User's Manual

飞凌嵌入式 FORLINX EMBEDDED

FCU3001

Embedded Development Platform

Ubuntu 18.04.5 LTS

Rev. 1.0

2021/12/03

Forlinx Embedded Technology Co. Ltd. www.forlinx.com

更新记录

日期	底板 版本	核心板 版本	手册 版本	更新内容
2021.12.03	V1.0		V1.0	FCU3001 软件手册第一版,EMMC 版本

目　录

概 述

本手册以使用户快速熟悉产品，了解接口功能和测试方法为目的，主要讲述了开发板接口功能的测试，烧写镜像方法，以及使用过程中出现的一些问题如何排查。在测试过程中，对一些命令进行了注释，方便用户理解，以实用够用为主。

- 本手册一共分为 7 部分：
  - 第一部分产品的整体概述；
  - 第二部分产品的首次开机配置，可采用串口登录和屏幕配置两种方式；
  - 第三部分产品的接口使用；
  - 第四部分产品的 GPU 高性能计算和深度学习；
  - 第五部分产品的多媒体；
  - 第六部分其它配置；
  - 第七部分刷机。
- 格式上本手册中的符号及命令书写做了一下约定

表现形式	含义
⚠	注意或者是需要特别关注的信息，一定要仔细阅读
🕮	对测试章节做的相关说明
🗁	表示相关路径
灰底蓝色字体	指在命令行输入的命令，需要手动输入
灰底黑色字体	输入命令后的串口输出信息
灰底黑色加粗	串口输出信息中的关键信息
//	对输入指令或输出信息的解释内容

⚠ **说明：该产品核心板支持的功能不局限手册中的提到的功能，飞凌仅对手册中的功能项目做测试验证，手册中未说明的功能不予以保证，用户可自行测试验证。**

第一章　FCU3001 简介

1.1 FCU3001 简介

FCU3001 嵌入式控制单元基于 NVIDIA 公司的 Jetson Xavier NX 模块设计，主频最高 1.9GHz，支持最高 21 TOPS 算力。支持 5G 移动网络，千兆网，USB3.0，CAN，RS485，DI，DO 等丰富外设接口，支持 Ubuntu 操作系统。

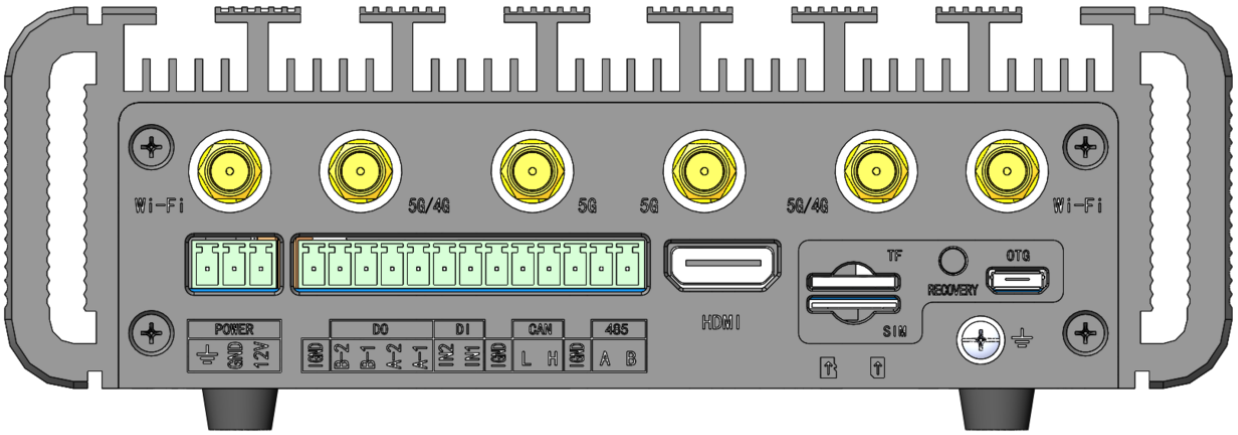
产品特点：

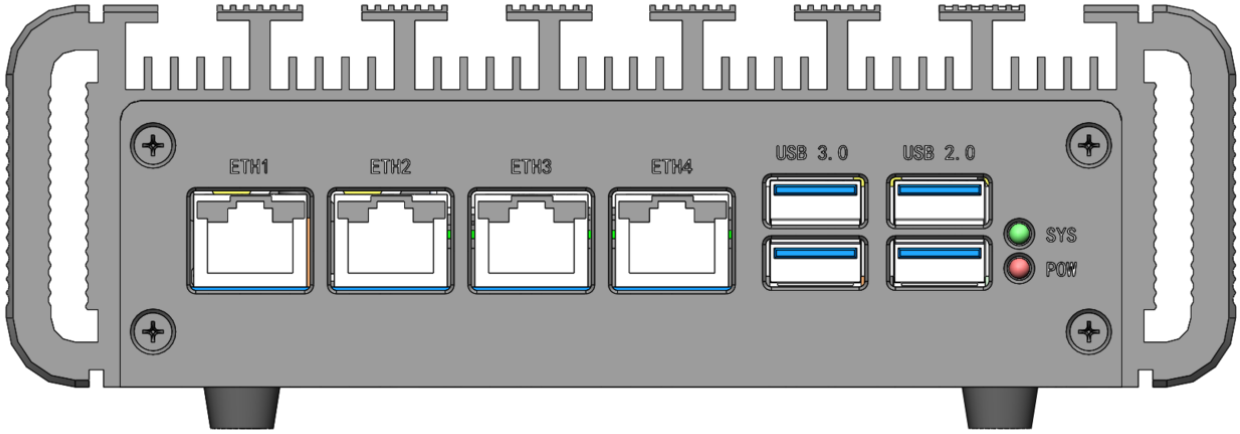
- 6 核 NVIDIA Carmel ARM®v8.2 处理器，最高支持 1.9GHz
- 8G 128bit LPDDR4x，可达 51.2GB/s 的带宽
- 强大的 384 核 NVIDIA Volta™ GPU，可以达到 21TOPS（INT8）的算力
- 具有强大的编解码能力，编码最大 2x4K@30（H265），解码最大 2x4K@60（H265）
- 无风扇设计，系统运行更稳定，温度控制更优秀，壳体密封性更强，适应更多环境情况
- m.2 Type 2230 接口，即 E KEY 标准接口，默认外接 INTEL 3168NGW WIFI 模块；
- M.2 KEY M (PCIex4 NVMe 2280)
- 支持 4G /5G 模组
- Ubuntu 18.04.5 LTS 系统，丰富的第三方应用和插件，方便用户开发。

1.2 应用领域

边缘 AI 计算网关、 智能物联网、智慧城市、工业自动化、视频监控、机器人等应用领域。

1.3 接口示意图


Wi-Fi
5G/4G
5G
5G
5G/4G
Wi-Fi
TF
OTG
RECOVERY
SIM
HDMI
POWER
DO
DI
CAN
485
L H
A B


ETH1
ETH2
ETH3
ETH4
USB 3.0
USB 2.0
SYS
POW

1.3 硬件参数

设备	描述
CPU	6-core NVIDIA Carmel ARM®v8.2 64-bit CPU
GPU	384 NVIDIA® CUDA® cores，48 Tensor cores
RAM	LPDDR4　8GB
ROM	16GB eMMC
移动通信（选配）	**5G 模块：** 移远 RM500U/RM500Q 模块； **4G 模块：** 移远 EM05 模块； 支持全网通； 采用标准 micro SIM 卡槽
开关量输出	2 路，电磁继电器隔离； 触点容量：5A 30VDC / 5A 250VAC 接口：3.81mm 间距绿端子
开关量输入	2 路，2.5KV 光耦隔离，采用 3.81mm 间距绿端子；
RS-485	1 路，3KV 隔离防护，ESD4 级防护，采用 3.81mm 间距绿端子；
CAN	1 路，2.5KV 隔离防护，ESD4 级防护，采用 3.81mm 间距绿端子；
USB	2 个 USB3.0，2 个 USB2.0 接口，采用标准 USB A 型插座，1 个 OTG 接口用于烧写系统，均为 ESD4 级防护
以太网络	4 路千兆网，标准 RJ-45 插座，10M/100/1000M 自适应，ESD4 级防护
HDMI	1 路，HDMI 2.0，支持最高 4K 显示
TF	1 路，支持储存扩展
SSD（选配）	1 路，M.2 PCIex4 2280/2260/2242 KEY M 标准接口
WiFi（选配）	1 路，M.2 PCIex1 2230 KEY E 标准接口
实时时钟	CR2032 电池
恢复按键	1 个，用于系统恢复
电源与功耗	额定电压：12V5A
尺寸	178*110*55mm
安装	可选挂耳，4 只Φ3mm 螺钉
工作环境	湿度：5%～95%，无凝露。 工作温度：0℃～70℃ 存储温度：-20℃～125℃

1.4 资料说明

FCU3001 目前提供了 Ubuntu 18.04.5 LTS 的相关资料。用户可以通过本公司提供的网盘链接获取软件和硬件的文档及源码。资料下载网站为：http://bbs.witech.com.cn/forum-96-1.html，用户需要通过销售获取下载权限方可自行下载。

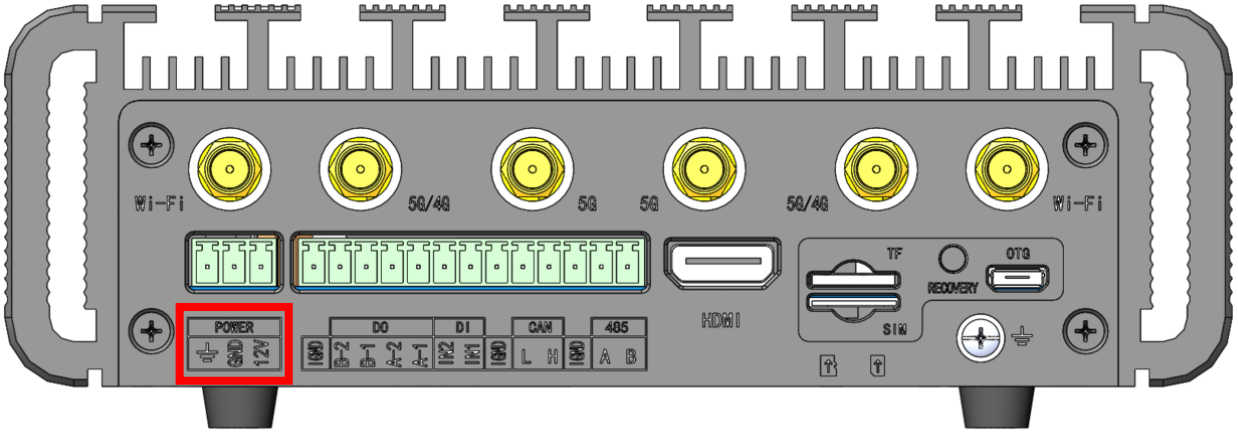
第二章　首次开机

2.1　开机前的准备

FCU3001 开机前硬件准备：

- 12V3A DC 电源
- 网线（如需上网）
- HDMI 线缆和显示器或者 micro USB

电源接口如下如所示：



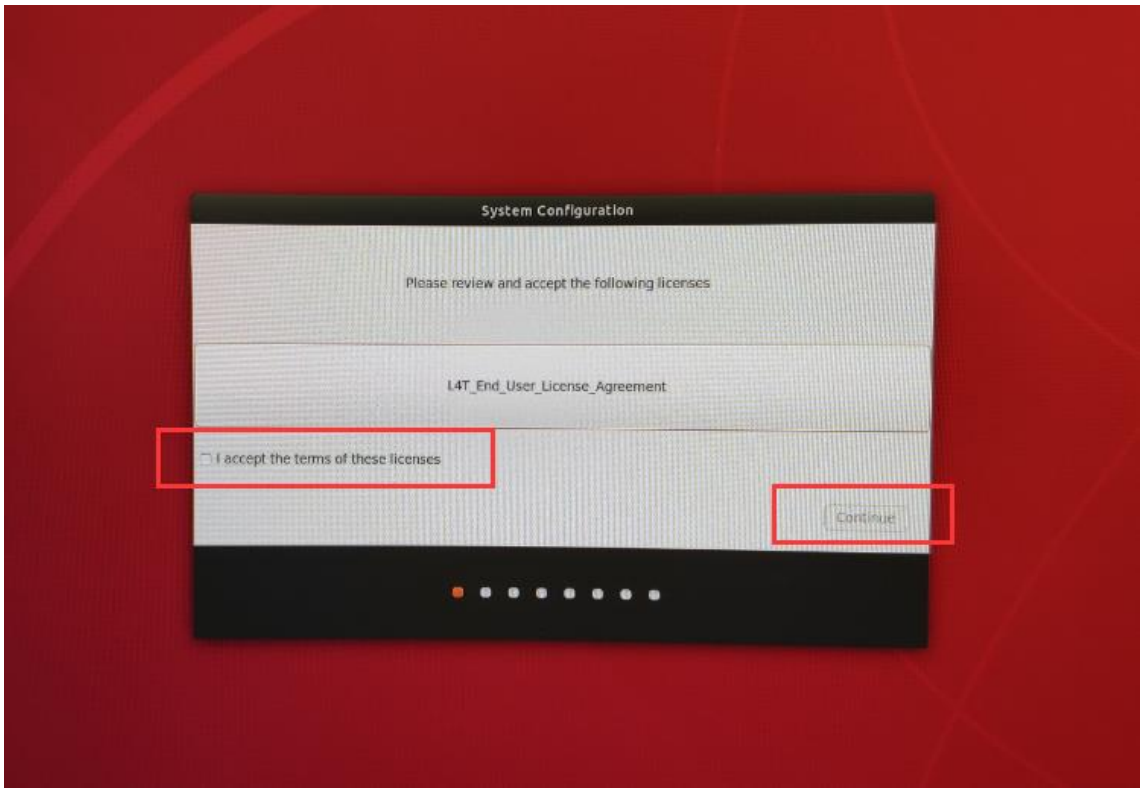
图形化配置或串口配置二选一

2.2　首次开机配置（图形化配置）

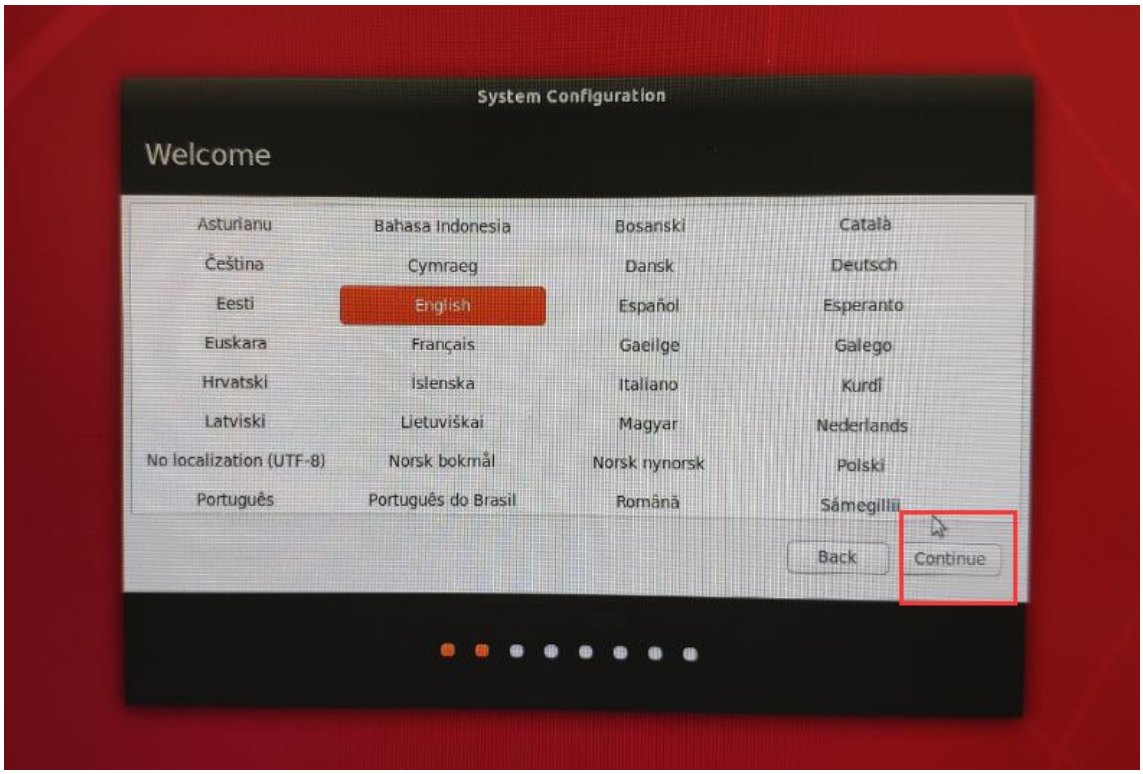
🕮 **说明：**

- **第一次上电开机需要配置用户名和密码，该用户名具有 sudo 权限。**
- **重新刷机后需要重新配置。**
- **出厂因飞凌检测需要，可能已经设置用户名密码为 forlinx**

