10.1
=> setenv netmask 255.255.255.0
=> setenv serverip 192.168.10.100
=> saveenv
Saving Environment to MMC...
Writing to MMC(1)... done
=>
=> ping 192.168.10.100
FEC1 Waiting for PHY auto negotiation to complete..... done
Using FEC1 device
host 192.168.10.100 is alive
```

图 1.4.6-5 uboot 阶段设置网络并测试 ping 虚拟机

uboot 阶段时, Ubuntu 虚拟机 ping 开发板是没有反馈的, 所以不用进行这一步。按照上述完成测试后, 就可以使用 TFTP 和 NFS 来挂载内核、设备树和文件系统。搭建 TFTP 请看[第二章](#)的内容, 搭建 NFS 请看第三章的内容。

1.4.7 小结

防火墙设置: 关闭电脑和虚拟机的防火墙。

网络连接: 电脑使用无线上网, 开发板网口直连电脑网口。

VMwear 设置: 设置网络适配器为桥接模式; 添加一个网络适配器 2 为 NAT 模式。

Ubuntu 设置: 设置 NAT 模式的网络适配器为自动获取模式 (DHCP); 手动设置桥接模式的网络适配器的 IP 信息。

Windows 设置: 手动设置以太网 TCP/IPv4 属性中的 IP 信息。

开发板设置: 手动设置开发板的 IP 信息。

注意: Ubuntu 里桥接模式的网络适配器、电脑的以太网、开发板的网口, 三者要在同一网段, 且不能与 NAT 模式的网络适配器、虚拟机子网在同一网段。Windows 和 Ubuntu 通过 NAT 模式的网络适配器通信。

1.5 电脑只有无线网卡

1.5.1 准备工作

使用场景: 一般电脑都是有一个以太网网口的, 但是一些超薄本、商务本电脑没有以太网网口, 这样并不方便我们的开发。我们可以使用 USB 转网口设备来给电脑拓展出一个以太网网口, 因为开发板必选通过有线的方式连接到电脑上。USB 转网口设备如下图所示。