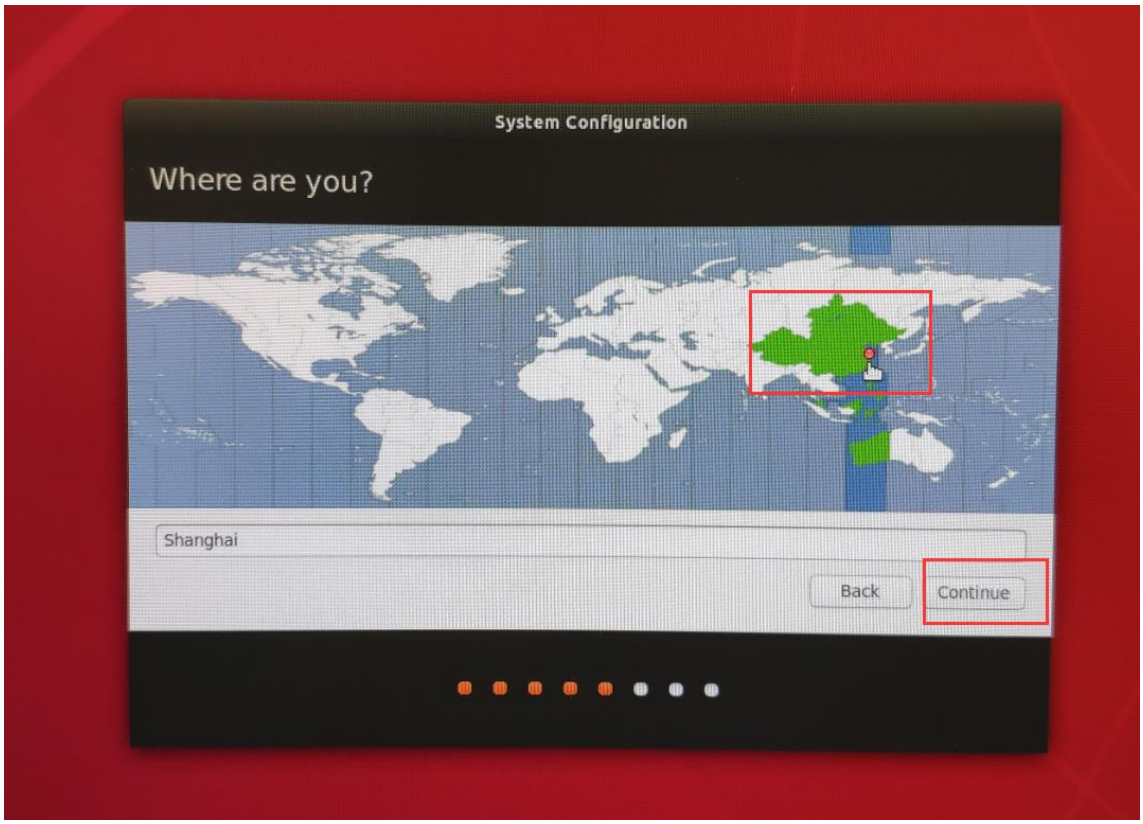
开机上电后 HDMI 显示器上出现的第一个界面如下：



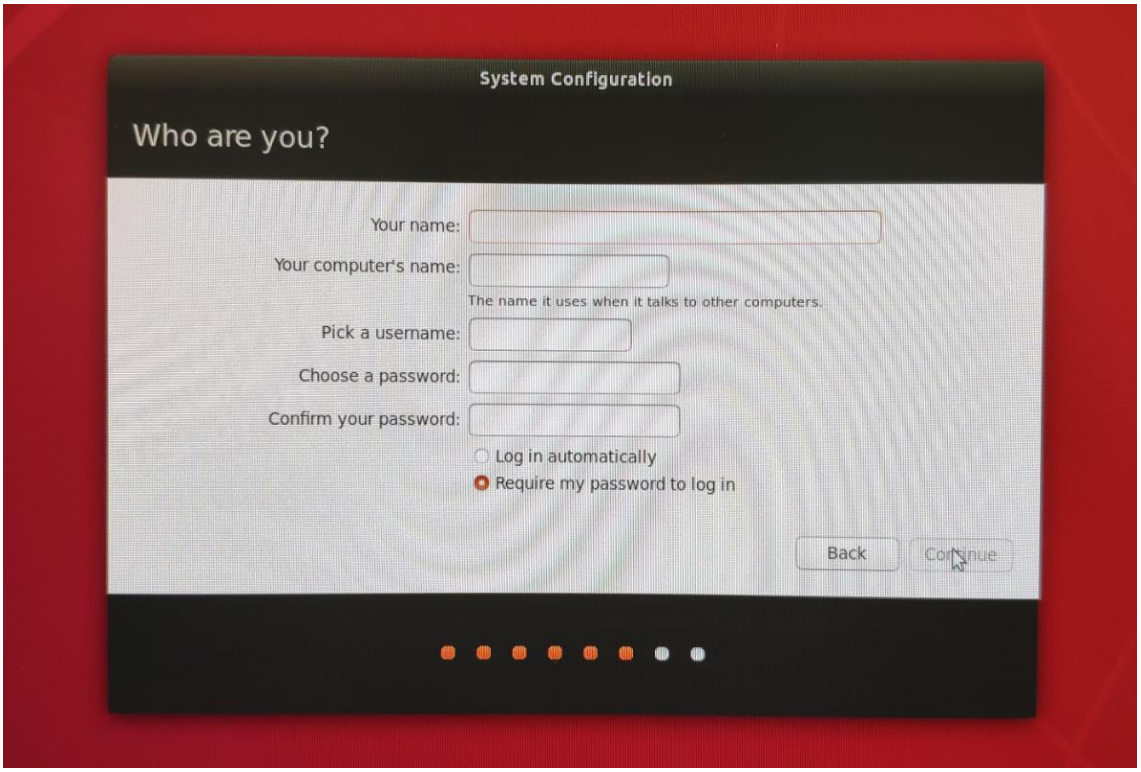
勾选“I accept the terms of these licenses”，然后点击右下“Continue”按钮，进入下一配置界面。



语言选择，按需要选择，一般按照默认即可，点击“Continue”进入下一步

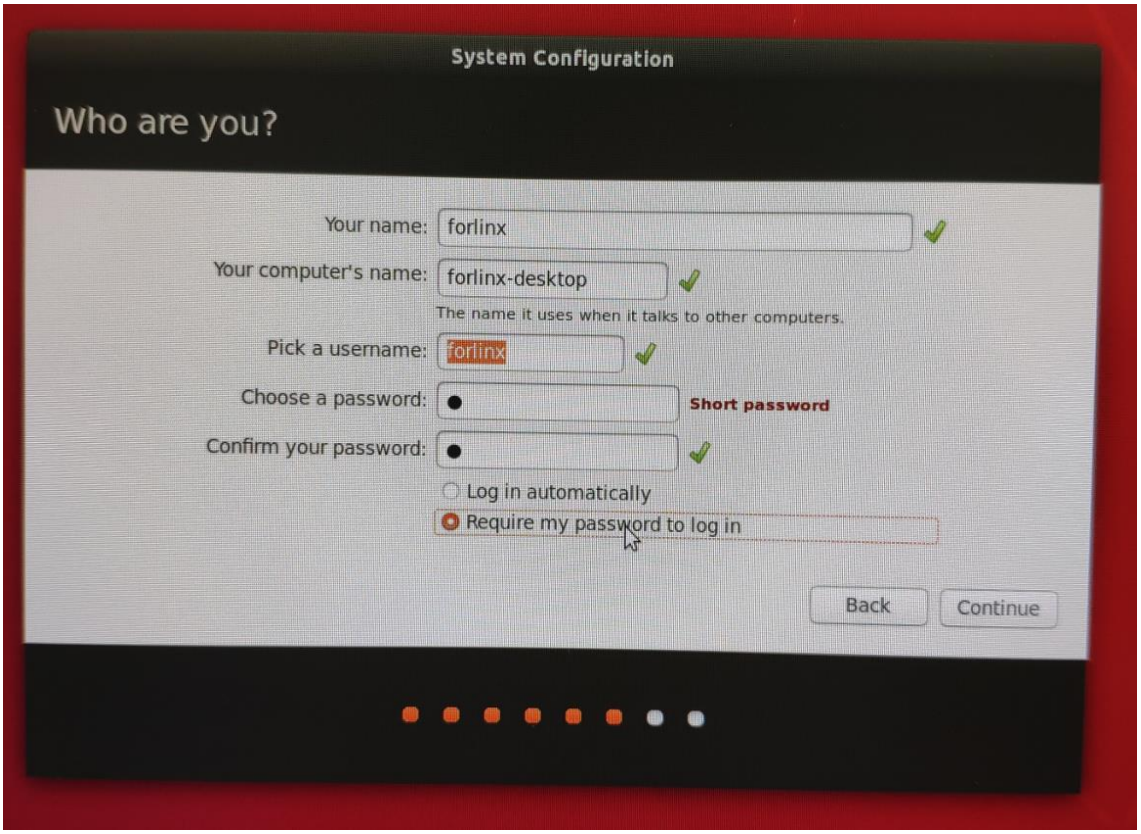


时区选择，根据实际情况选择点击地图，国内大部分地区选择“Shanghai”，然后点击“Continue”

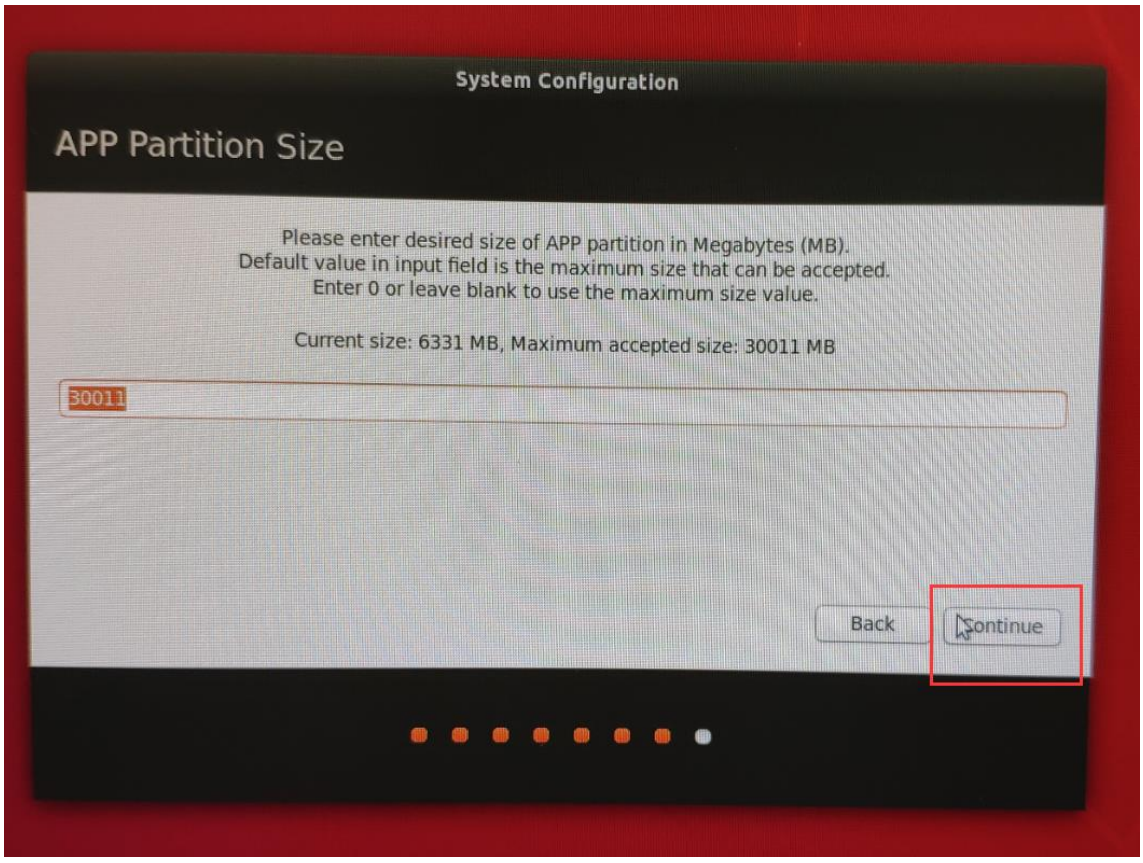


设置初始用户名和密码

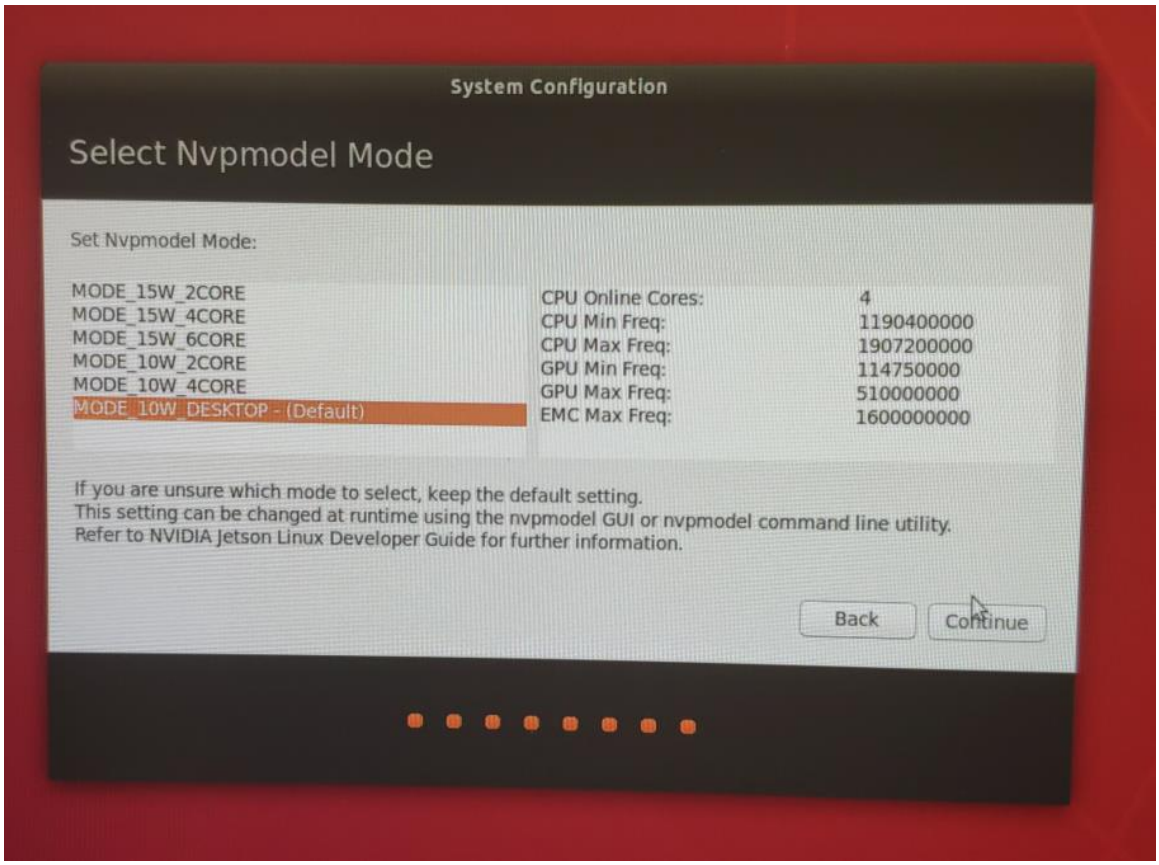
本手册均以“forlinx”为例，参考配置如下：



然后点击“Continue”（OTG 刷机后无此界面）。

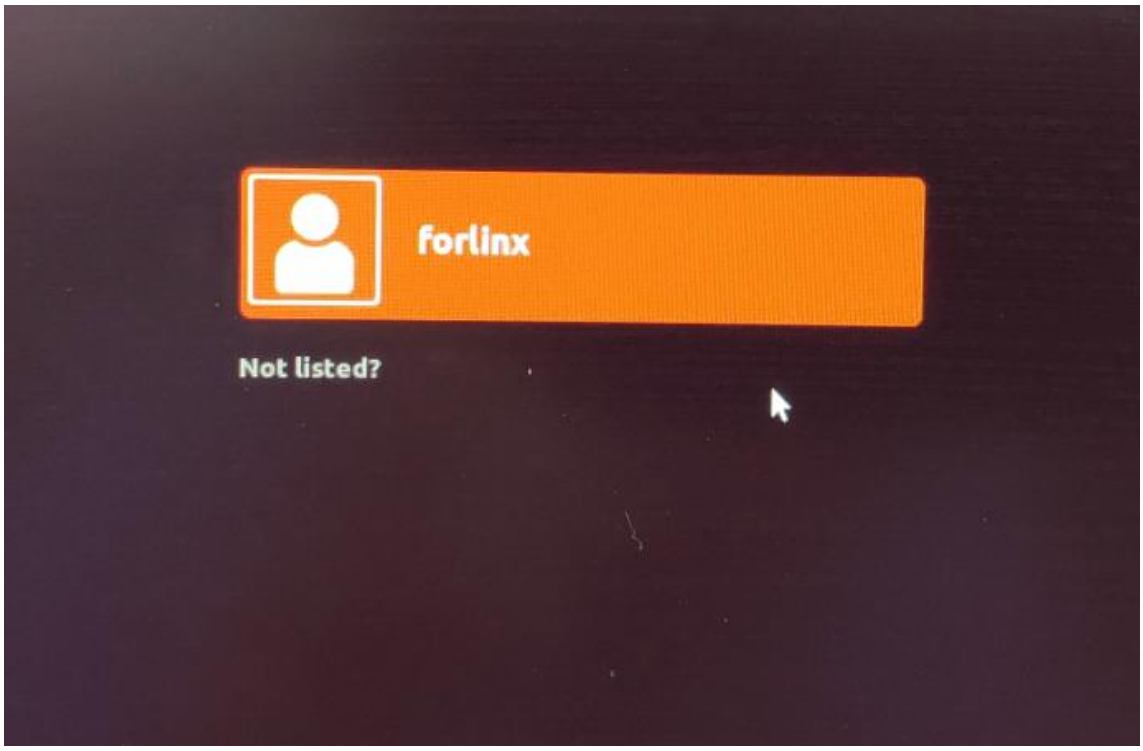


按照默认点击“Continue”

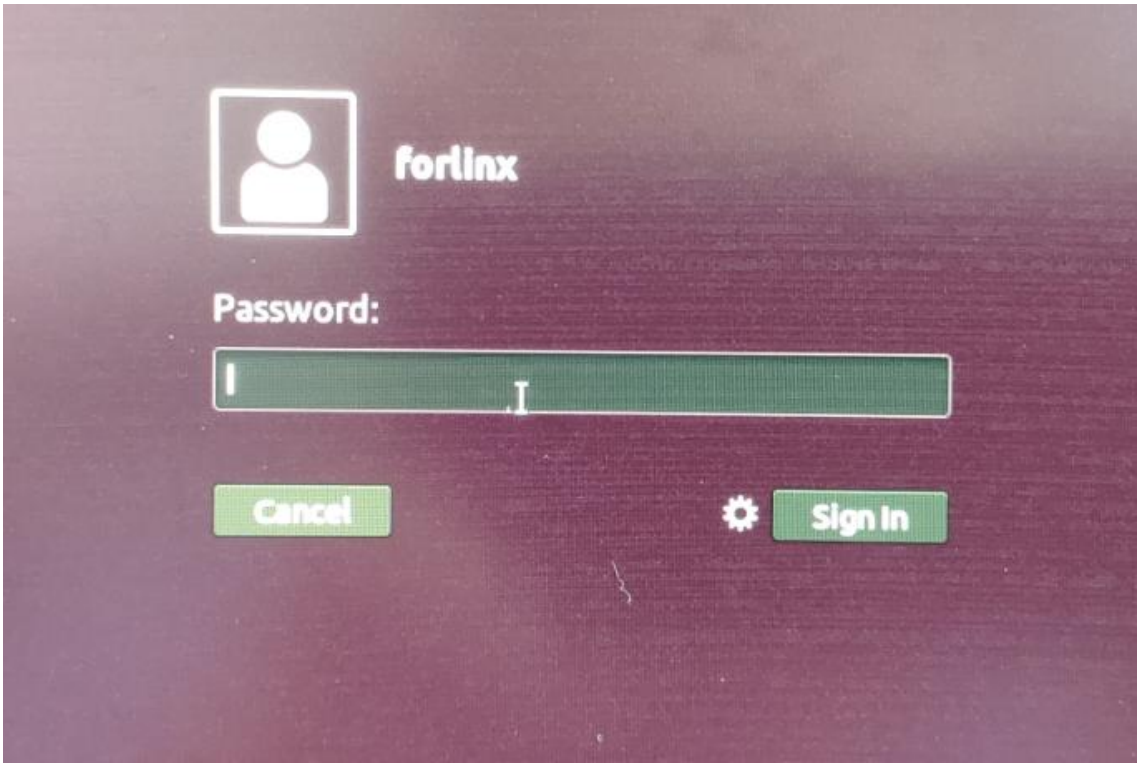


根据场景选着合理的电源管理模型，MODE_15W_6CORE 为最大功率，选择完成后点击“Continue”。

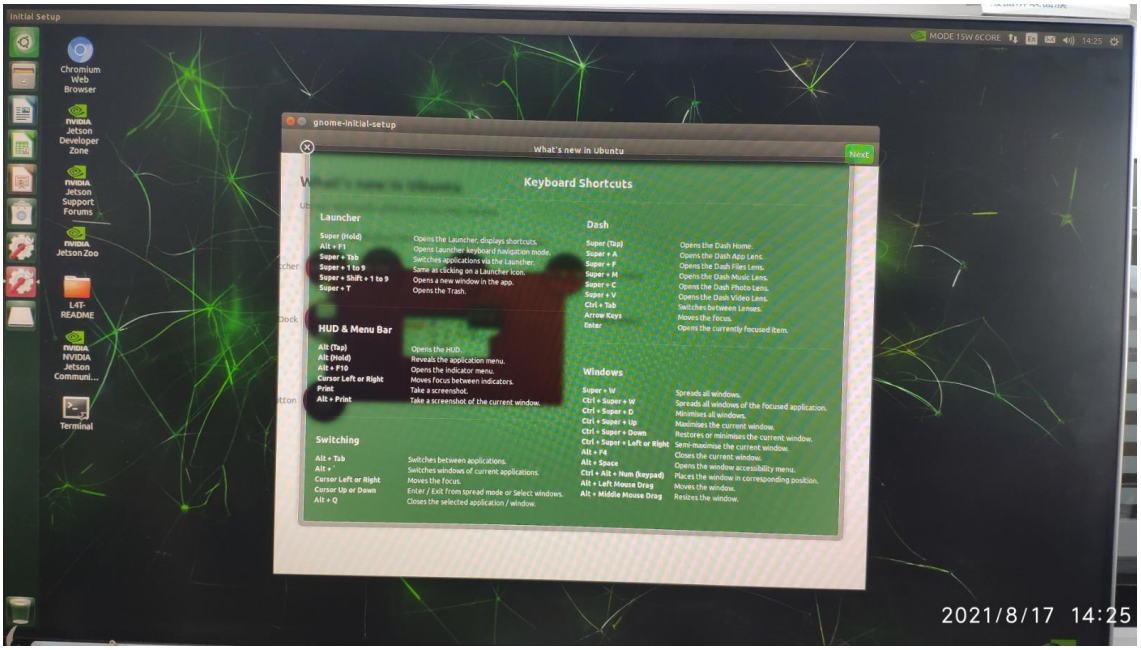
系统进入自动配置过程，配置完成后可弹出登录界面，输入之前设置的用户名和密码即可进入系统。



点击“forlinx”



输入密码后，点击“Sign In”

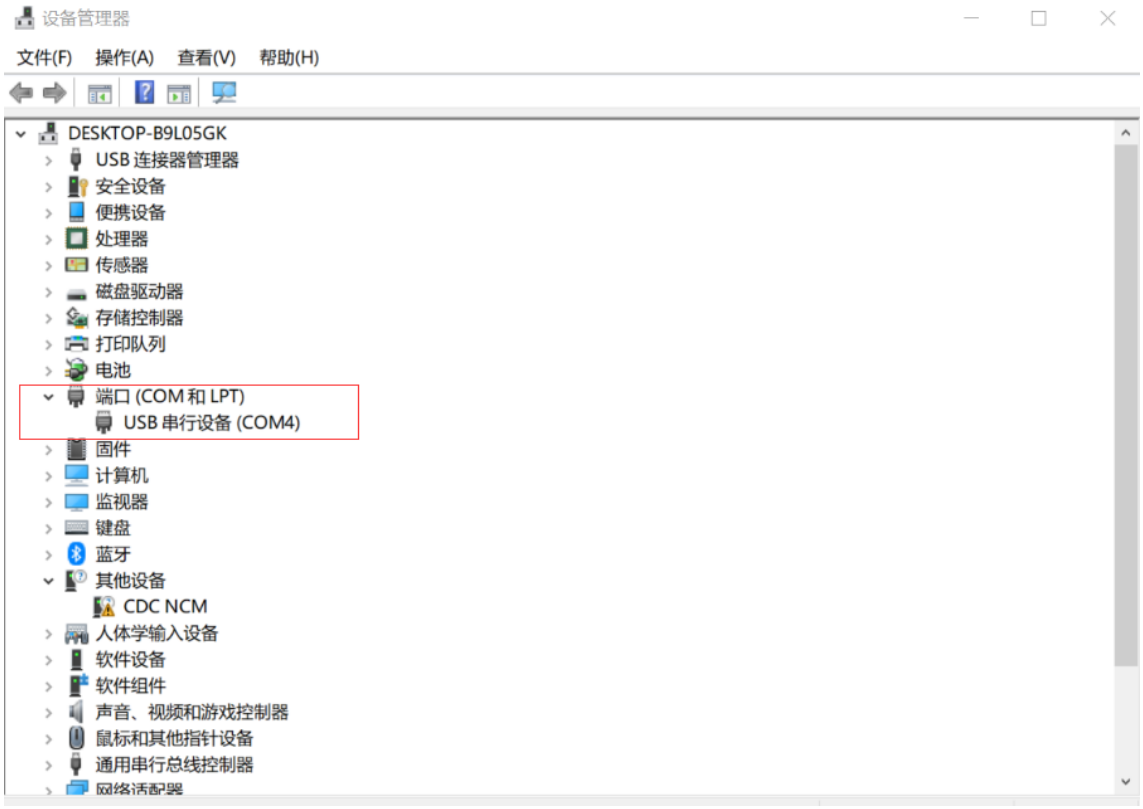


首次进入系统会弹出系统两个操作指引的两个界面，关闭即可。开机配置完成。

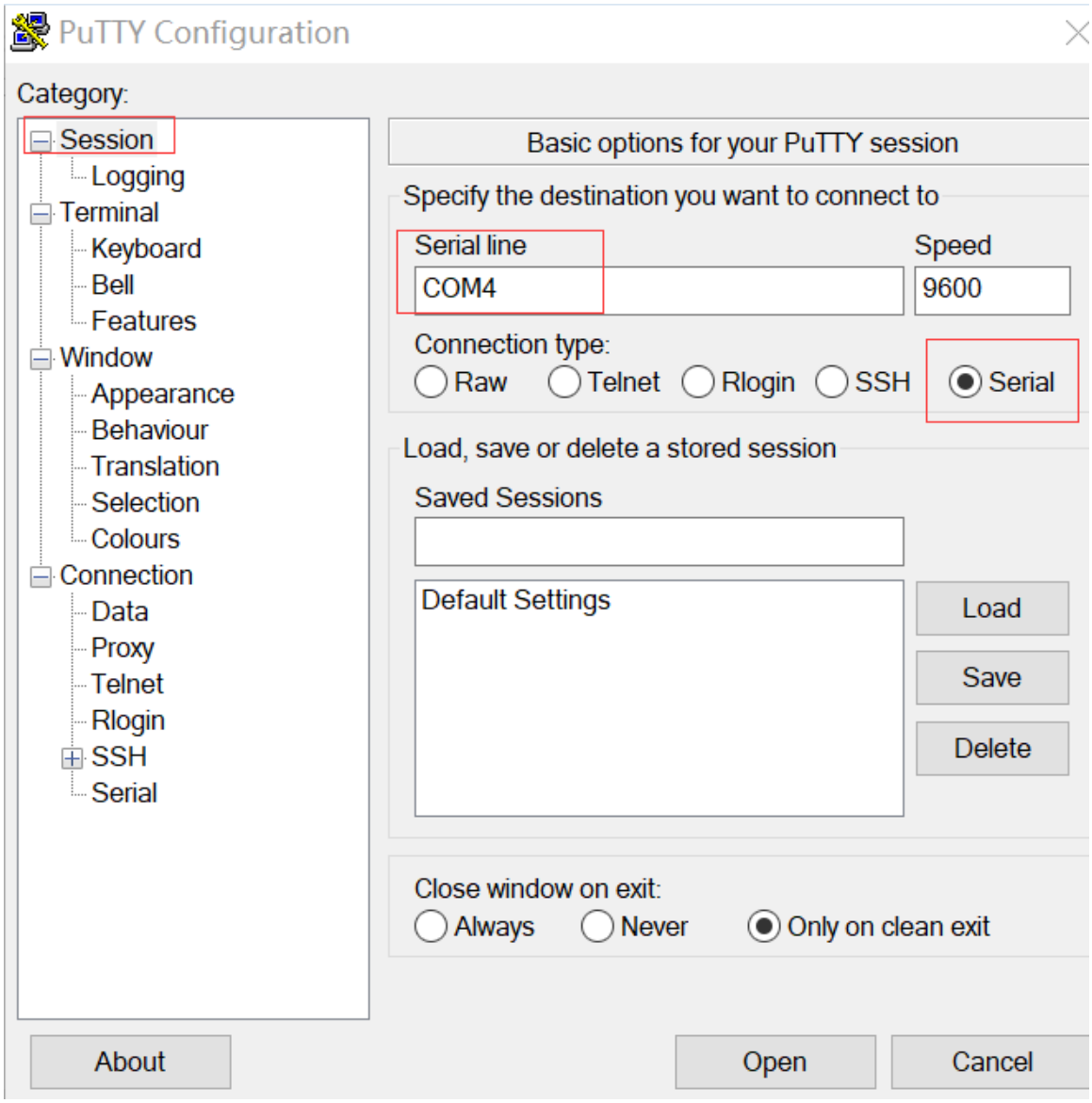
2.3 首次开机配置（串口配置）

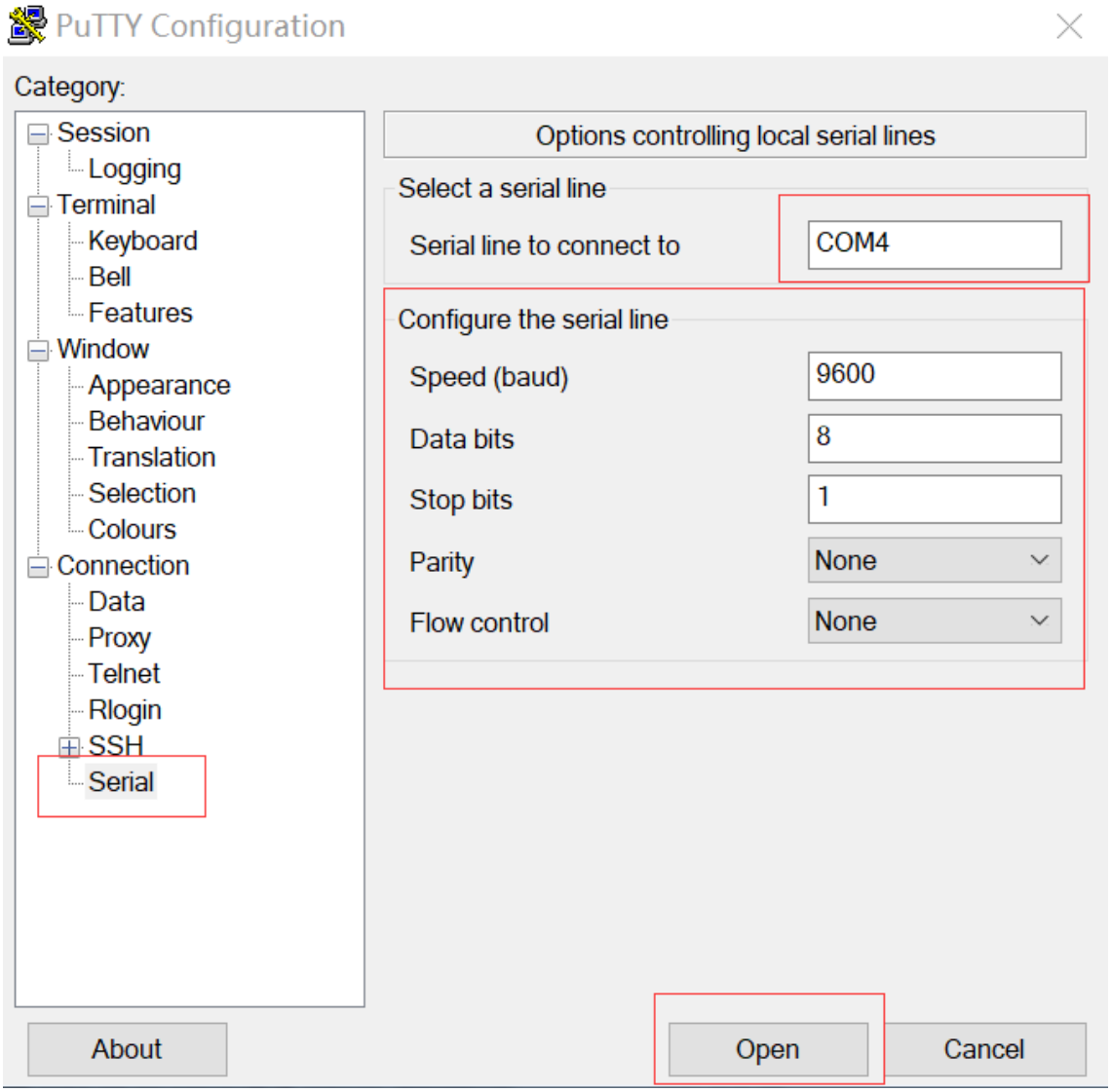
该种配置方式适合在没有显示器的场景下，做第一次上电开机配置。步骤比较多，并且不如图形界面直观，推荐使用上一节“图形配置”。具体步骤如下：

开机上电，将 OTG 口通过 micro USB 线缆连接到 Windows10 电脑。打开设备管理器，等待虚拟串口识别。在设备管理器出现如下图 COM 后，继续下一步操作，COM 口编号随具体电脑和接口变动。