图 1.5.1-1 USB 转网口

这里要说明下, 有些 USB 转网口遇到无法免驱使用、校园网使用异常、使用后电脑无法上网等问题, 可以向购买的店家咨询下相关问题, 尽量选择有品牌、销量好的产品。比如本文档使用的就是绿联的 USB 转网口。

设备: 电脑 (Windows)、正点原子 Linux 开发板 (出厂系统)、直通网线、USB 转网口。

网络拓扑结构:

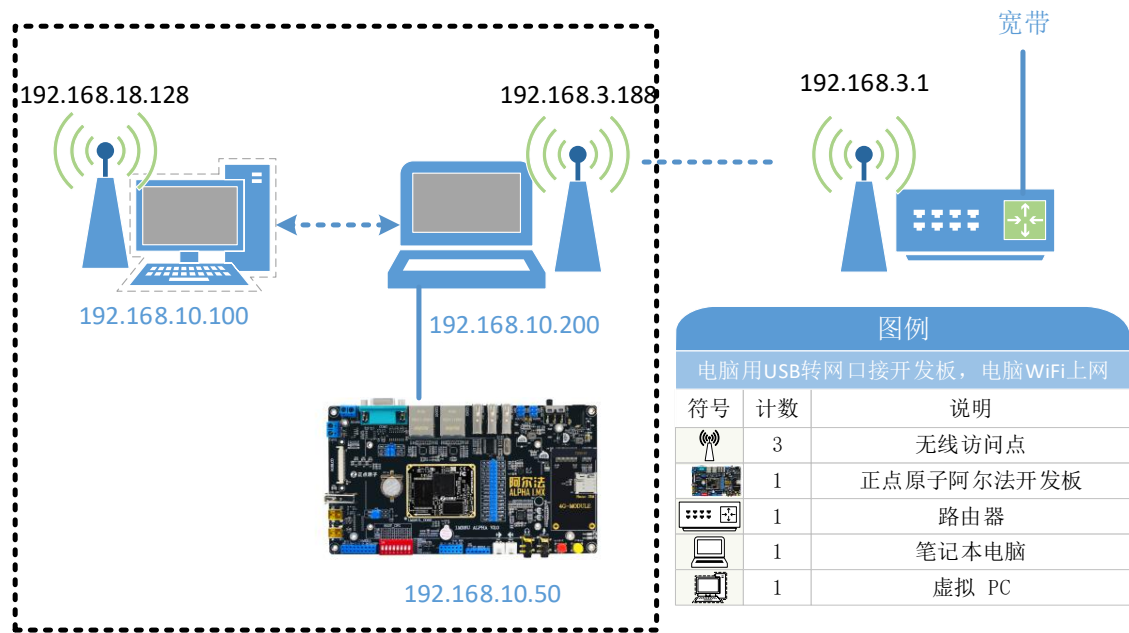
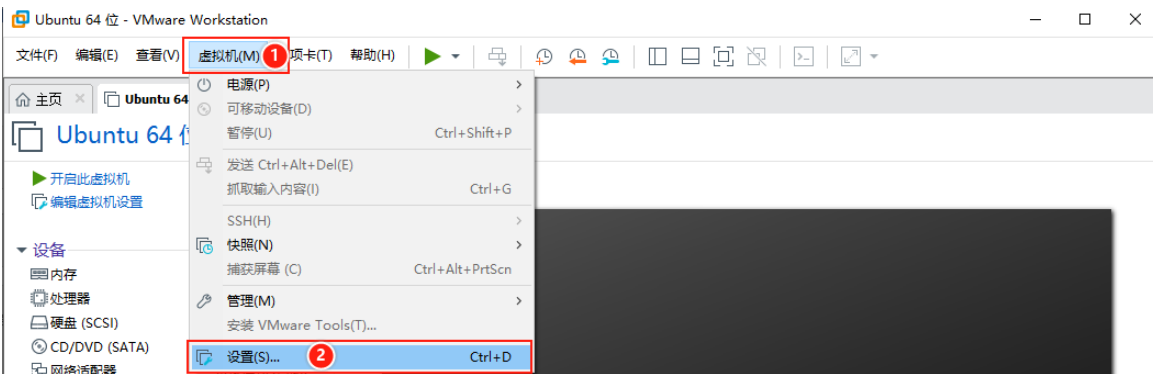


图 1.5.1-2 电脑 USB 转网口接开发板

1.5.2 VMware 设置

打开 VMware Workstation Pro 里的 虚拟机 -> 设置



设置网络适配器为桥接模式。

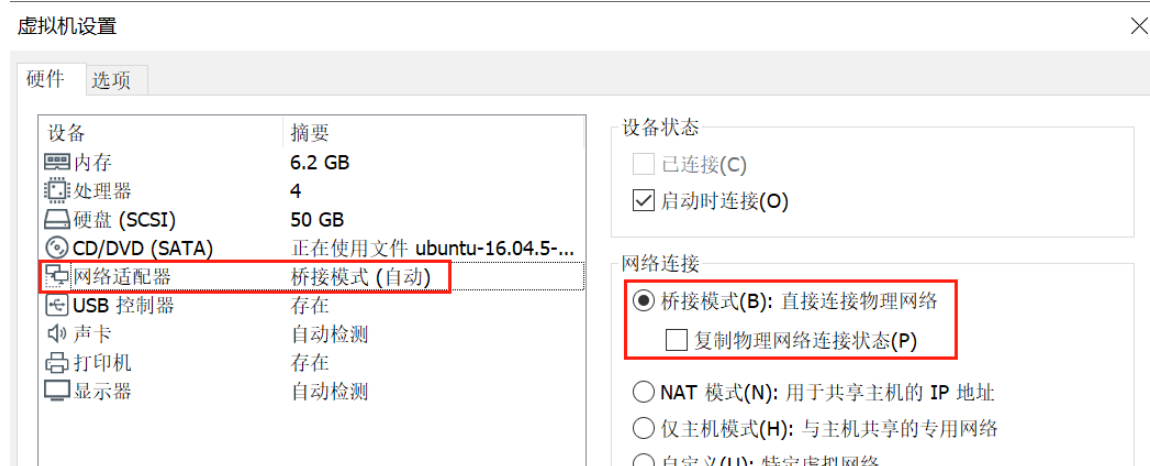


图 1.5.2-2 设置桥接模式

因为电脑是 WIFI 上网,所以需要添加一个网络适配器并设置成 NAT 模式,供虚拟机上网。具体操作如下图所示。

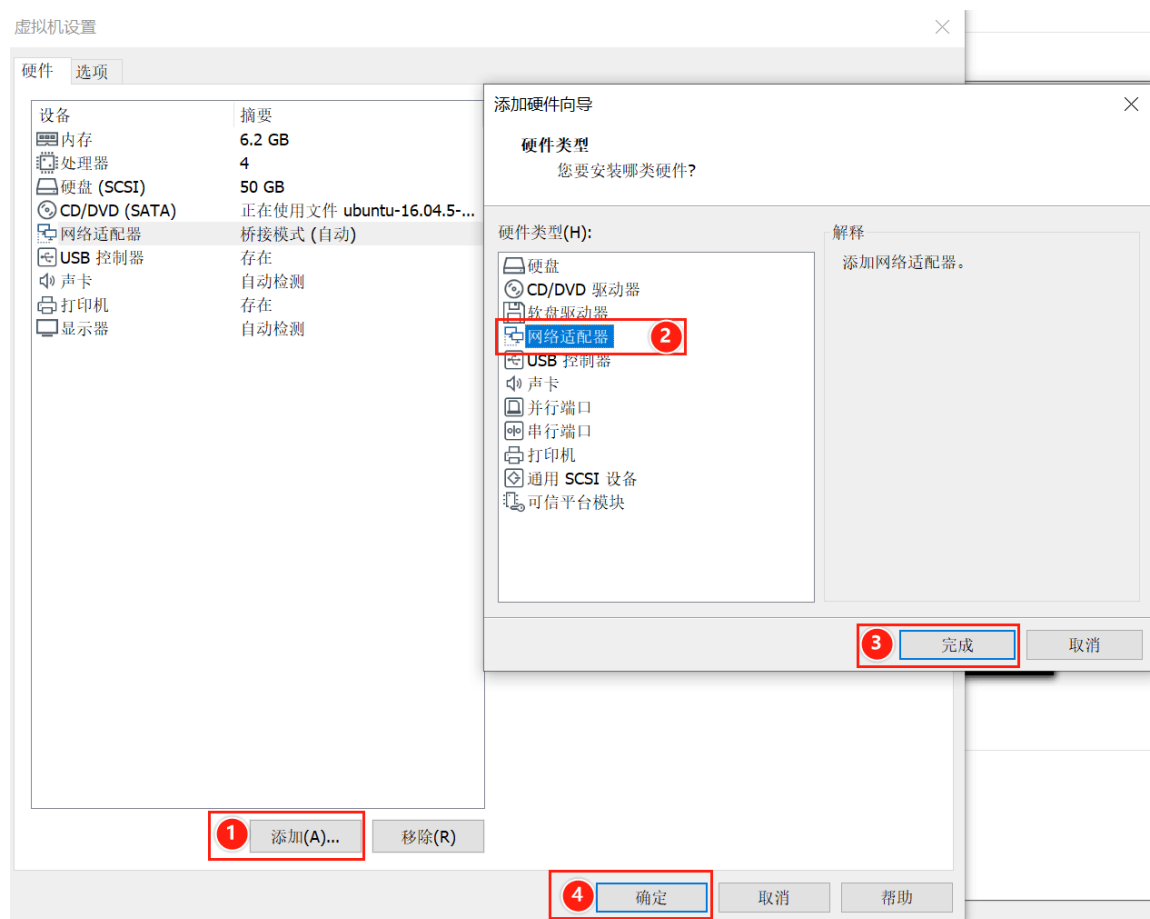


图 1.5.2-3 添加网络适配器

默认添加的网络适配器是 NAT 模式的,如果不是 NAT 模式则要手动设置成 NAT 模式。

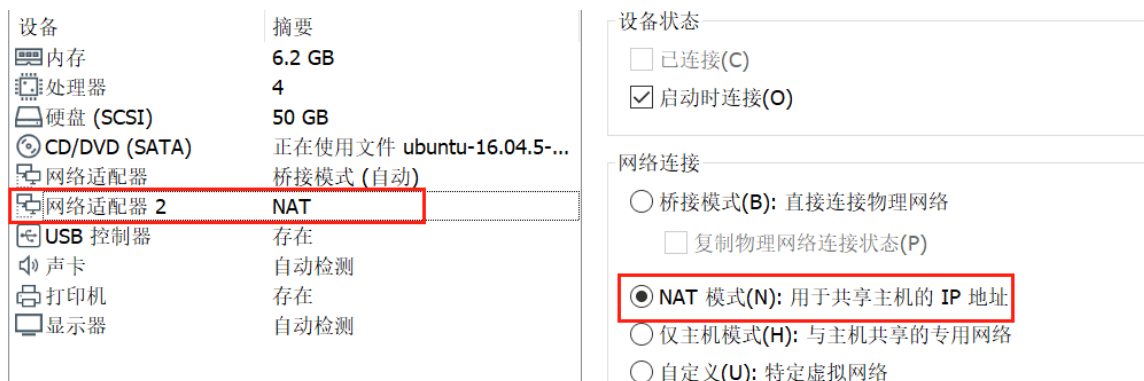


图 1.5.2-4 设置网络适配器 2 为 NAT 模式

打开菜单栏的 编辑 -> 虚拟网络编辑器。

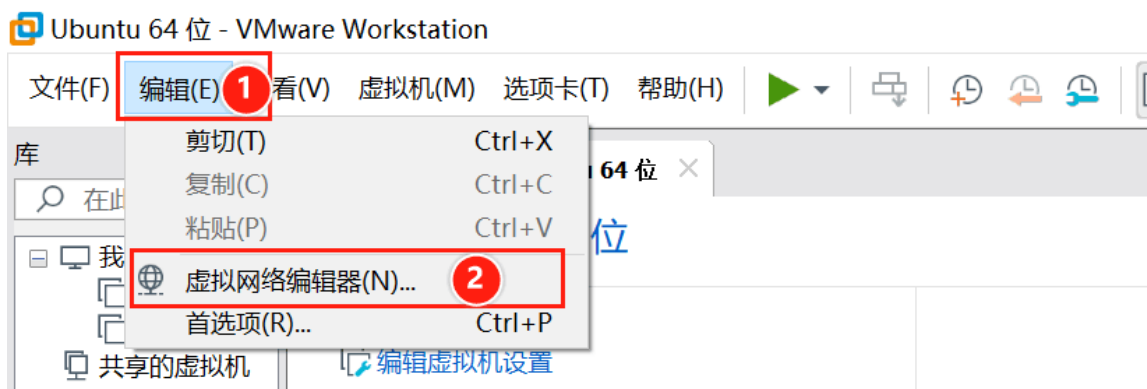


图 1.5.2-5 打开虚拟网络编辑器

点击虚拟网络编辑器的 更改设置 选项。



图 1.5.2-6 更改设置

这个需要电脑管理员权限，如果有提示用户账户控制的提示框，点击 是 就可以了。