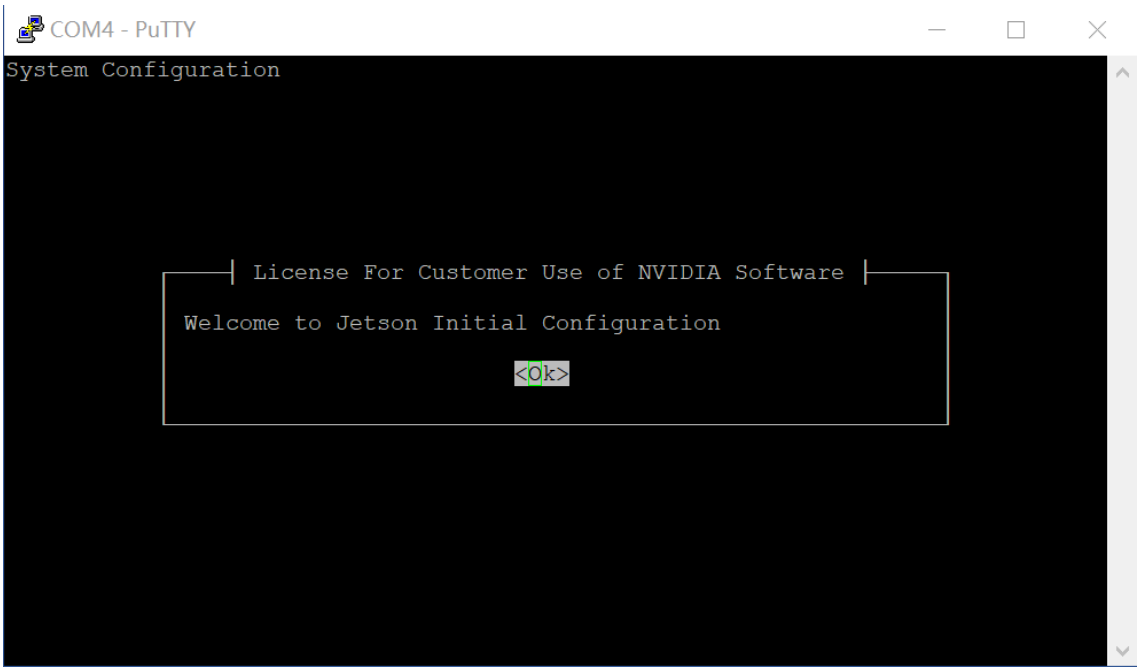


设置 Putty，打开 USB 虚拟串口：

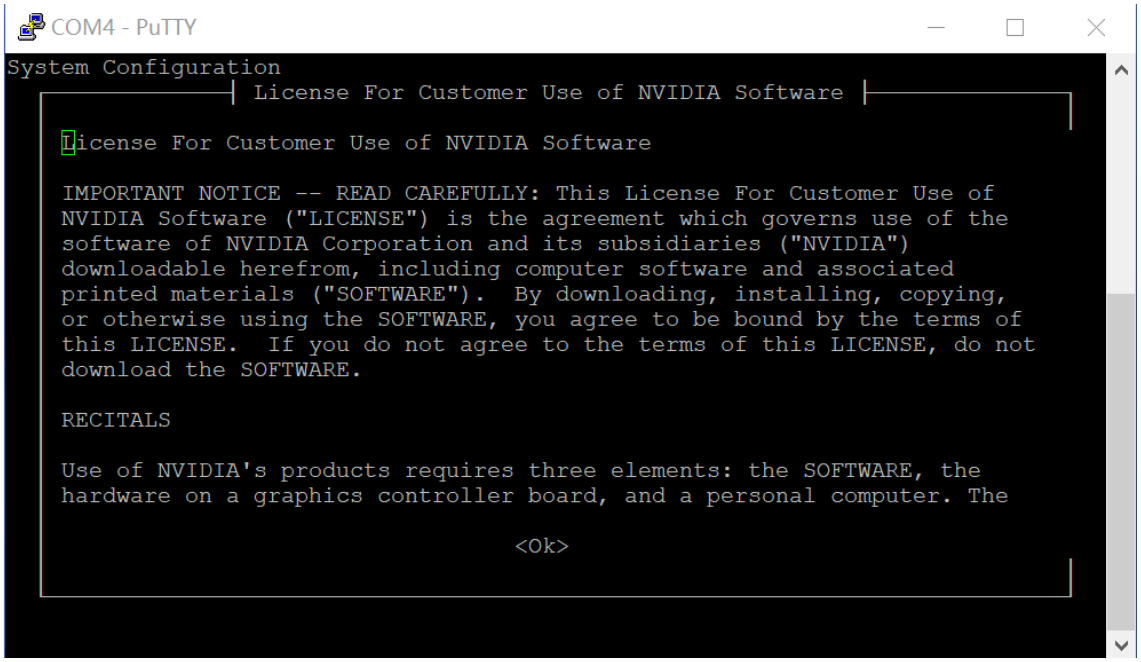

PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Serial
Basic options for your PuTTY session
Specify the destination you want to connect to
Serial line
Speed
COM4
9600
Connection type:
Raw
Telnet
Rlogin
SSH
Serial
Load, save or delete a stored session
Saved Sessions
Default Settings
Load
Save
Delete
Close window on exit:
Always
Never
Only on clean exit
About
Open
Cancel



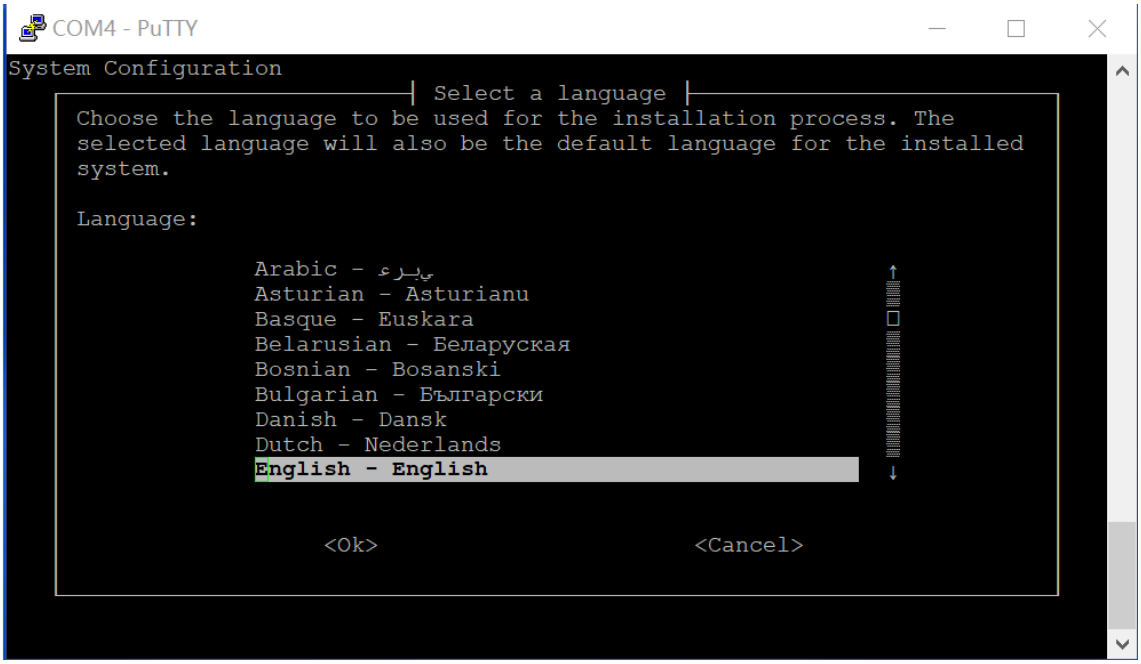
串口打印出系统配置的第一个界面：



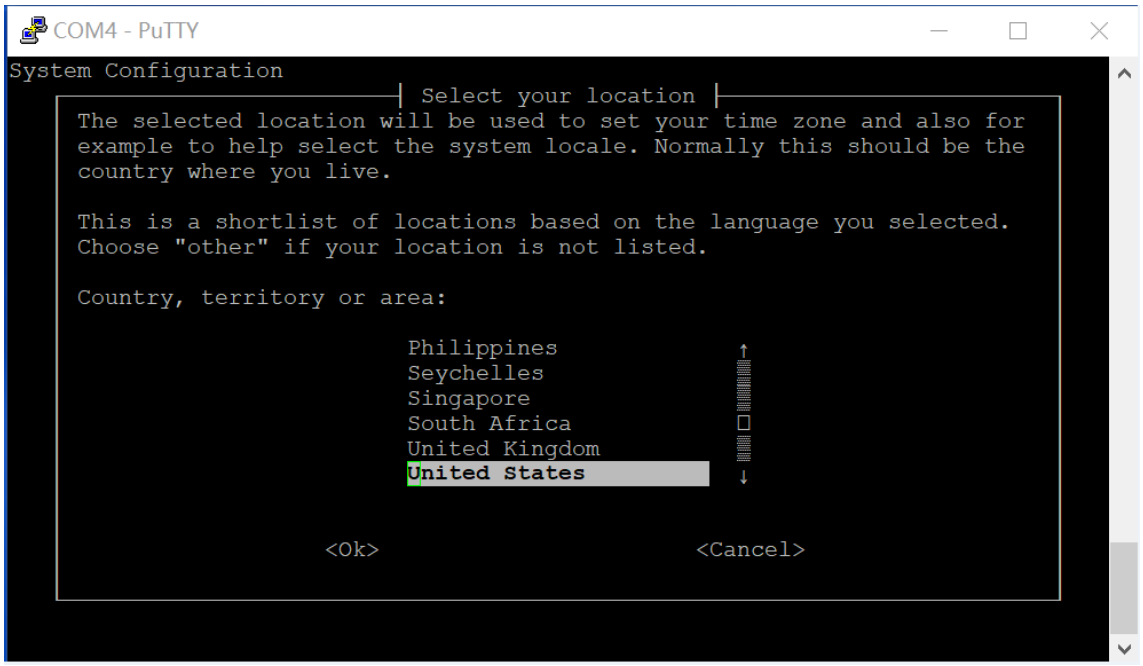
按回车，出现 License 界面：



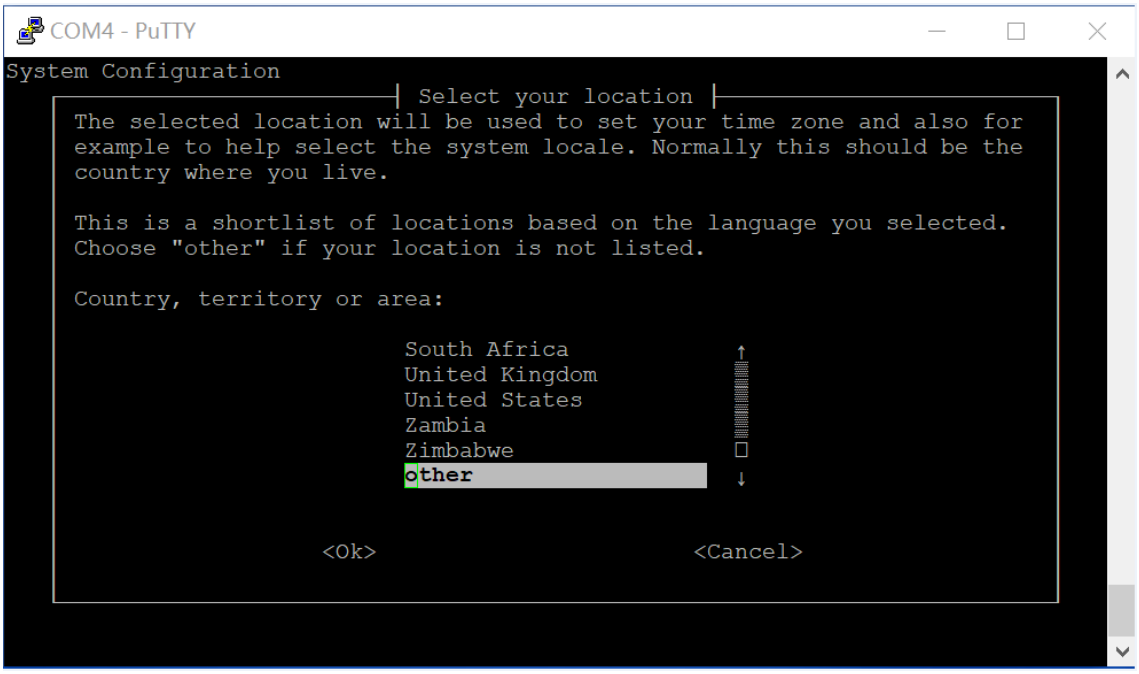
按 ESC 退出 License 界面，出现语言选择界面，根据实际需求选择，这里以英语为例：