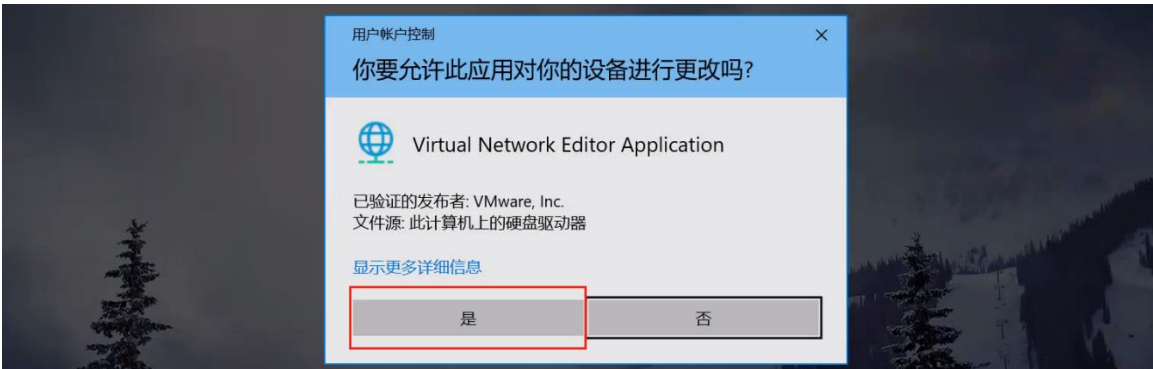


图 1.5.2-7 赋予修改权限

重新打开虚拟网络编辑器，可以看到如下界面。



图 1.5.2-8 赋予权限后的虚拟网络编辑器界面

如果之前设置乱了，可以点击虚拟网络编辑器左下角的还原默认设置，将虚拟机网络复原到默认设置。

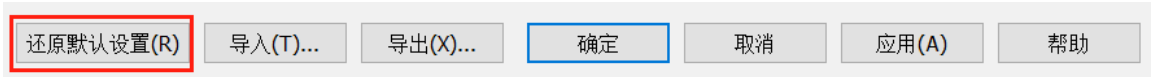


图 1.5.2-9 还原默认设置

开发板是直连到电脑的网口的，所以需要虚拟网络编辑器里的网络适配器 1（即 VMnet0）桥接到有线网卡上。但是电脑网口已经被宽带占用了，我们要用到 USB 转网口来拓展一个网口。这里先不接 USB 转网口，打开 VMnet0 的桥接模式设置，可以看到可以桥接的选项。

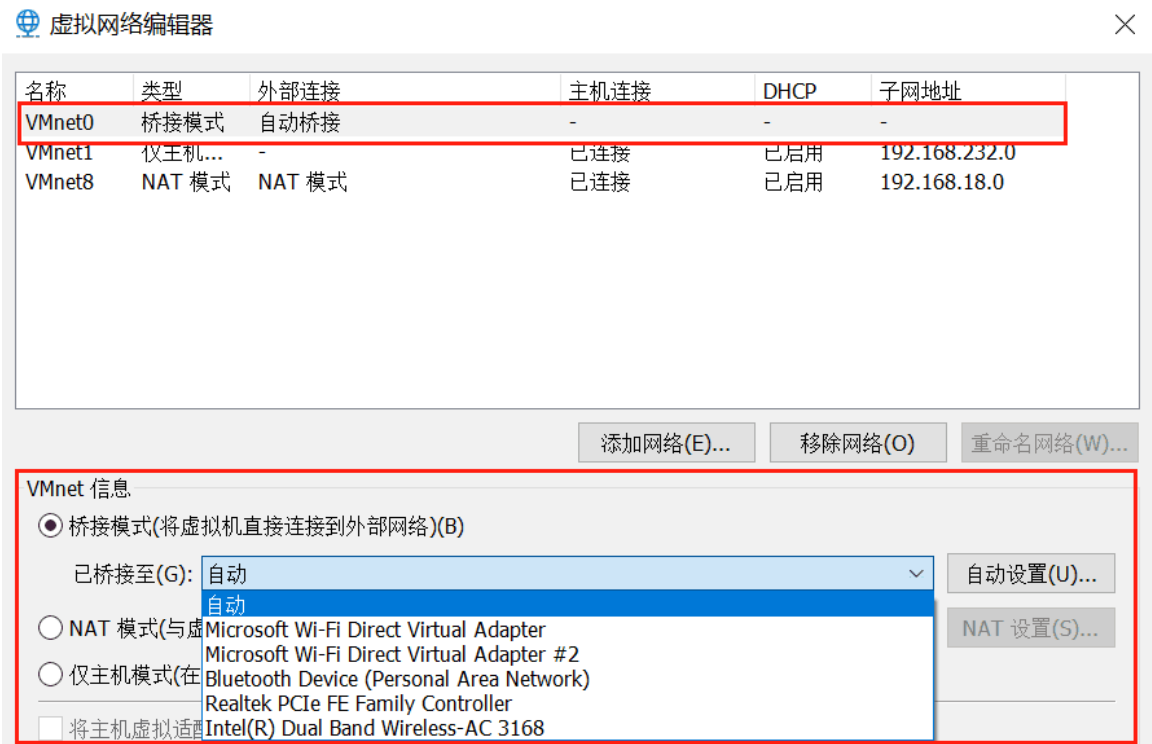


图 1.5.2-10 没接入 USB 转网口前的可桥接选项

此时我们需要关闭虚拟网络编辑器,再接入 USB 转网口,再打开虚拟网络编辑器中 VMnet0 的桥接选项,记住一定要重新打开虚拟网络编辑器,可以看到多了一个 USB2.0 的桥接选项,这个就是我们的 USB 转网口。(注:不同的 USB 转网口设备名不同。以带 USB 字符的为准。)

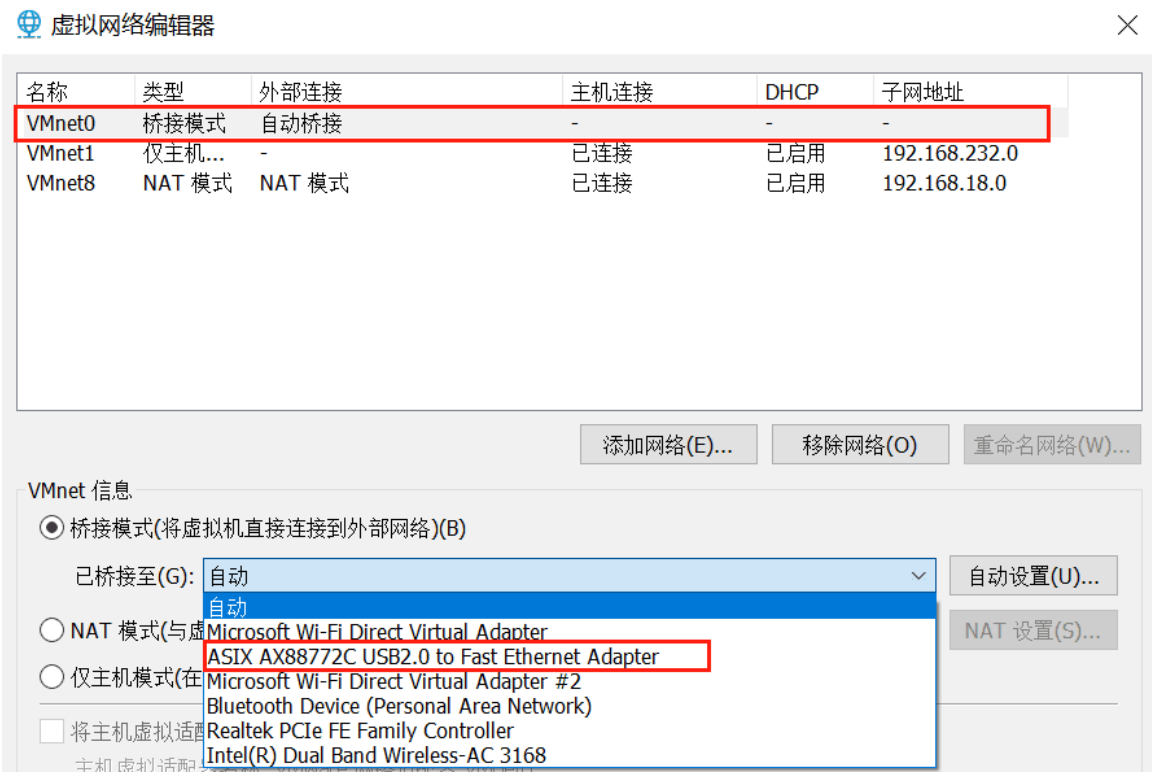


图 1.5.2-11 接入 USB 转网口后的可桥接选项

以上步骤是在没开启 Ubuntu 虚拟机前操作的, 如果用户此时电脑在运行 Ubuntu 虚拟机, 则会弹出 USB 设备连接提示, 这里注意要点连接到主机 (也就是连接到 Windows) !



图 1.5.2-12 将 USB 转网口设备连接到 Windows 主机

如果没有 USB 设备连接提示, 请按如图所示检查 USB 转网口设备的连接状态, 不要让设备连接到虚拟机。

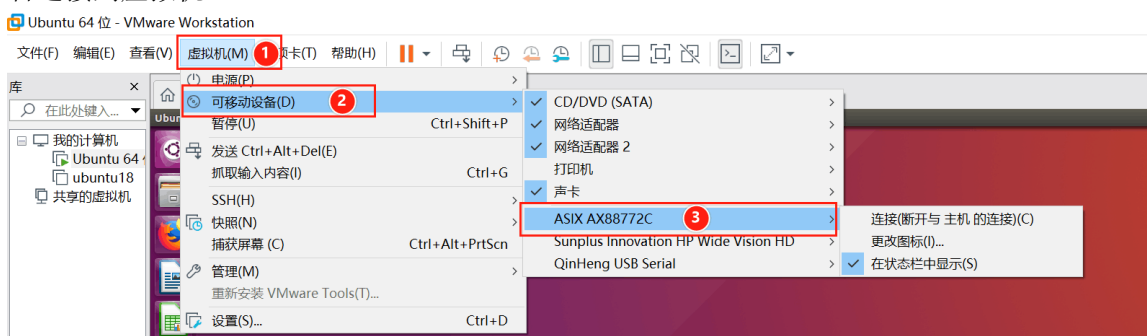


图 1.5.2-13 查看 USB 转网口设备连接状态