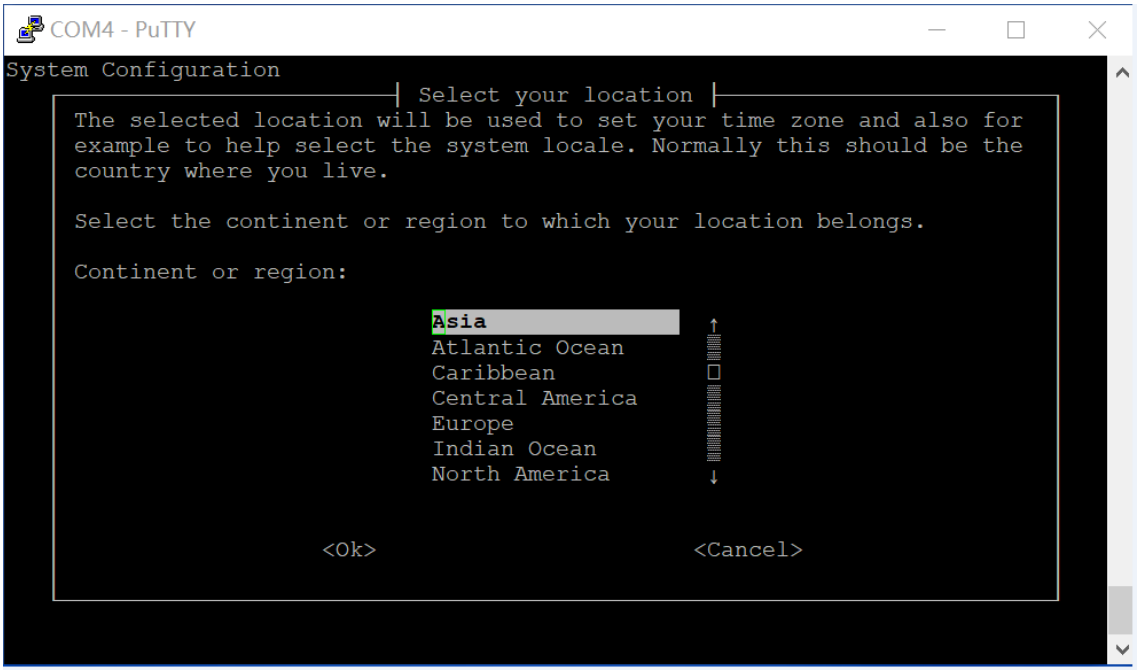
按回车，进入位置选择，位置关系到时区准确性，需要如实选：



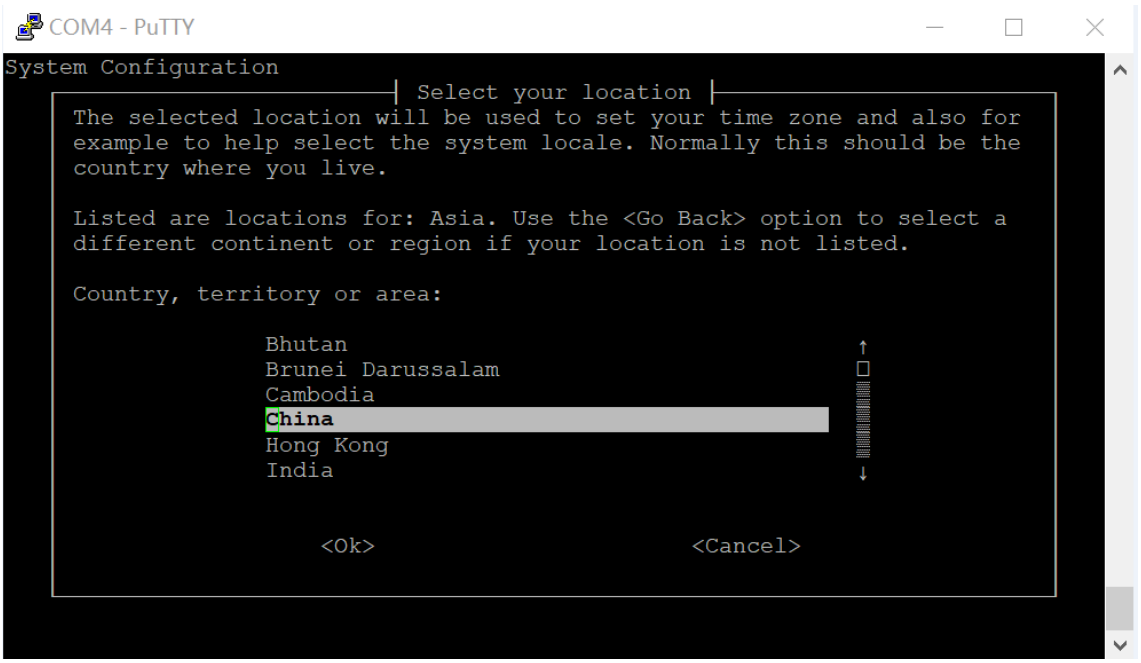
对于中国客户，按 PC 机下键找到最后一个 Other 选项：



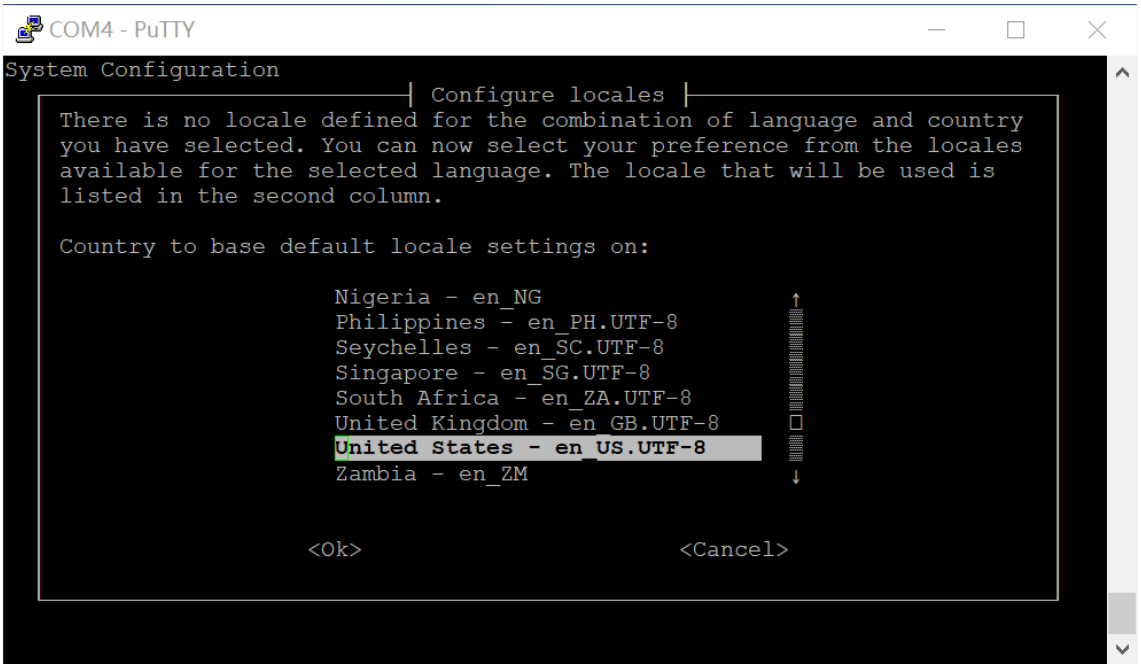
按回车，从新的界面中找到 Asia，亚洲：



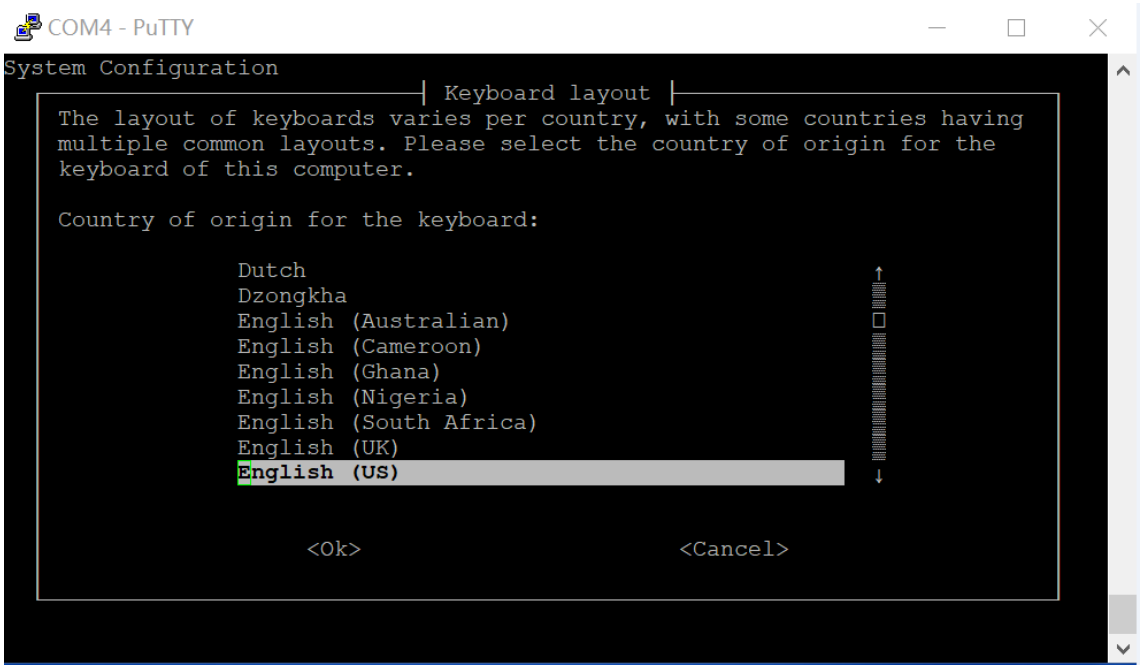
按回车，从新的界面找到 China：



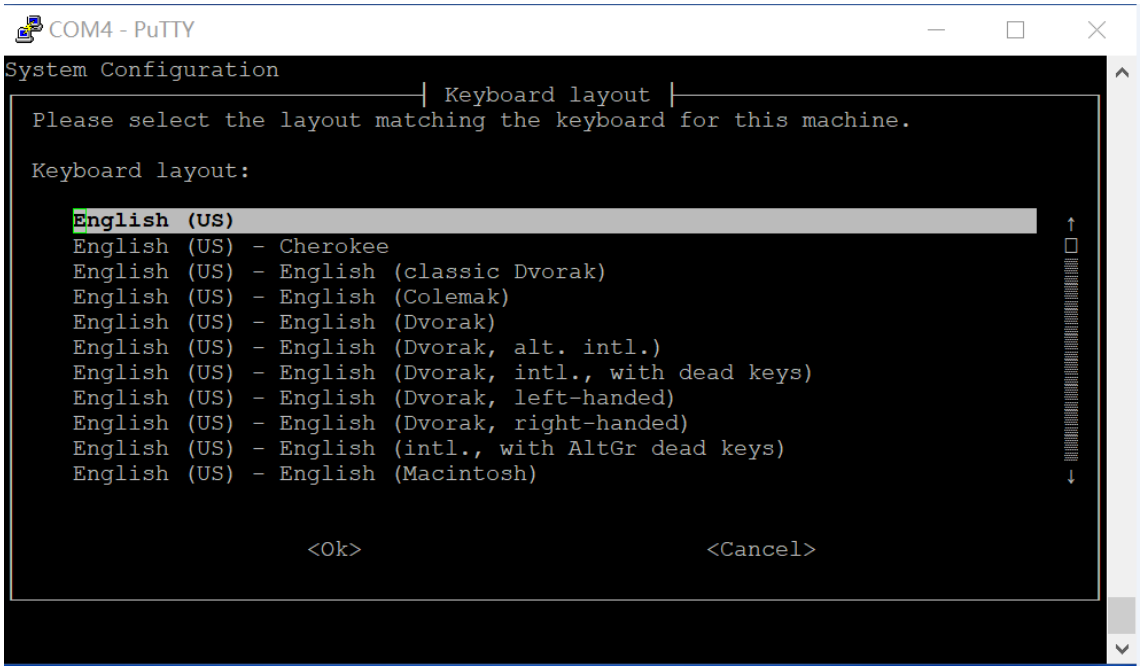
按回车，进入选择字符编码界面：



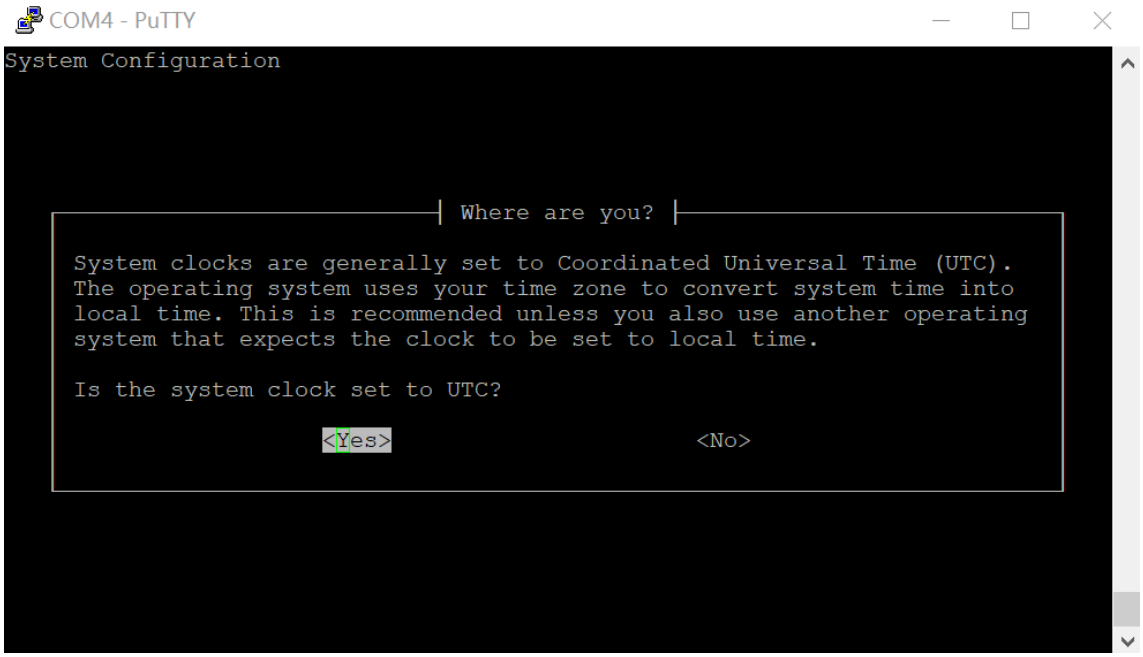
按照默认 en_US.UTF-8，（如果需要显示中文，自行选择 zh_CN），按回车进入键盘布局界面：