如果勾选了连接, 点击取消断开连接。(虚拟机可能默认将设备接入到 Ubuntu 中, 所以要检查下)

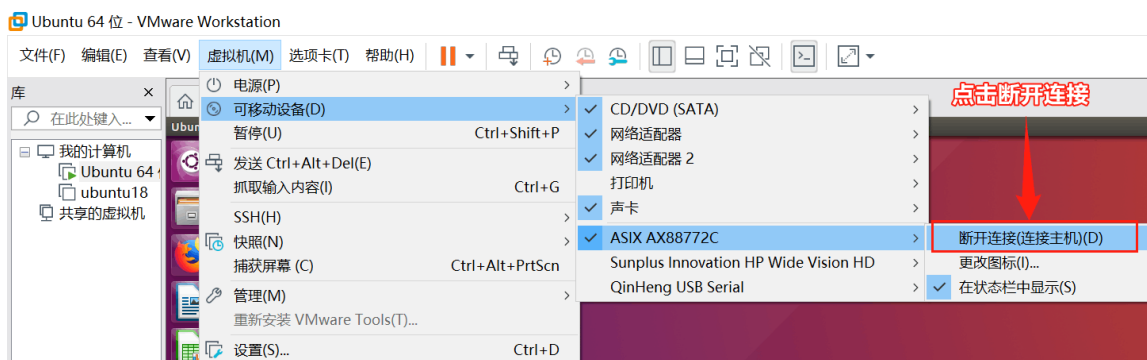


图 1.5.2-14 将 USB 转网口从虚拟机中断开

我们需要将 USB 转网口接在 Windows 主机端。接入主机后，在 Windows 主机的设备管理器 -> 网络适配器 里也可以看到我们这个 USB 转网口的设备。



图 1.5.2-15 Windows 设备管理器中查看 USB 转网口

如果没有识别到，可能设备驱动有问题，请咨询下商家。

回到虚拟网络编辑器中，将 VMnet0 桥接到 USB 转网口，如下图所示。

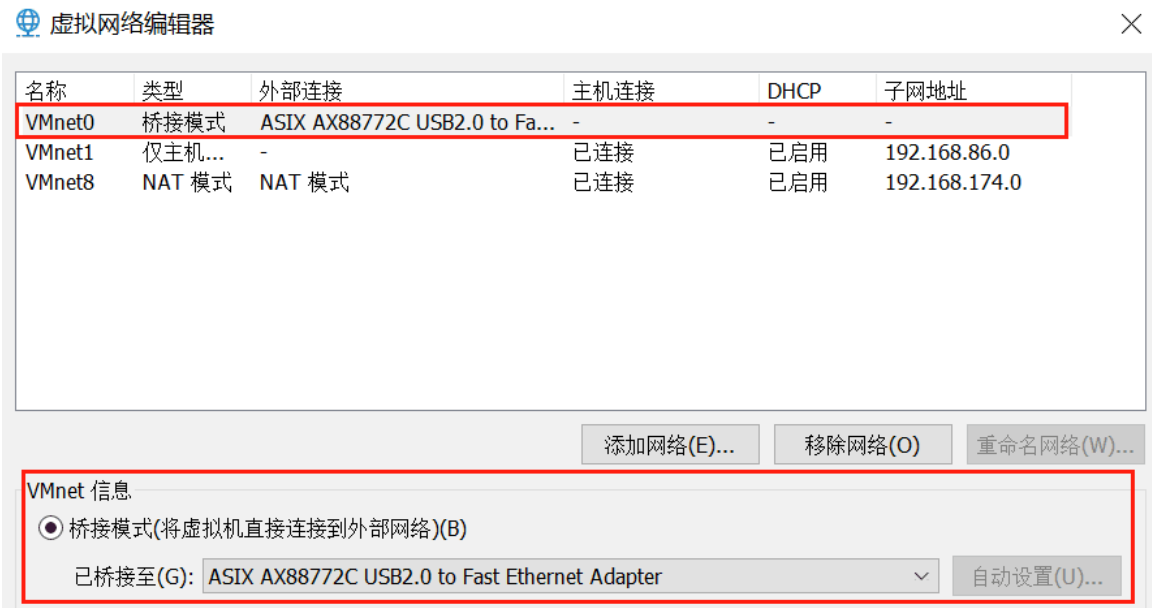


图 1.5.2-16 将 VMnet0 桥接到 USB 转网口

用直通网线将开发板和 USB 转网口设备的网口端连接，开发板上电。设置完成后，进入虚拟机 Ubuntu 系统进行网络设置，方法和 1.4.3 小节的一样，请参考 [1.4.3 小节](#)。

第二章 搭建 TFTP 环境

本章旨在快速搭建 TFTP 环境, 需要用户有熟练使用 Shell 操作和 vi 编辑器的能力, 如果不熟悉这些的, 请先学习下《【正点原子】I.MX6U 嵌入式 Linux 驱动开发指南》2.3 小节和 2.5.2 小节。

本章所使用的 Ubuntu 条件: Ubuntu16.04 版本, 能上网。

2.1 TFTP 简介

TFTP (Trivial File Transfer Protocol, 简单文件传输协议) 是 TCP/IP 协议族中的一个用来在客户机与服务器之间进行简单文件传输的协议, 提供不复杂、开销不大的文件传输服务。我们可以使用 TFTP 来加载内核 zImage、设备树和其他较小的文件到开发板 DDR 上, 从而实现网络挂载。

2.2 搭建 TFTP

2.2.1 安装和配置 xinetd

执行以下指令, 安装 xinetd。

```
sudo apt-get install xinetd
```

```
alientek@ubuntu16:~$ sudo apt-get install xinetd
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
下列【新】软件包将被安装:
  xinetd
升级了 0 个软件包, 新安装了 1 个软件包, 要卸载 0 个软件包, 有 99 个软件包未被升级。
需要下载 107 kB 的归档。
解压缩后会消耗 299 kB 的额外空间。
0% [执行中]
获取:1 http://mirrors.aliyun.com/ubuntu xenial/main amd64 xinetd amd64 1:2.3.15-6 [107 kB]
已下载 107 kB, 耗时 1秒 (57.4 kB/s)
正在选中未选择的软件包 xinetd。
(正在读取数据库 ... 系统当前共安装有 238387 个文件和目录。)
正准备解包 .../xinetd_1%3a2.3.15-6_amd64.deb ...
正在解包 xinetd (1:2.3.15-6) ...
正在处理用于 man-db (2.7.5-1) 的触发器 ...
正在处理用于 doc-base (0.10.7) 的触发器 ...
Processing 1 added doc-base file...
正在处理用于 ureadahead (0.100.0-19) 的触发器 ...
正在处理用于 systemd (229-4ubuntu21.27) 的触发器 ...
正在设置 xinetd (1:2.3.15-6) ...
```

图 2.2.1-1 安装 xinetd

查询/etc/下是否存在 xinetd.conf 文件, 没有的话则自己新建一个。已经有 xinetd.conf 文件可以跳到 2.2.2 小节。

```
ls /etc/xinetd.conf
```

```
alientek@ubuntu16:~$ ls /etc/xinetd.conf
/etc/xinetd.conf
alientek@ubuntu16:~$
```

图 2.2.1-2 查询 xinetd.conf 文件

这里已经查到有 xinetd.conf 文件, 如果没有, 就执行以下指令创建一个 xinetd.conf 文件。

```
sudo vi /etc/xinetd.conf
```

创建出来的文件是空白的, 修改 xinetd.conf 文件内容如下:

```
# Simple configuration file for xinetd
#
# Some defaults, and include /etc/xinetd.d/
defaults
{
# Please note that you need a log_type line to be able to use log_on_success
# and log_on_failure. The default is the following :
# log_type = SYSLOG daemon info
}
includedir /etc/xinetd.d
```

```
# Simple configuration file for xinetd
#
# Some defaults, and include /etc/xinetd.d/
defaults
{
# Please note that you need a log_type line to be able to use log_on_success
# and log_on_failure. The default is the following :
# log_type = SYSLOG daemon info
}
includedir /etc/xinetd.d
```

图 2.2.1-3 xinetd.conf 内容

2.2.2 TFTP 目录

新建 TFTP 目录, 这里建立在 /home/alientek/linux 目录下, 目录名为 tftp。将 tftp 目录赋予可读可写可执行权限。

```
mkdir -p /home/alientek/linux/tftp
sudo chmod 777 /home/alientek/linux/tftp/
cd /home/alientek/linux/
ls
```

```
alientek@ubuntu16:~/linux$ mkdir -p /home/alientek/linux/tftp
alientek@ubuntu16:~/linux$ sudo chmod 777 /home/alientek/linux/tftp/
alientek@ubuntu16:~/linux$ cd /home/alientek/linux/
alientek@ubuntu16:~/linux$ ls
IMX6ULL  my_linux  tftp  tool
```

图 2.2.2-1 创建 tftp 目录并赋予权限

2.2.3 tftp-hpa 和 tftpd-hpa 服务程序

执行以下程序安装 tftp-hpa 和 tftpd-hpa 服务程序

```
sudo apt-get install tftp-hpa tftpd-hpa
```

```

alientek@ubuntu16:~/linux$ sudo apt-get install tftp-hpa tftpd-hpa
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
建议安装:
  pxelinux
下列【新】软件包将被安装:
  tftp-hpa tftpd-hpa
升级了 0 个软件包, 新安装了 2 个软件包, 要卸载 0 个软件包, 有 99 个软件包未被升级。
需要下载 57.1 kB 的归档。
解压缩后会消耗 170 kB 的额外空间。
获取:1 http://mirrors.aliyun.com/ubuntu xenial-updates/main amd64 tftp-hpa amd64 5.2+20150808-1ubuntu1.16.04.1 [18.0 kB]
获取:2 http://mirrors.aliyun.com/ubuntu xenial-updates/main amd64 tftpd-hpa amd64 5.2+20150808-1ubuntu1.16.04.1 [39.1 kB]
已下载 57.1 kB, 耗时 0秒 (317 kB/s)
正在预设软件包 ...
正在选中未选择的软件包 tftp-hpa。
(正在读取数据库 ... 系统当前共安装有 238392 个文件和目录。)
正准备解包 .../tftp-hpa_5.2+20150808-1ubuntu1.16.04.1_amd64.deb ...
正在解包 tftp-hpa (5.2+20150808-1ubuntu1.16.04.1) ...
正在选中未选择的软件包 tftpd-hpa。
正准备解包 .../tftpd-hpa_5.2+20150808-1ubuntu1.16.04.1_amd64.deb ...
正在解包 tftpd-hpa (5.2+20150808-1ubuntu1.16.04.1) ...
正在处理用于 man-db (2.7.5-1) 的触发器 ...
正在处理用于 ureadahead (0.100.0-19) 的触发器 ...
正在处理用于 systemd (229-4ubuntu21.27) 的触发器 ...
正在设置 tftp-hpa (5.2+20150808-1ubuntu1.16.04.1) ...
正在设置 tftpd-hpa (5.2+20150808-1ubuntu1.16.04.1) ...
tftpd user (tftp) already exists, doing nothing.

```

图 2.2.3-1 安装 tftp-hpa 和 tftpd-hpa

执行以下指令打开 tftpd-hpa 配置文件, 修改 tftp 目录为 TFTP 服务器工作目录。

```
sudo vi /etc/default/tftpd-hpa
```

```

# /etc/default/tftpd-hpa

TFTP_USERNAME="tftp"
TFTP_DIRECTORY="/home/alientek/linux/tftp"
TFTP_ADDRESS=":69"
TFTP_OPTIONS="-l -c -s"

```

图 2.2.3-2 修改 tftpd-hpa

执行以下指令创建/etc/xinetd.d/tftp 配置文件。(如果没有 xinetd.d 这个目录, 可以先自己手动创建)

```
sudo vi /etc/xinetd.d/tftp
```

```

server tftp
{
    socket_type = dgram
    wait = yes
    disable = no
    user = root
    protocol = udp
    server = /usr/sbin/in.tftpd
    server_args = -s /home/alientek/linux/tftp -c
    #log_on_success += PID HOST DURATION
    #log_on_failure += HOST
    per_source = 11
    cps = 100 2
    flags = IPv4
}

```

图 2.2.3-3 需要添加/etc/xinetd.d/tftp 内容

添加如下内容, 保存即可。