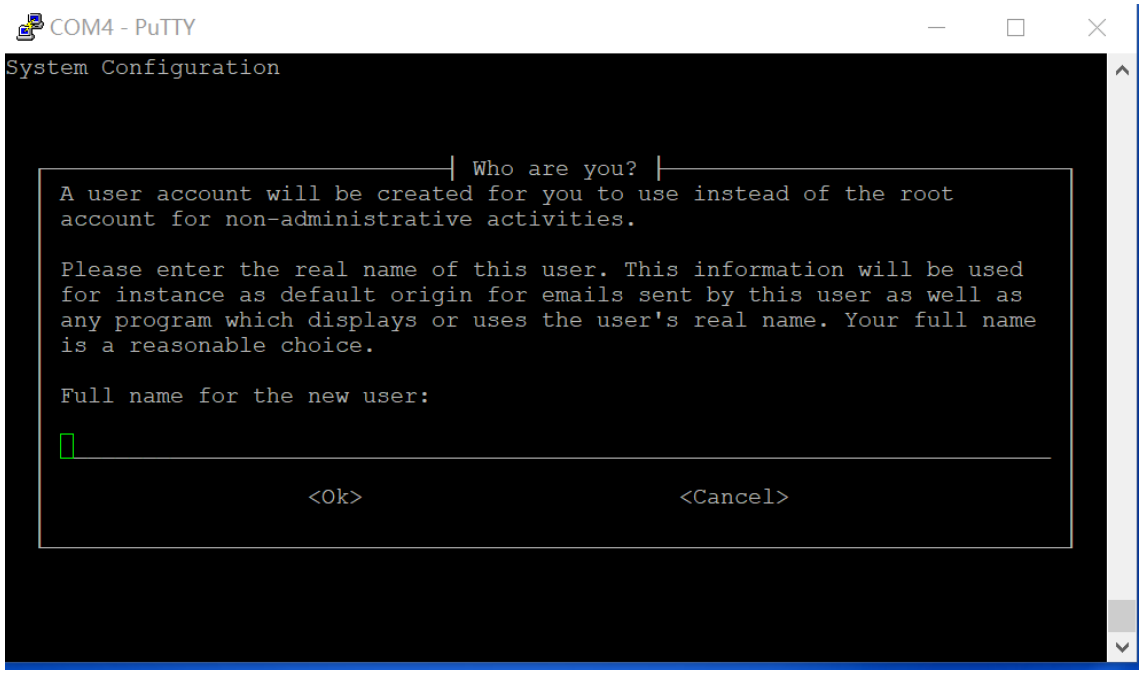
按照默认，按回车进入键盘布局第二个界面：



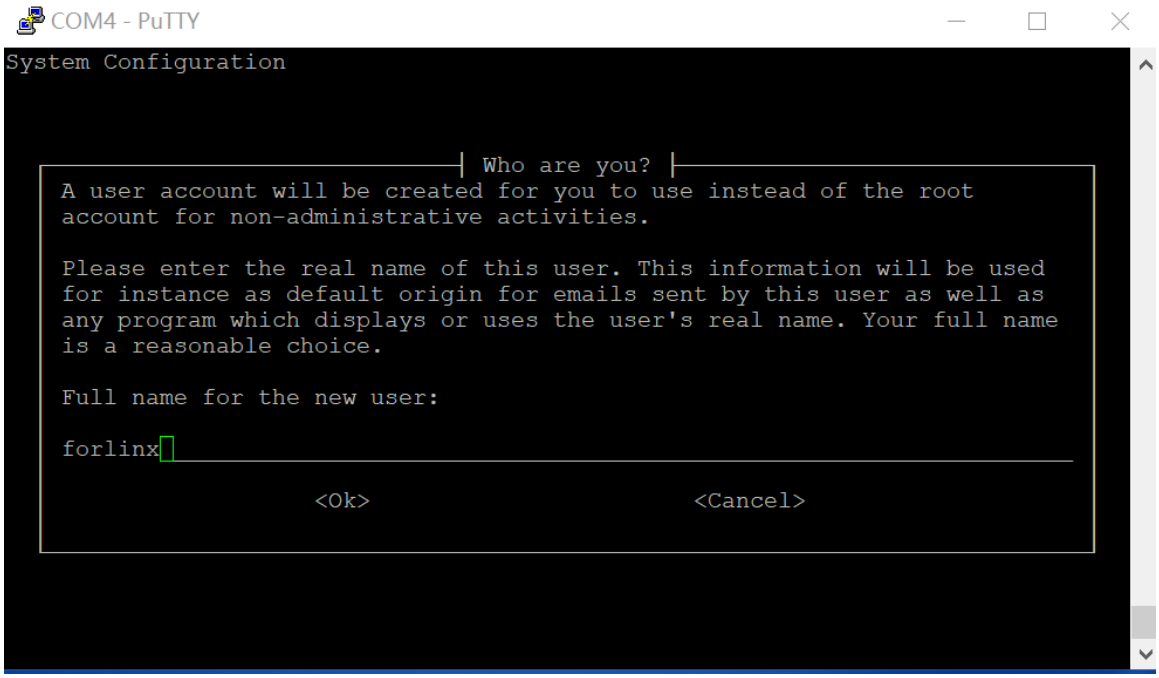
按照默认，按回车，进入是否使能 UTC 时钟界面：



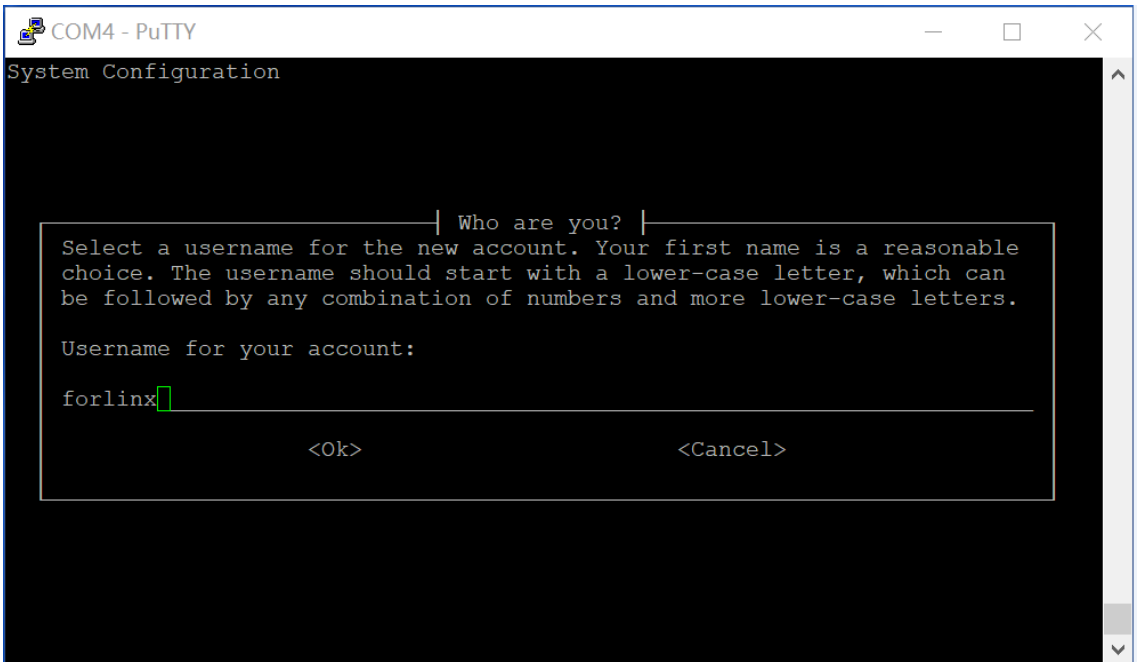
选择 Yes，按回车，进入设置用户名界面：



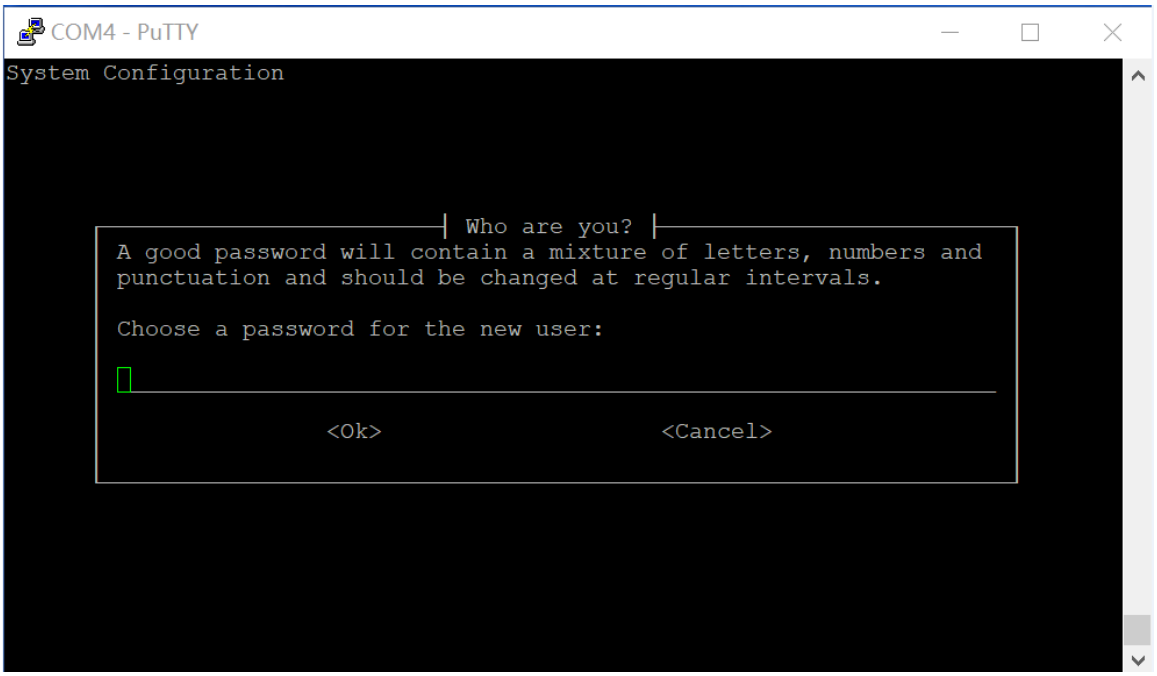
根据具体输入：



按回车，进入登录用户名设置界面，根据实际输入：

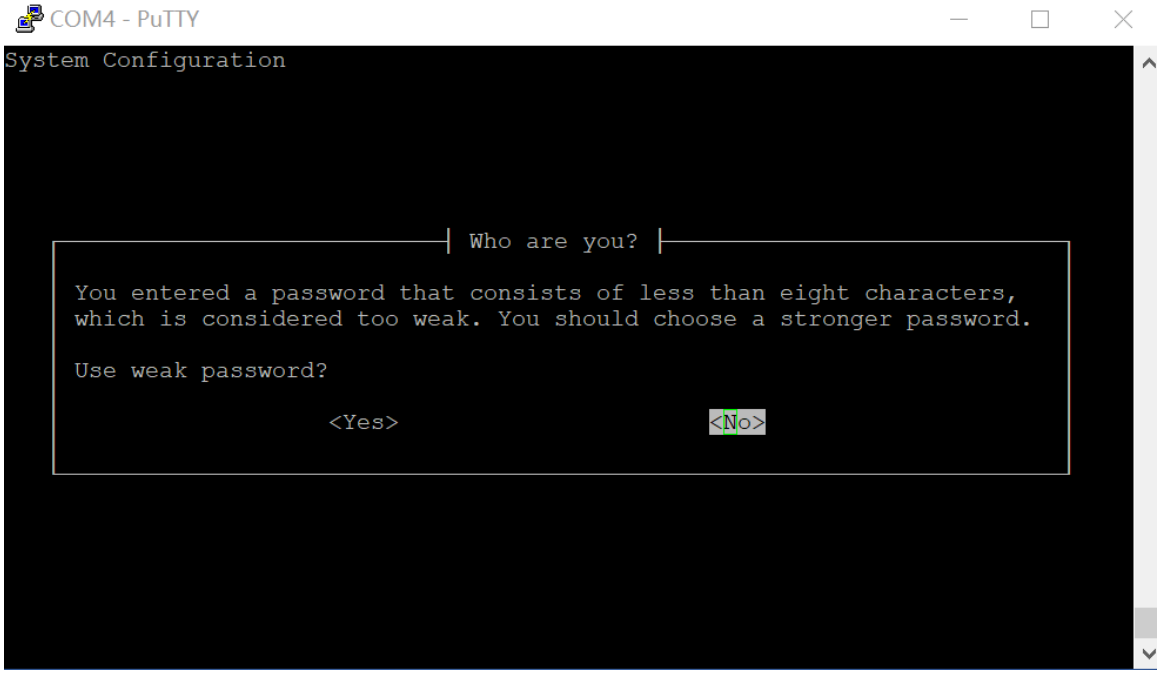


按回车，到输入密码界面：

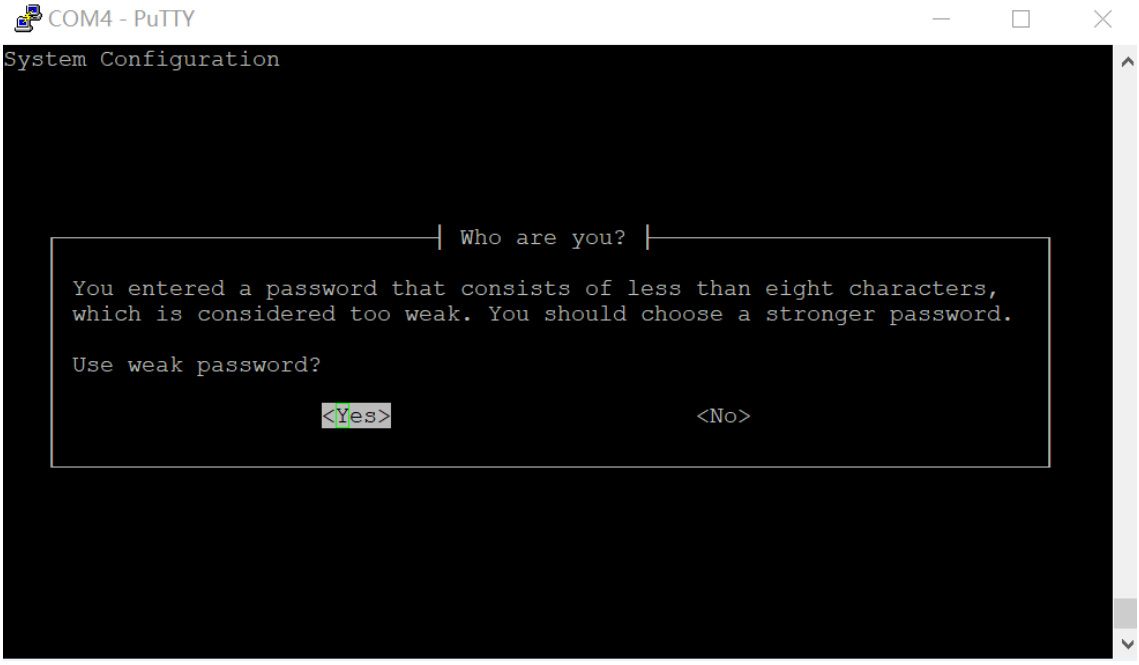


根据实际需要输入密码后，按回车，进入确认密码界面：

再次输入相同密码后，按回车，如果密码设置太简单弹出如下界面：

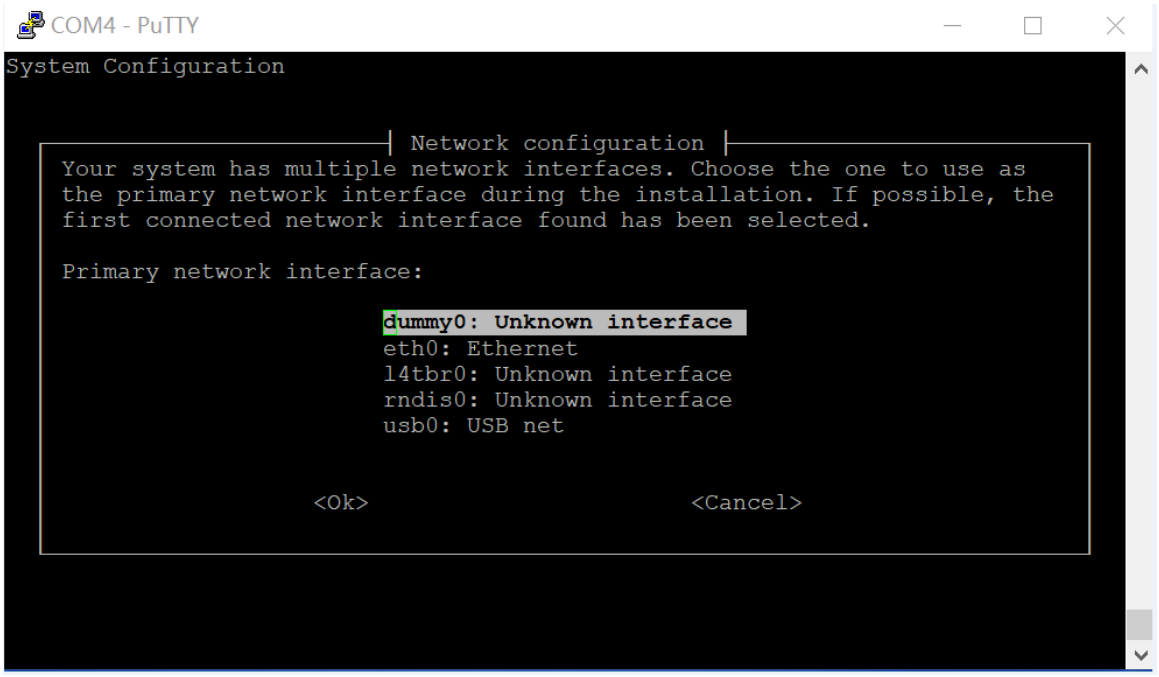


如果依然要保持简单密码，按 PC 键盘左键将光标移动至“Yes”