```
server tftp
{
    socket_type = dgram
    wait = yes
    disable = no
    user = root
    protocol = udp
    server = /usr/sbin/in.tftpd
    server_args = -s /home/alientek/linux/tftp -c
    #log_on_success += PID HOST DURATION
    #log_on_failure += HOST
    per_source = 11
    cps = 100 2
    flags = IPv4
}
```

注意 `server_args = -s` 后面要添加自己的 tftp 工作路径。

修改/添加 tftp 文件后, 执行以下指令重启 tftpd-hpa。

```
sudo service tftpd-hpa restart
```

重启 xinetd 服务。

```
sudo service xinetd restart
```

```
alientek@ubuntu16:~/linux$ sudo service tftpd-hpa restart
alientek@ubuntu16:~/linux$ sudo service xinetd restart
```

图 2.2.3-4 重启 tftpd-hpa 和 xinetd

2.2.4 网络环境

确保网络环境正常, Ubuntu、Windows 和开发板能相互 ping 通。

这里结合 1.4 小节的配置来验证, 即:

开发板 IP: 192.168.10.50

虚拟机 IP: 192.168.10.100

电脑网口的 IP: 192.168.10.200

使用 1.2 小节或者 1.3 小节的搭建环境也可以, 注意 IP 是否对应正确。

2.2.5 TFTP 测试

在 /home/alientek/linux/tftp 目录下创建一个 test.c 文件, 在里面写入 hello alientek!

```
alientek@ubuntu16:~/linux/tftp$ vi test.c
alientek@ubuntu16:~/linux/tftp$ cat test.c
hello alientek!
alientek@ubuntu16:~/linux/tftp$
```

图 2.2.5-1 test.c 文件

在开发板文件系统执行以下指令设置开发板 IP, 将虚拟机 (192.168.10.100) TFTP 工作目录下的 test.c 文件拷贝到开发板中。

```
ifconfig eth0 192.168.10.50  
tftp -g -r test.c 192.168.10.100  
cat test.c
```

```
root@ATK-IMX6U:~# ifconfig eth0 192.168.10.50  
root@ATK-IMX6U:~# tftp -g -r test.c 192.168.10.100  
root@ATK-IMX6U:~# cat test.c  
hello alientek!  
root@ATK-IMX6U:~#
```

图 2.2.5-2 测试 TFTP

可以看到 test.c 成功传输到开发板上, TFTP 环境搭建成功。

第三章 搭建 NFS 环境

本章旨在快速搭建 NFS 环境, 需要用户有熟练使用 Shell 操作和 vi 编辑器的能力, 如果不熟悉这些的, 请先学习下《【正点原子】I.MX6U 嵌入式 Linux 驱动开发指南》2.3 小节和 2.5.2 小节。

本章所使用的 Ubuntu 条件: Ubuntu16.04 版本, 能上网。

3.1 NFS 简介

网络文件系统, 英文 Network File System(NFS), 是由 SUN 公司研制的 UNIX 表示层协议 (presentation layer protocol), 能使使用者访问网络上别处的文件就像在使用自己的计算机一样。我们可以在 Ubuntu 上制作开发板的根文件系统, 然后使用 NFS 来加载根文件系统到开发板的 DDR 上。

3.2 安装 NFS 服务

在 Ubuntu 终端执行以下指令安装 NFS。

```
sudo apt-get install nfs-kernel-server
```



```
alientek@ubuntu16:~$ sudo apt-get install nfs-kernel-server
[sudo] alientek 的密码:
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
下列【新】软件包将被安装:
  nfs-kernel-server
升级了 0 个软件包, 新安装了 1 个软件包, 要卸载 0 个软件包, 有 99 个软件包未被升级。
需要下载 0 B/87.8 kB 的归档。
解压缩后会消耗 487 kB 的额外空间。
正在选中未选择的软件包 nfs-kernel-server。
(正在读取数据库 ... 系统当前共安装有 238377 个文件和目录。)
正准备解包 .../nfs-kernel-server_1%3a1.2.8-9ubuntu12.3_amd64.deb ...
正在解包 nfs-kernel-server (1:1.2.8-9ubuntu12.3) ...
正在处理用于 systemd (229-4ubuntu21.27) 的触发器 ...
正在处理用于 ureadahead (0.100.0-19) 的触发器 ...
正在处理用于 man-db (2.7.5-1) 的触发器 ...
正在设置 nfs-kernel-server (1:1.2.8-9ubuntu12.3) ...
alientek@ubuntu16:~$
```

图 2.2.5-1 安装 NFS

3.3 NFS 共享目录

新建 NFS 共享目录, 并给予 NFS 目录可读可写可执行权限。

```
sudo mkdir /home/alientek/linux/nfs
```

```
sudo chmod 777 /home/alientek/linux/nfs/
```

```
alientek@ubuntu16:~$ sudo mkdir /home/alientek/linux/nfs
alientek@ubuntu16:~$ sudo chmod 777 /home/alientek/linux/nfs/
alientek@ubuntu16:~$ cd /home/alientek/linux/
alientek@ubuntu16:~/linux$ ls
IMX6ULL  my_linux  nfs  tftp  tool
alientek@ubuntu16:~/linux$
```

图 2.2.5-1 创建 NFS 目录并赋予权限

3.4 配置 NFS 服务

执行以下指令打开 `etc/exports` 文件

```
sudo vi /etc/exports
```

进入 `etc/exports` 文件, 在最后添加如下内容

```
/home/alientek/linux/nfs *(rw,sync,no_root_squash)
```

`/home/alientek/linux/nfs` 表示 NFS 共享的目录

`*`表示允许所有的网络段访问

`rw` 表示访问者具有可读写权限

`sync` 表示将缓存写入设备中, 可以说是同步缓存的意思

`no_root_squash` 表示访问者具有 root 权限。

修改完如下图所示。

```
# /etc/exports: the access control list for filesystems which may be exported
# to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_c
heck)
#
# Example for NFSv4:
# /srv/nfs4 gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes gss/krb5i(rw,sync,no_subtree_check)
#
/home/alientek/linux/nfs *(rw,sync,no_root_squash)
```

图 2.2.5-1 修改/etc/exports

修改完以后保存退出。

执行以下指令重启 NFS 服务器。

```
sudo /etc/init.d/nfs-kernel-server restart
```

```
alientek@ubuntu16:~/linux$ sudo /etc/init.d/nfs-kernel-server restart
[ ok ] Restarting nfs-kernel-server (via systemctl): nfs-kernel-server.service.
alientek@ubuntu16:~/linux$
```

图 2.2.5-2 重启 NFS

执行以下指令查看 NFS 共享目录。

```
showmount -e
```

```
alientek@ubuntu16:~/linux$ showmount -e
Export list for ubuntu16:
/home/alientek/linux/nfs *
```

图 2.2.5-3 查看 NFS 共享路径

3.5 网络环境

确保网络环境正常, Ubuntu、Windows 和开发板能相互 ping 通。

这里结合 1.4 小节的配置来验证, 即:

开发板 IP: 192.168.10.50

虚拟机 IP: 192.168.10.100

电脑网口的 IP: 192.168.10.200

使用 1.2 小节或者 1.3 小节的搭建环境也可以, 注意 IP 是否对应正确。

3.6 NFS 测试

在 /home/alientek/linux/nfs 目录下创建一个 mytest.c 文件, 在里面写入 hello world!