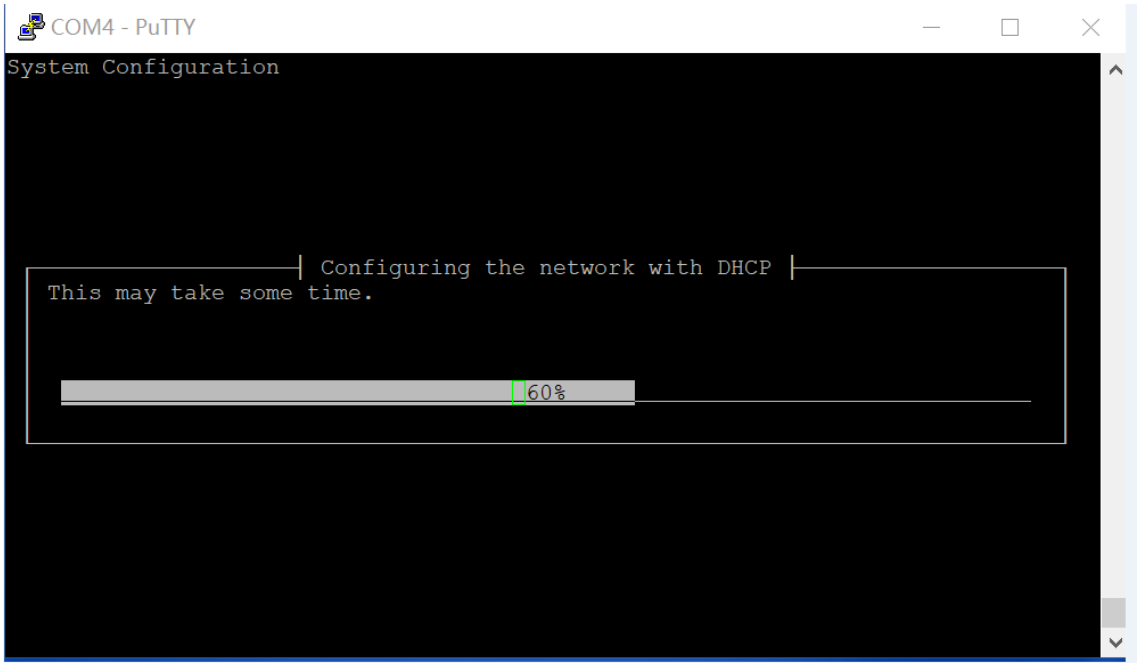


按回车，进入 APP 分区大小设置界面，默认就是可用最大值，不需要修改（如果需要预留空间，这里要减掉预留值。OTG 刷机无此界面）：

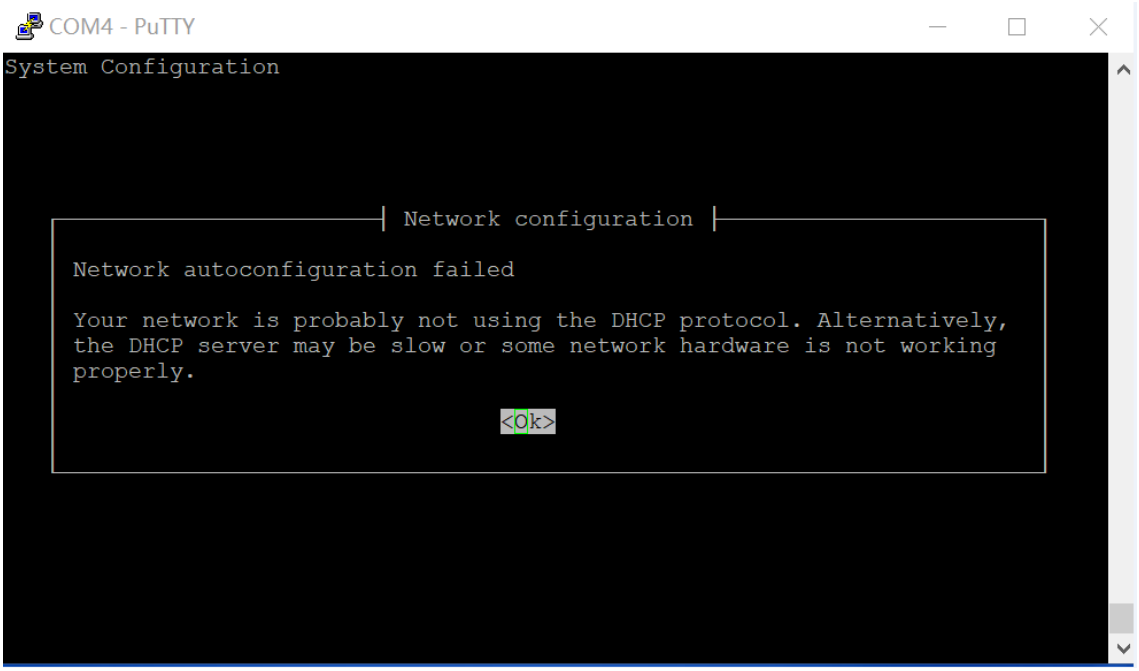
按回车后进入网络配置的网口选择界面：



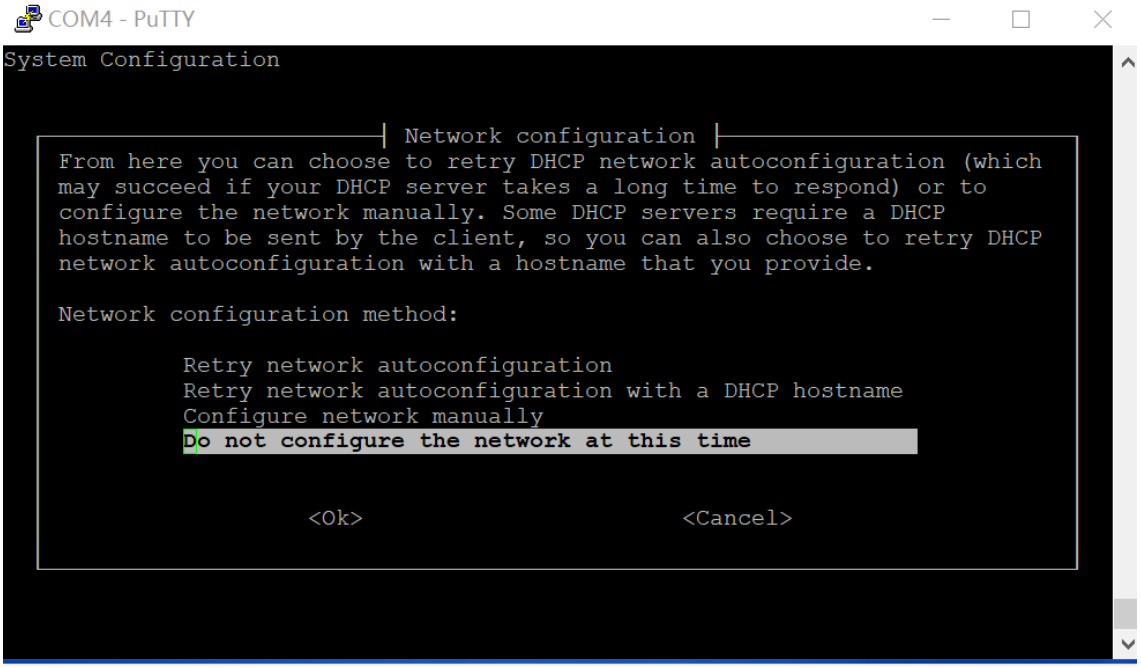
当前不做配置，按照默认，输入回车：



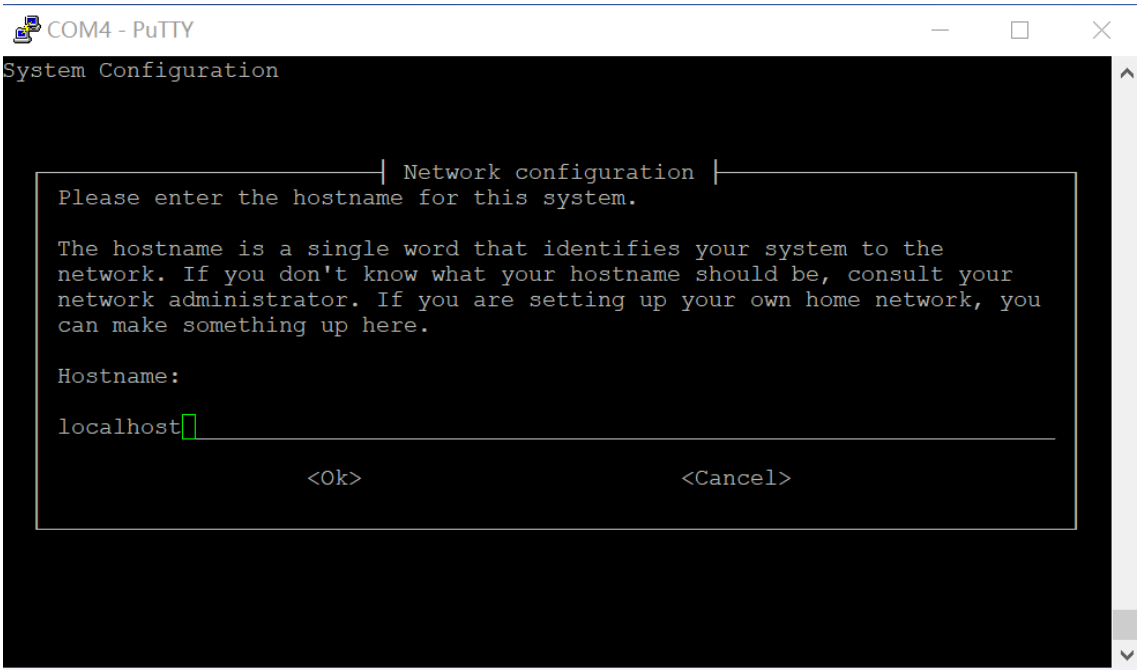
等待网络自动配置失败，弹出如下界面：



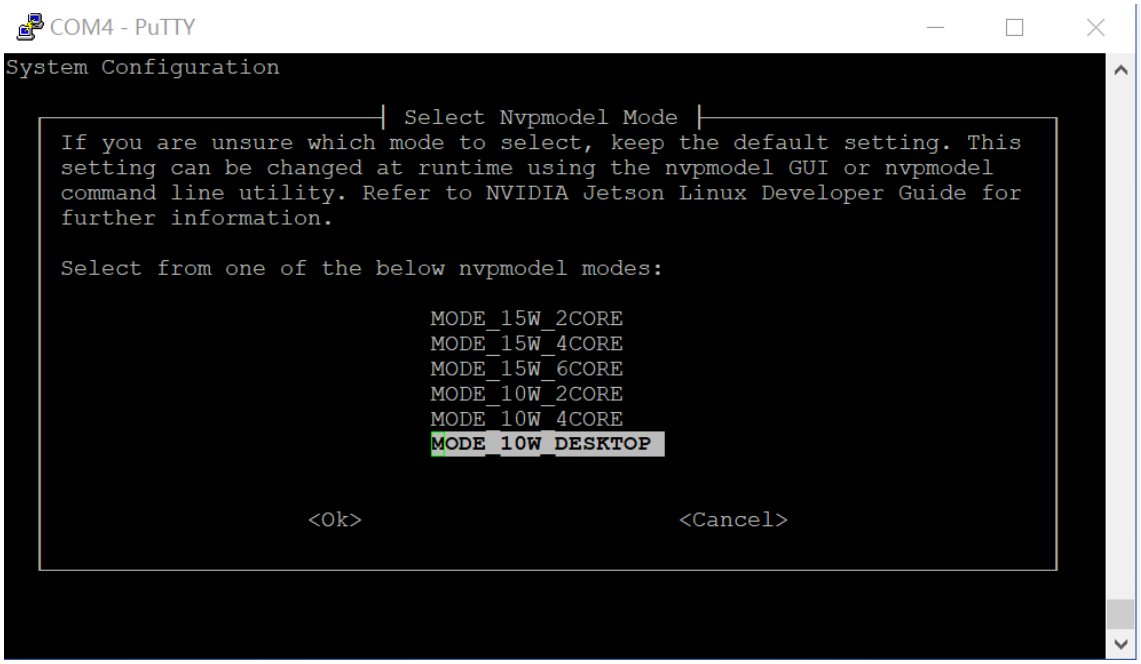
输入回车后，弹出网络配置界面：



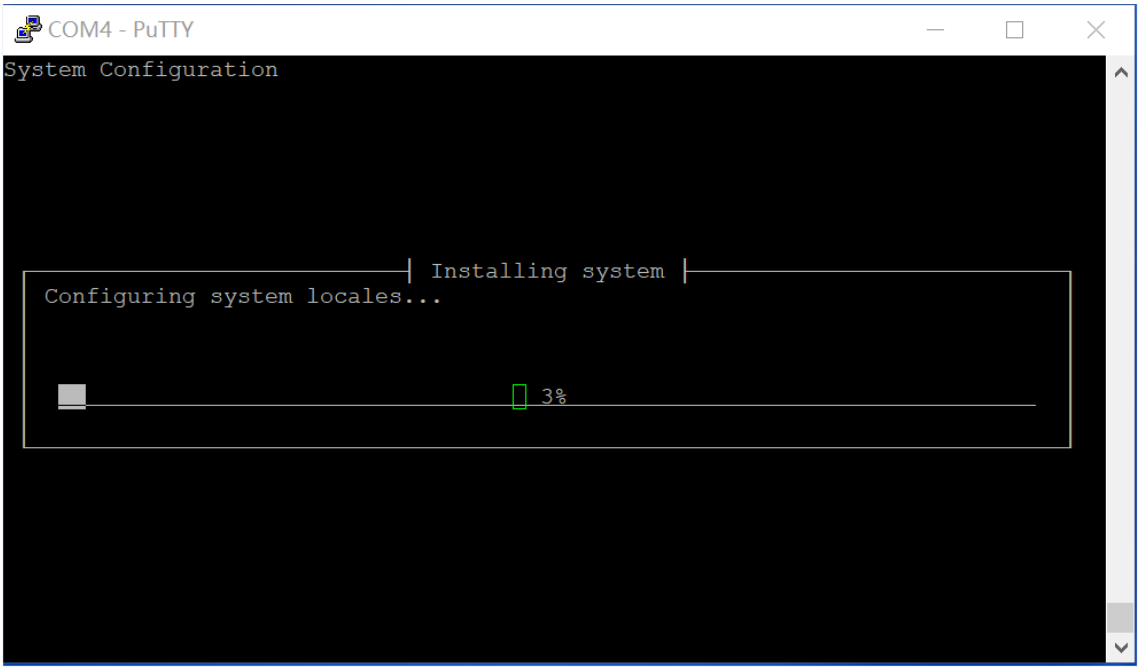
移动光标到最后一个选项，暂时先不配置网络（如有需求配置网络，在这界面选择 Configure network manually），输入回车到 hostname 设置界面：



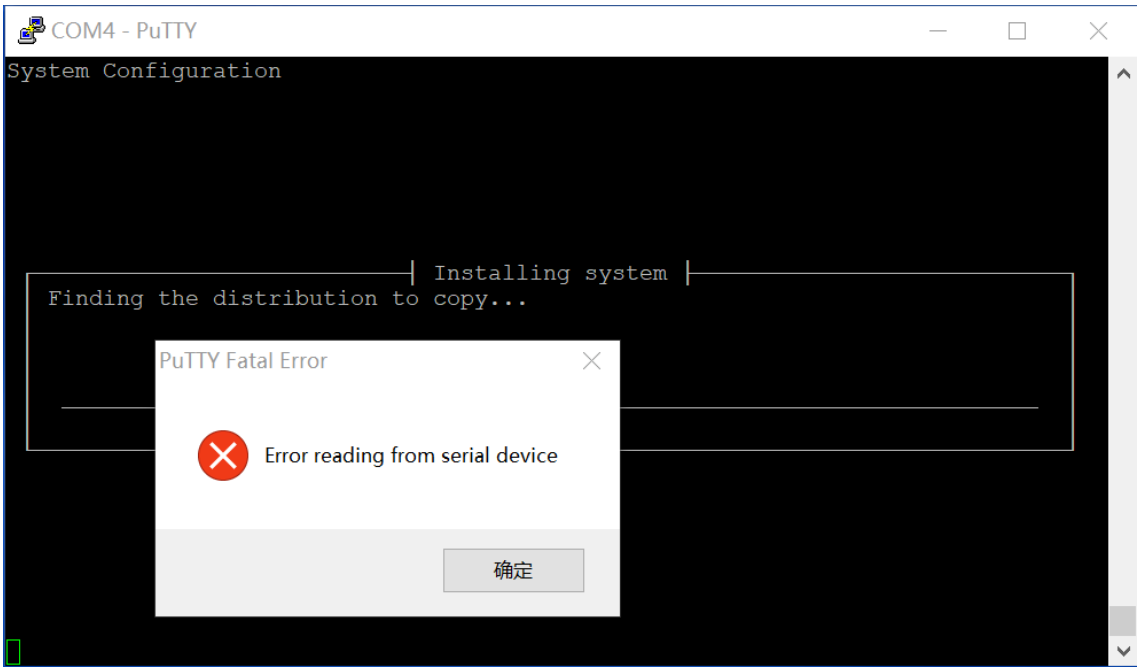
根据实际设置完 hostname 后，按回车进入 NVIDIA 功耗配置界面：



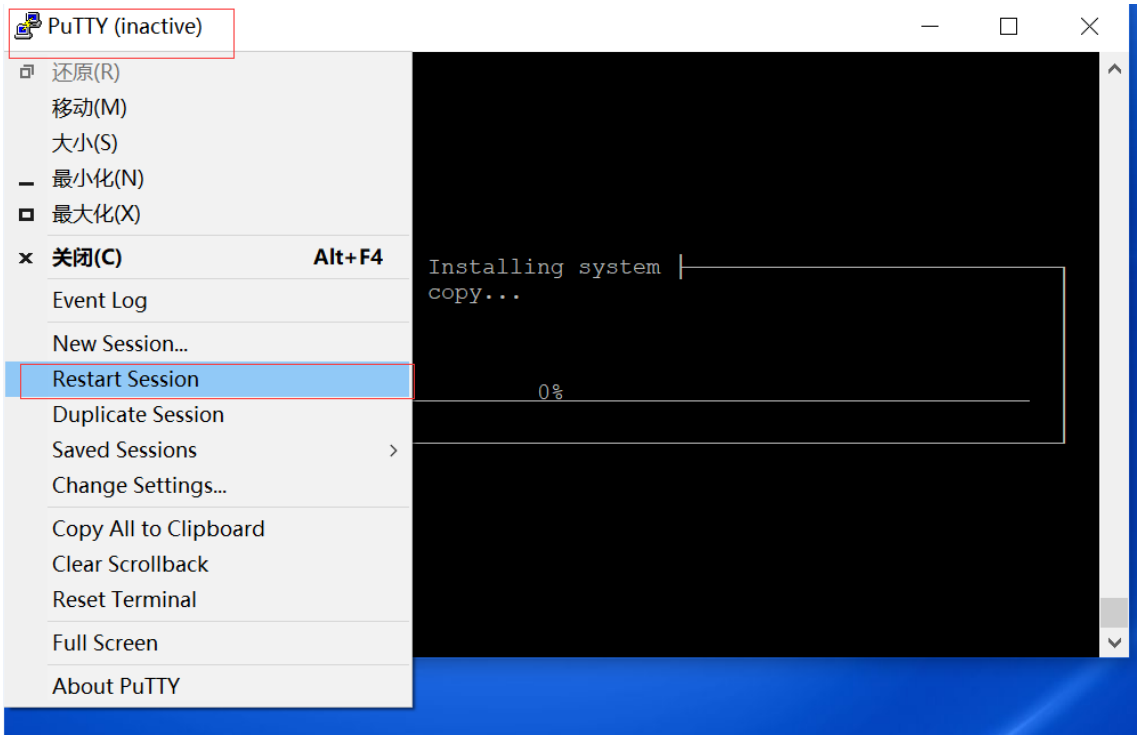
根据实际需求选择，最大功耗为 15W，大的功耗会带来更多的性能和发热，选择完成后按回车，系统开始自动配置：



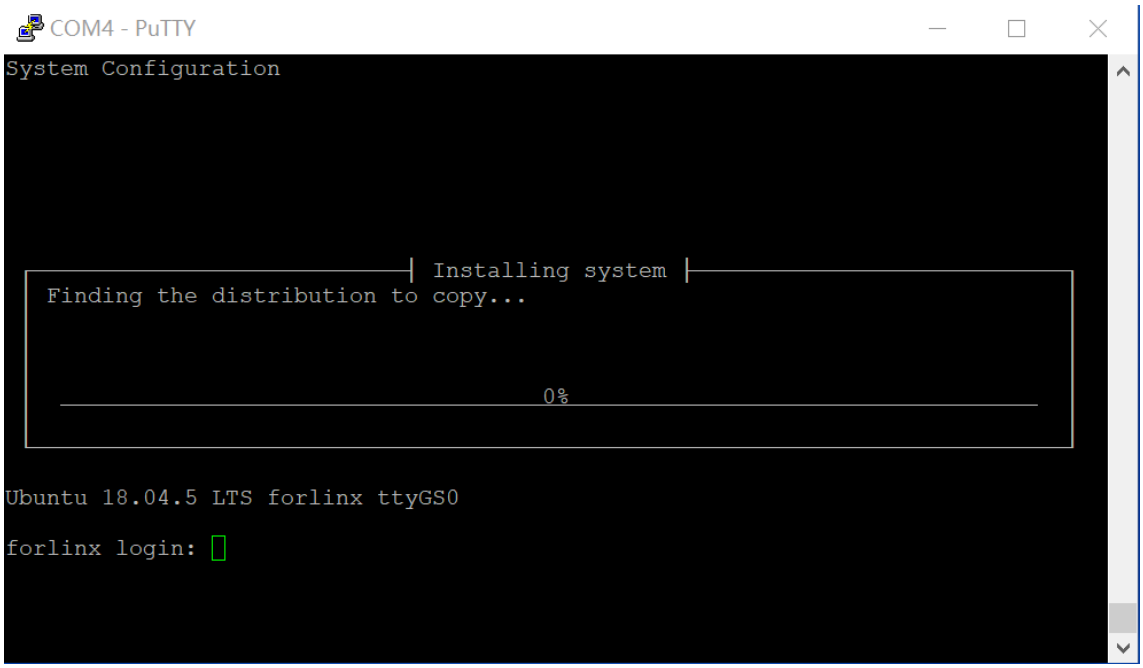
待配置完成后，USB 虚拟串口会断开，



点击“确定”按钮，点击左上角从弹出的界面中选择重新连接该虚拟串口：



此时，配置完成，USB 虚拟串口可以通过上述设置的用户名和密码登录到系统内：


COM4 - PuTTY
System Configuration
Installing system
Finding the distribution to copy...
0%
Ubuntu 18.04.5 LTS forlinx ttyGS0
forlinx login:

第三章　FCU3001 平台接口

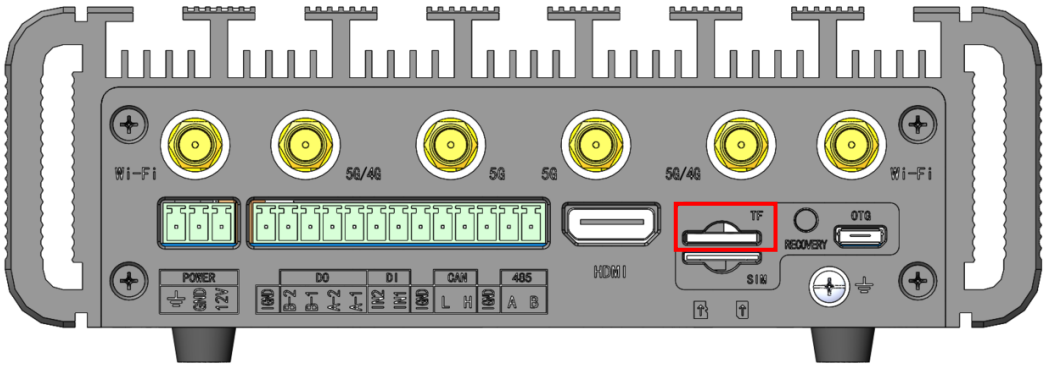
此章节主要说明 FCU3001 外设接口的使用方法。

3.1　TF 卡使用

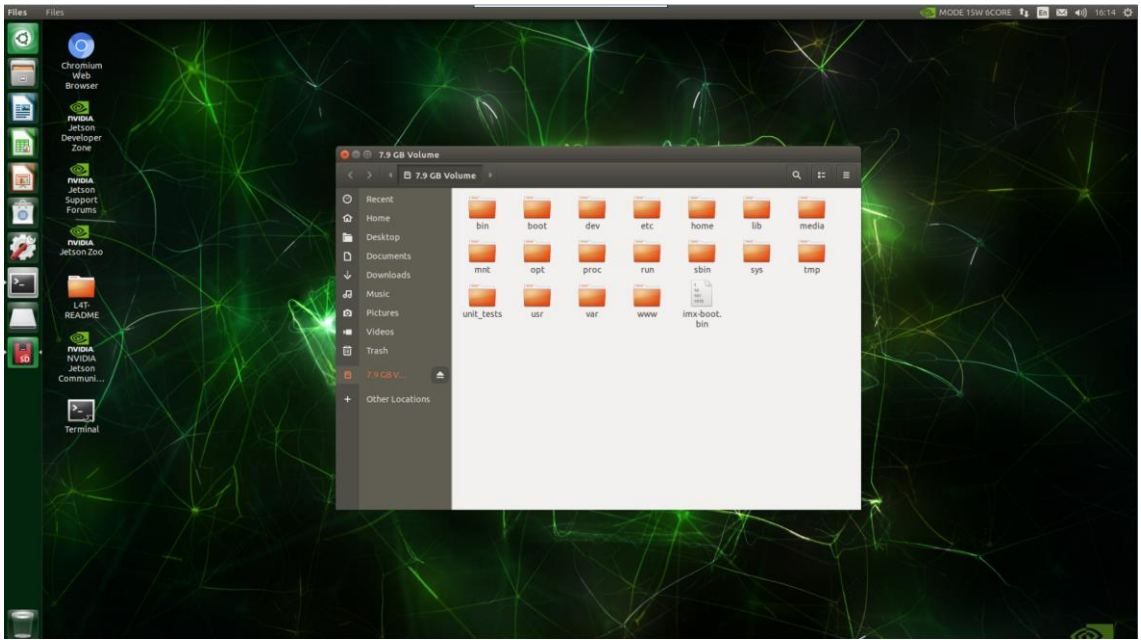
🕮 **说明：**

- **TF 卡使用前需用格式化工具格式化为 FAT32 格式。**
- **TF 卡挂载目录为/ media，支持热插拔，终端会打印关于 TF 卡的信息。**
- **不同的 TF 卡显示信息可能会有差别，本公司采用闪迪 8G、32G 的 TF 卡进行测试。**