```
alientek@ubuntu16:~/linux/nfs$ ls
alientek@ubuntu16:~/linux/nfs$ vi mytest.c
alientek@ubuntu16:~/linux/nfs$ cat mytest.c
hello world!
alientek@ubuntu16:~/linux/nfs$
```

图 2.2.5-1 mytest.c 文件

执行以下指令设置开发板 IP, 创建一个 get 目录, 将虚拟机 (192.168.10.100) NFS 共享目录挂载到开发板的 get 目录中。

```
mkdir get
```

```
mount -t nfs -o nolock,nfsvers=3 192.168.10.100:/home/alientek/linux/nfs get/
```

```
root@ATK-IMX6U:~# mkdir get
root@ATK-IMX6U:~# mount -t nfs -o nolock,nfsvers=3 192.168.10.100:/home/alientek/linux/nfs get/
root@ATK-IMX6U:~# ls
driver get shell test.c
root@ATK-IMX6U:~# cd get/
root@ATK-IMX6U:~/get# ls
mytest.c
root@ATK-IMX6U:~/get# cat mytest.c
hello world!
root@ATK-IMX6U:~/get#
```

图 2.2.5-2 测试 NFS

查看挂载的 NFS 目录:

```
df
```

```
root@ATK-IMX6U:~# df
Filesystem            1K-blocks    Used Available Use% Mounted on
/dev/root              7244864    517736   6352448   8% /
devtmpfs              187632      4    187628   1% /dev
tmpfs                  40          0         40      0% /mnt/.psplash
tmpfs                 253432     160    253272   1% /run
tmpfs                 253432     156    253276   1% /var/volatile
/dev/mmcblk1p1        129039     6899    122141   6% /run/media/mmcblk1p1
192.168.10.100:/home/alientek/linux/nfs 307487744 80192512 211652608 28% /home/root/get
root@ATK-IMX6U:~#
```

图 2.2.5-3 查看挂载目录

卸载 NFS 目录:

```
umount get
```

```
root@ATK-IMX6U:~# umount get
root@ATK-IMX6U:~# df
Filesystem      1K-blocks    Used Available Use% Mounted on
/dev/root        7244864 517736   6352448   8% /
devtmpfs         187632      4    187628    1% /dev
tmpfs            40          0        40     0% /mnt/.psplash
tmpfs            253432    156    253276    1% /run
tmpfs            253432    156    253276    1% /var/volatile
/dev/mmcblkp1    129039    6899    122141    6% /run/media/mmcblkp1
root@ATK-IMX6U:~#
```

图 2.2.5-4 卸载 NFS 目录

可以看到 192.168.10.100:/home/alientek/linux/nfs 已经卸载了。

第四章 挂载教程系统

此章节为了方便一些用户的快速开发,要求用户至少会编译原子修改好的内核源码且会搭建 busybox 根文件系统,如果不会这些基本操作,请先去看《【正点原子】I.MX6U 嵌入式 Linux 驱动开发指南》中的相关章节。

以下以 emmc 版本开发板、7 寸 1024*600 分辨率屏幕为例,使用网络挂载 I.MX6U 的教程系统。

4.1 准备工作

4.1.1 网络环境

已经完成网络环境 ping 测试,完成 TFTP 和 NFS 环境搭建和验证。

4.1.2 内核文件准备

将光盘里的教程系统镜像中的 **zImage** 文件和 **imx6ull-alientek-emmc.dtb** 设备树拷贝到 Ubuntu 的 tftp 工作目录下。

光盘路径: 开发板光盘 A-基础资料\8、系统镜像\2、教程系统镜像\2、kernel 镜像
赋予这两个文件可读可写可执行权限。

```
sudo chmod 777 zImage imx6ull-alientek-emmc.dtb
```

最终 tftp 工作目录如下。

```
alientek@ubuntu16:~/linux/tftp$ sudo chmod 777 zImage imx6ull-alientek-emmc.dtb
[sudo] alientek 的密码:
alientek@ubuntu16:~/linux/tftp$ ls
imx6ull-alientek-emmc.dtb  test.c  zImage
alientek@ubuntu16:~/linux/tftp$
```

图 4.1.2-1 将内核、设备树放入 tftp 工作目录

4.1.3 文件系统准备

在 nfs 共享目录下创建一个 rootfs 目录用于存放文件系统。

```
mkdir rootfs
```

将光盘里的教程 busybox 文件系统 **rootfs.tar.bz2** 拷贝到 Ubuntu 的 nfs 共享目录下的 rootfs 文件夹中。

光盘路径: 开发板光盘 A-基础资料\8、系统镜像\2、教程系统镜像\3、文件系统\2、busybox-1.29.0 根文件系统\rootfs.tar.bz2

拷贝完的 nfs 共享目录如下。

```
alientek@ubuntu16:~/linux/nfs/rootfs$ ls
rootfs.tar.bz2
alientek@ubuntu16:~/linux/nfs/rootfs$
```

图 4.1.3-1 nfs 共享目录下的文件系统包

执行以下指令将文件系统包解压。

```
tar vxjf rootfs.tar.bz2
```

解压完的文件系统如下。

```
alien@ubuntu16:~/linux/nfs/rootfs$ ls
bin      etc      include  minicom.log  proc      sbin      tmp      video
dev      gdbtest  lib      mnt          root      share     usr      中文测试
drivers  home     linuxrc  music        rootfs.tar.bz2  sys      var
```

图 4.1.3-2 解压完的文件系统

4.2 TFTP 挂载内核、设备树

这里将教程的 uboot 烧录到 EMMC 中,如果是烧到 SD 卡,拨码开关需要改到 SD 卡启动。

这里开发板拨码开关 EMMC 模式启动,进入 uboot 命令行,设置开发板的 IP 信息。

为防止可能之前修改过环境变量而导致无法挂载,我们可以使用 `env default -a` 指令先清除环境变量。

```
env default -a;saveenv
setenv ipaddr 192.168.10.50
setenv ethaddr 00:04:9f:04:d2:35
setenv gatewayip 192.168.10.1
setenv netmask 255.255.255.0
setenv serverip 192.168.10.100
saveenv
```

```
Error: FEC1 address not set.

Normal Boot
Hit any key to stop autoboot: 0
=> env default -a;saveenv
## Resetting to default environment
Saving Environment to MMC...
Writing to MMC(1)... done
=> setenv ipaddr 192.168.10.50
=> setenv ethaddr 00:04:9f:04:d2:35
=> setenv gatewayip 192.168.10.1
=> setenv netmask 255.255.255.0
=> setenv serverip 192.168.10.100
=> saveenv
Saving Environment to MMC...
Writing to MMC(1)... done
=>
```

图 4.1.3-1 设置开发板网络信息

测试下 uboot 是否能 ping 虚拟机。

```
ping 192.168.10.100
```

```
=> ping 192.168.10.100
Using FEC1 device
host 192.168.10.100 is alive
```

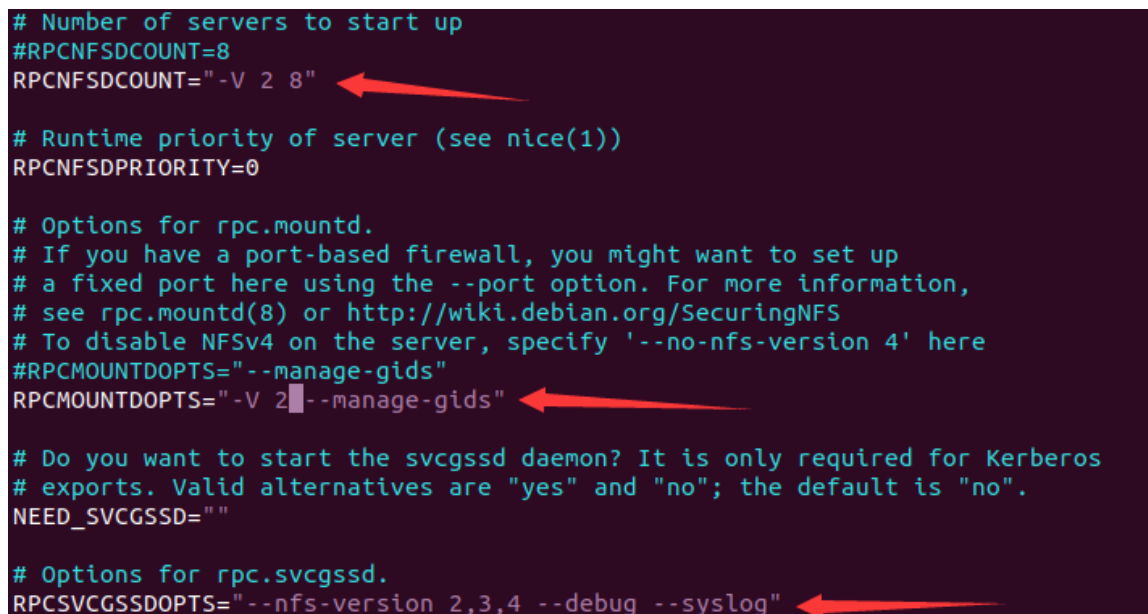
图 4.1.3-2 uboot 阶段 ping 虚拟机

注意,uboot 阶段,虚拟机 ping 开发板是没有反馈的,这里不用虚拟机去 ping 开发板。

设置环境变量 `bootcmd` 来挂载 tftp 目录下的内核、设备树文件到开发板内存中。

这样我们成功挂载了教程系统,使用同样的方法可以挂载出厂的文件系统和其他文件系统,这里就不重复了。这里我们是体验挂载教程系统,学习驱动要结合教程的内核、要修改设备树,所以建议大家还是要学会编译内核,加载自己的 zImage 和设备树。

注意: 以上挂载的是 Ubuntu16 系统下的共享目录,如果是挂载 Ubuntu18 系统及更高版本的系统下的 nfs 共享目录,uboot 无法通过 nfs 启动 Ubuntu 系统内的共享目录。需要在 /etc/default/nfs-kernel-server 文件进行修改,改好了保存退出,然后重启一下 nfs 就可以了,或者报错 Loading:*ww ERROR:File lookup fail 的也是按照下面的方法来解决。



```
# Number of servers to start up
#RPCNFSDCOUNT=8
RPCNFSDCOUNT="-V 2 8"

# Runtime priority of server (see nice(1))
RPCNFSDPRIORITY=0

# Options for rpc.mountd.
# If you have a port-based firewall, you might want to set up
# a fixed port here using the --port option. For more information,
# see rpc.mountd(8) or http://wiki.debian.org/SecuringNFS
# To disable NFSv4 on the server, specify '--no-nfs-version 4' here
#RPCMOUNTDOPTS="--manage-gids"
RPCMOUNTDOPTS="-V 2 -manage-gids"

# Do you want to start the svcgssd daemon? It is only required for Kerberos
# exports. Valid alternatives are "yes" and "no"; the default is "no".
NEED_SVCGSSD=""

# Options for rpc.svcgssd.
RPCSVCGSSDOPTS="--nfs-version 2,3,4 --debug --syslog"
```

图 4.1.3-4 修改/etc/default/nfs-kernel-server

按照上面截图改完后保存退出,再执行以下指令重启 nfs 服务。

```
sudo service nfs-kernel-server restart
```

然后按照 3.6 小节的方法来测试 NFS 挂载。

如果 TFTP 加载成功,NFS 挂载失败,可以重新仔细检查下前面的配置,特别是 bootargs。有些 NFS 软件版本可能升级了,需要按照 Ubuntu18 的 NFS 配置来操作。有些路由器不能使用 NFS,建议改用开发板直连电脑、电脑 wifi 上网的方式。甚至有一些公司或者学校的网络拦截和权限问题,这个建议换一个环境测试下 NFS。

第五章 附录

5.1 VMware 虚拟网卡问题

5.1.1 不存在虚拟网卡 VMnet1 和 VMnet8

正常来说,我们构建完虚拟机后,在“控制面板\网络和 Internet\网络连接”界面可以看到 VMnet1 和 VMnet8 两个虚拟网卡,如下图所示。

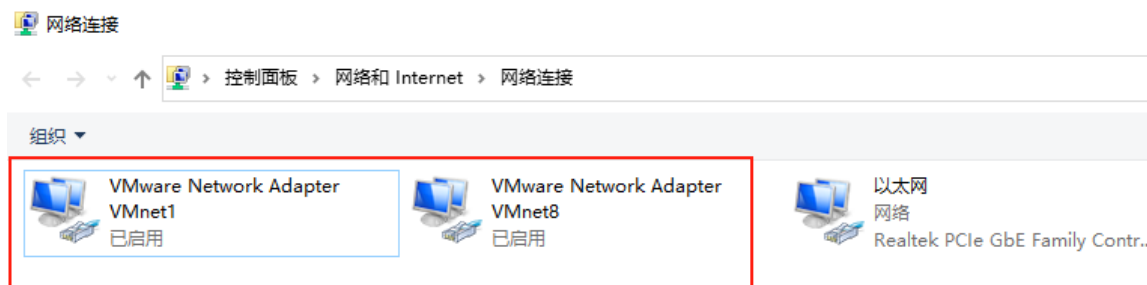


图 5.1.1-1 虚拟网卡

解决思路:

卸载干净 VMware, 重新安装 VMware。

安装 VMware 过程中如果提示 VMCI 相关错误, 如下图报错信息:



图 5.1.1-2 报错信息

解决方法可以参考下百度上的思路,把报错信息输入到百度中查找。可以参考下这个帖子:

<https://tieba.baidu.com/p/6031541992>

驱动有问题话, 可以用驱动精灵更新下驱动试下。

还有一个可能就是 VMware 版本太高了, 可以去 VMware 官网下载 VMware14 或者 VMware12 版本的软件试下。

5.2 双网卡虚拟机无法上网

在 1.4 小节我们设置了虚拟机双网卡, 按照前面的配置, vmnet0 是桥接网卡, vmnet8 是 NAT 网卡。如果按照前面的配置, 重启后出现虚拟机无法上网的情况, 可以参考下面来检查下。

5.2.1 只保留 NAT 网卡就可以上网

如果使用双网卡配置无法上网, 可以试下把桥接网卡关闭, 只保留 NAT 网卡, 然后虚拟机 ping 百度测试看下, 可以多 ping 几次测试。



图 5.2.1-1 断开桥接网卡

那么问题就可能出在 ubuntu 网卡优先级上, 可以尝试参考下面链接的方法来设置两个网卡的优先级。在修改文件前, 可以先拷贝一份要修改的文件做备份, 或者做系统快照, 以防操作失误。

参考链接: <https://www.cnblogs.com/louby/p/10839187.html>

主要就是删除默认的有线网关, 添加 wifi IP 为默认地址, 修改完成后重新打开这两个网卡。

5.2.2 只保留 NAT 网卡也无法上网

参考 1.4 小节的步骤, 检查 NAT 网卡设置、关闭电脑和虚拟机防火墙、DNS 等相关问题。如果电脑是直连网上网的, 虚拟机无法自动获取 IP, 可以尝试手动设置 IP (和电脑 IP 在同一网段, 且不与其他 IP 冲突), DNS 设置为 114.114.114.114 或者 8.8.8.8。或者尝试使用虚拟机网络编辑的还原默认配置。