TF 卡插入开发板的 TF 卡槽后，系统会自动检查并挂载 TF 卡，挂载成功后，可对 TF 卡进行读写操作。TF 卡槽位置如图红框所示



插入 TF 卡后，桌面自动弹出文件浏览器显示卡内容。

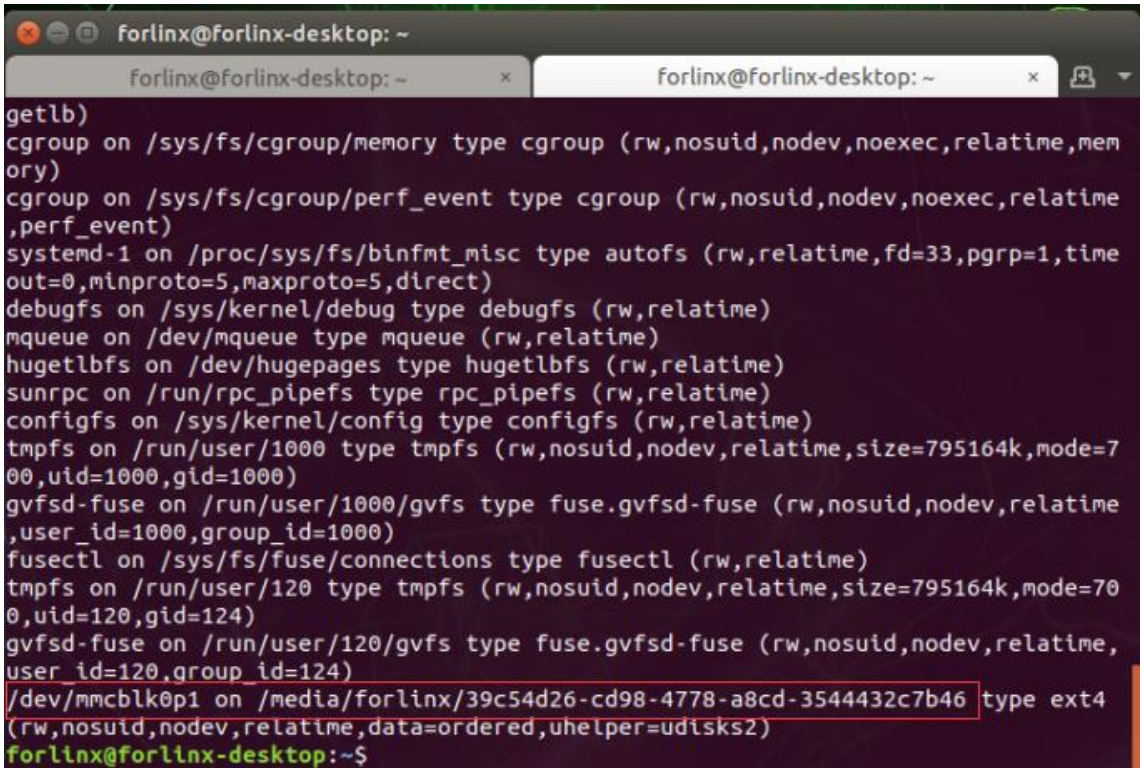


命令行操作：

1、查看挂载：

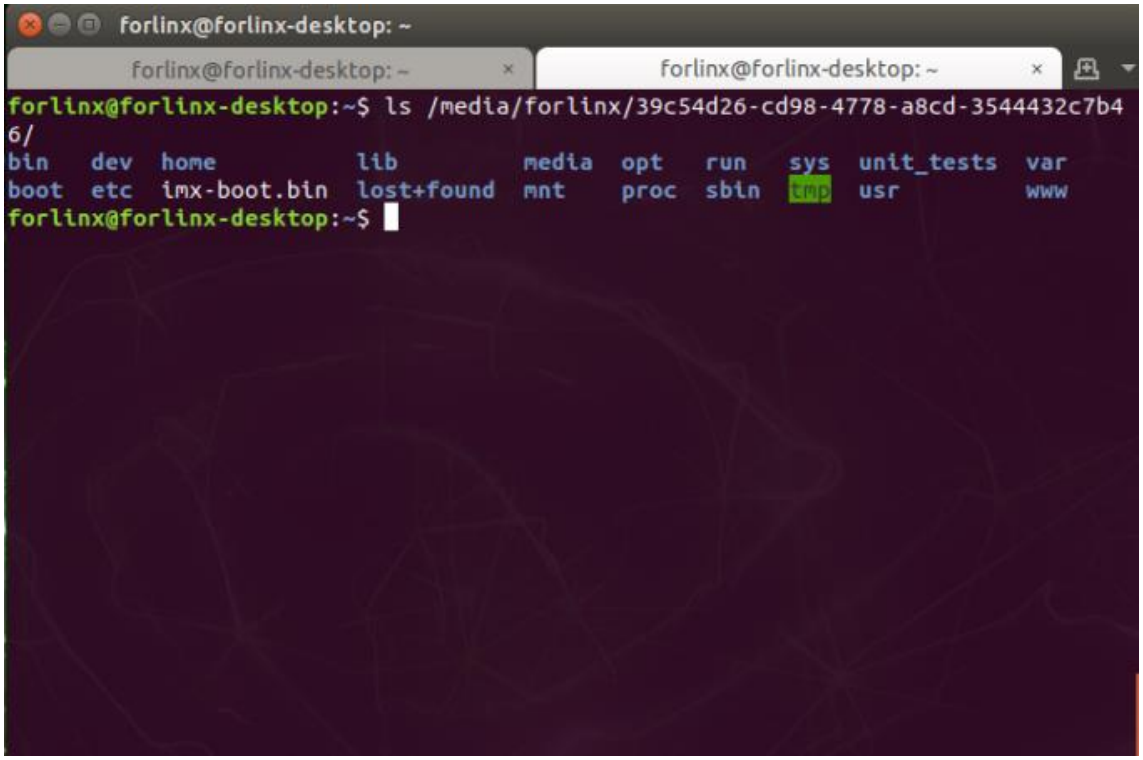
```
forlinx@forlinx-desktop:~$ mount
```

打印如下，红框所示为 TF 卡自动挂载位置：



2、查看该 TF 卡的文件（卡的名字会随卡变动）：

```
forlinx@forlinx-desktop:~$ ls /media/forlinx/39c54d26-cd98-4778-a8cd-3544432c7b46/
bin    dev   home          lib         media  opt   run   sys  unit_tests  var
boot   etc   imx-boot.bin  lost+found  mnt    proc  sbin  tmp  usr         www
```



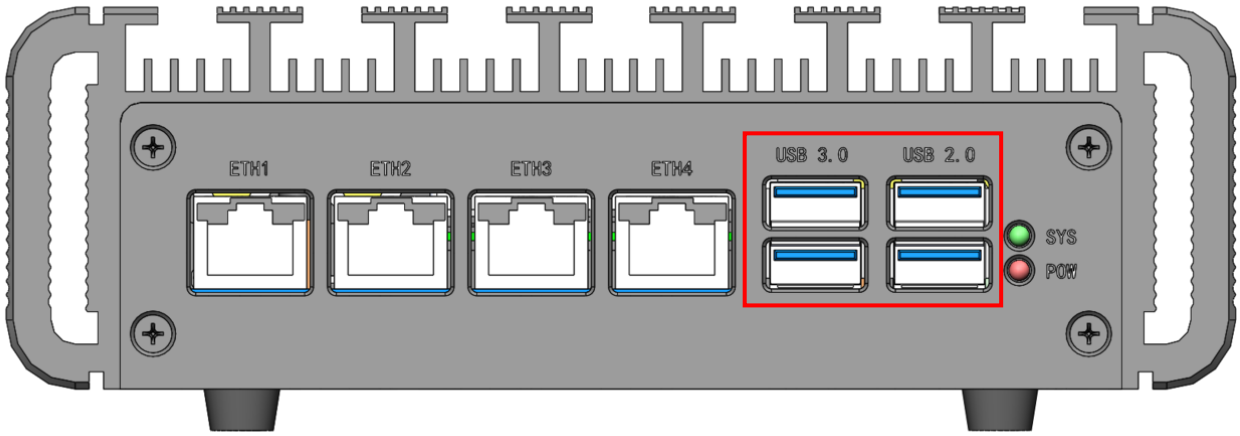
⚠ **注意：尽量不要在读写 TF 卡的同时拔出 TF 卡**

3.2 USB 接口使用

3.2.1 USB HOST 接口存储测试

🕮 **说明：**

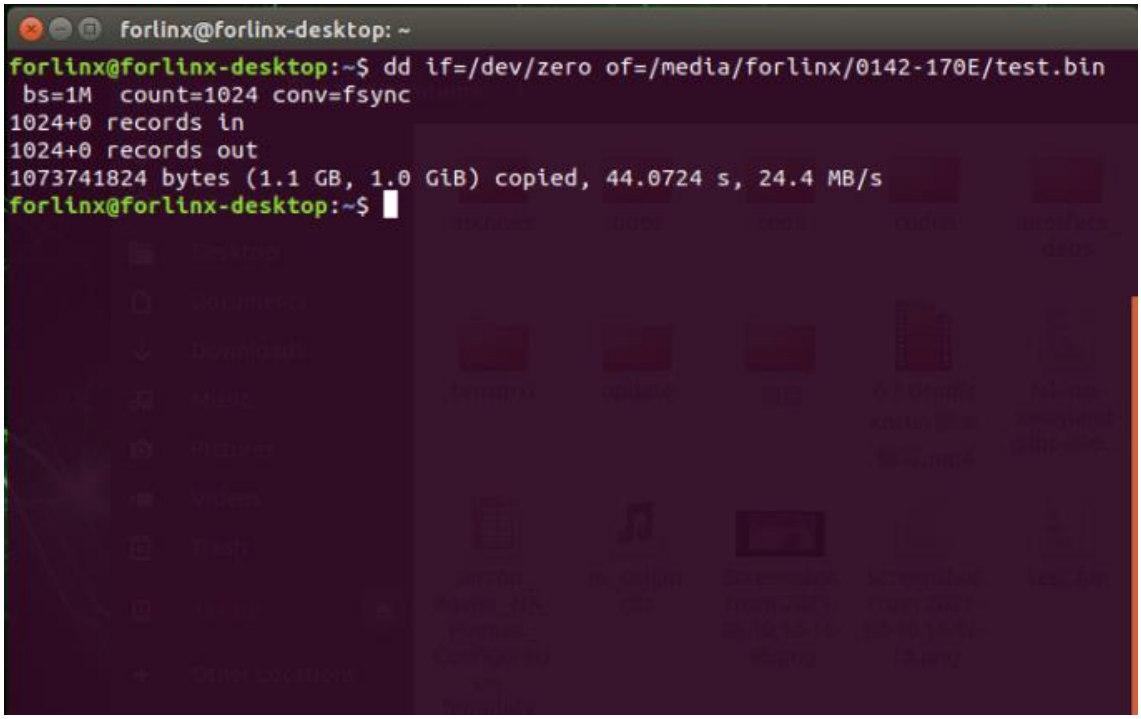
- **支持 USB 鼠标、USB 键盘、U 盘设备的热插拔。**
- **使用 U 盘测试时，建议用格式化工具将其格式化为能被 linux 系统识别的 FAT32 格式。**
- **目前 U 盘测试支持到 124G，124G 以上并未测试。**
- **U 盘挂载目录为/meidia/“用户名”/**



FCU3001 上有四个 USB HOST 接口，如红框所示，两个蓝色接口为 USB3.0，白色接口为 USB2.0。可选择任意一个进行测试，插入 U 盘系统文件浏览器自动显示 U 盘内容。

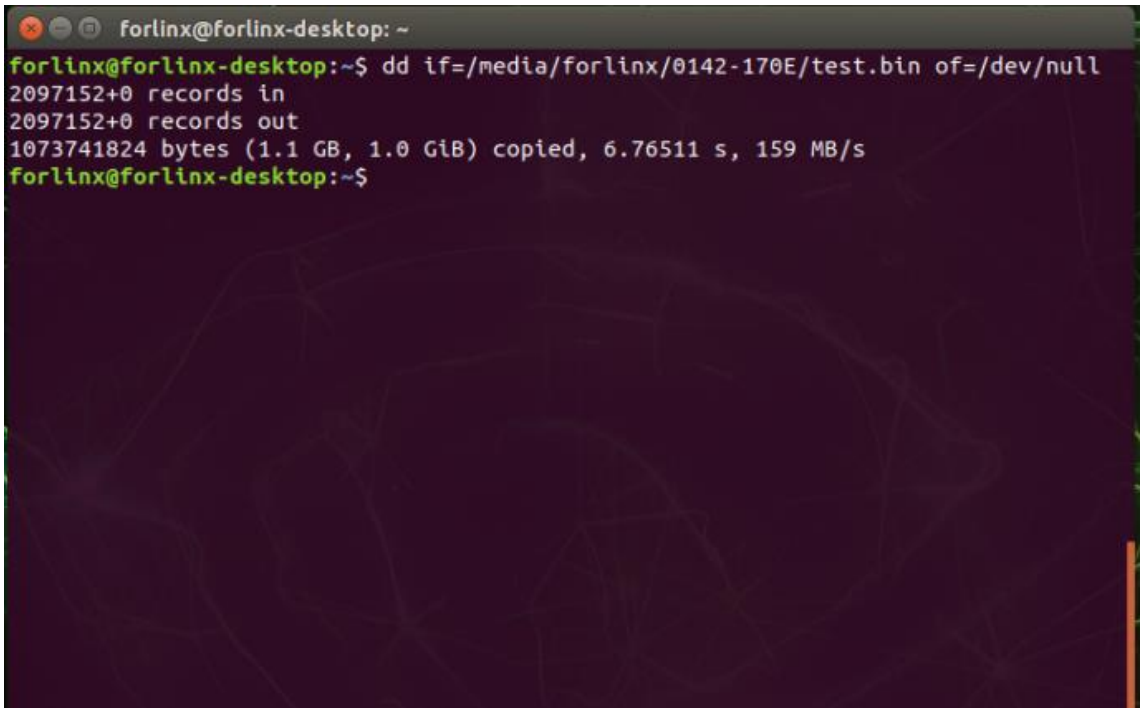
读写速度测试：

1、写测试：




```
forlinx@forlinx-desktop:~$ dd if=/dev/zero of=/media/forlinx/0142-170E/test.bin  bs=1M count=1024 conv=fsync
1024+0 records in
1024+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 44.0724 s, 24.4 MB/s
```

1、 读测试：




```
forlinx@forlinx-desktop:~$ dd  if=/media/forlinx/0142-170E/test.bin  of=/dev/null
2097152+0  records  in
```

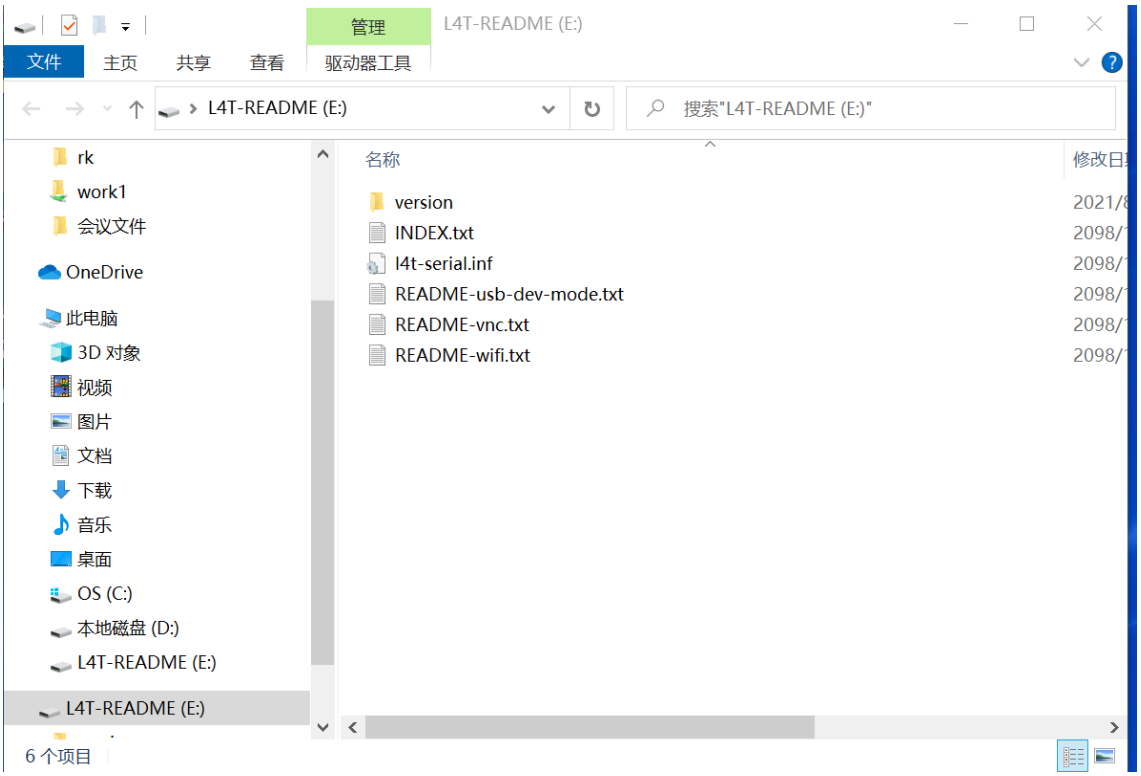
```
2097152+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 6.76511 s, 159 MB/s
```

3.2.2 USB OTG 远程登录

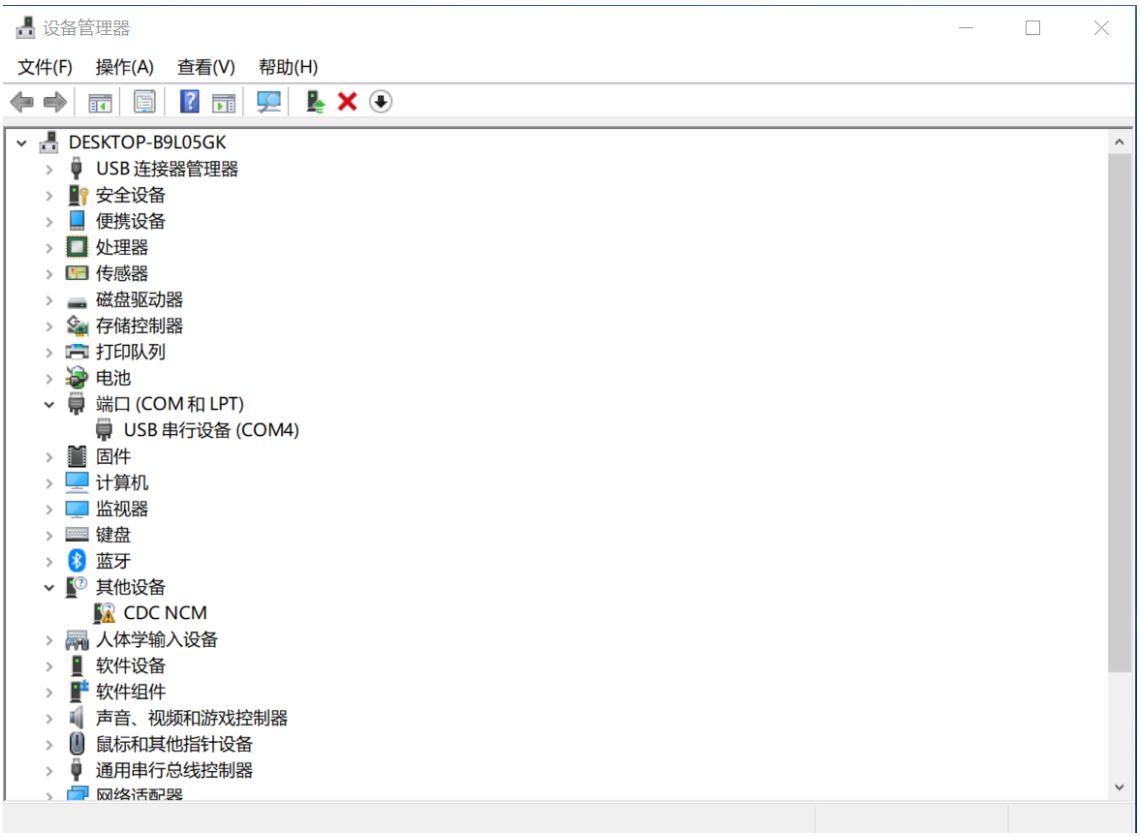
USB OTG 优先推荐在 Windows10 上使用，Windows7 系统无驱动，其它系统参考 storage 里的 README-usb-dev-mode.txt。

将 OTG 口通过 micro USB 线缆连接到 Windows10 PC 机。识别成功后会显示 USB 移动存储设备和串口设备。

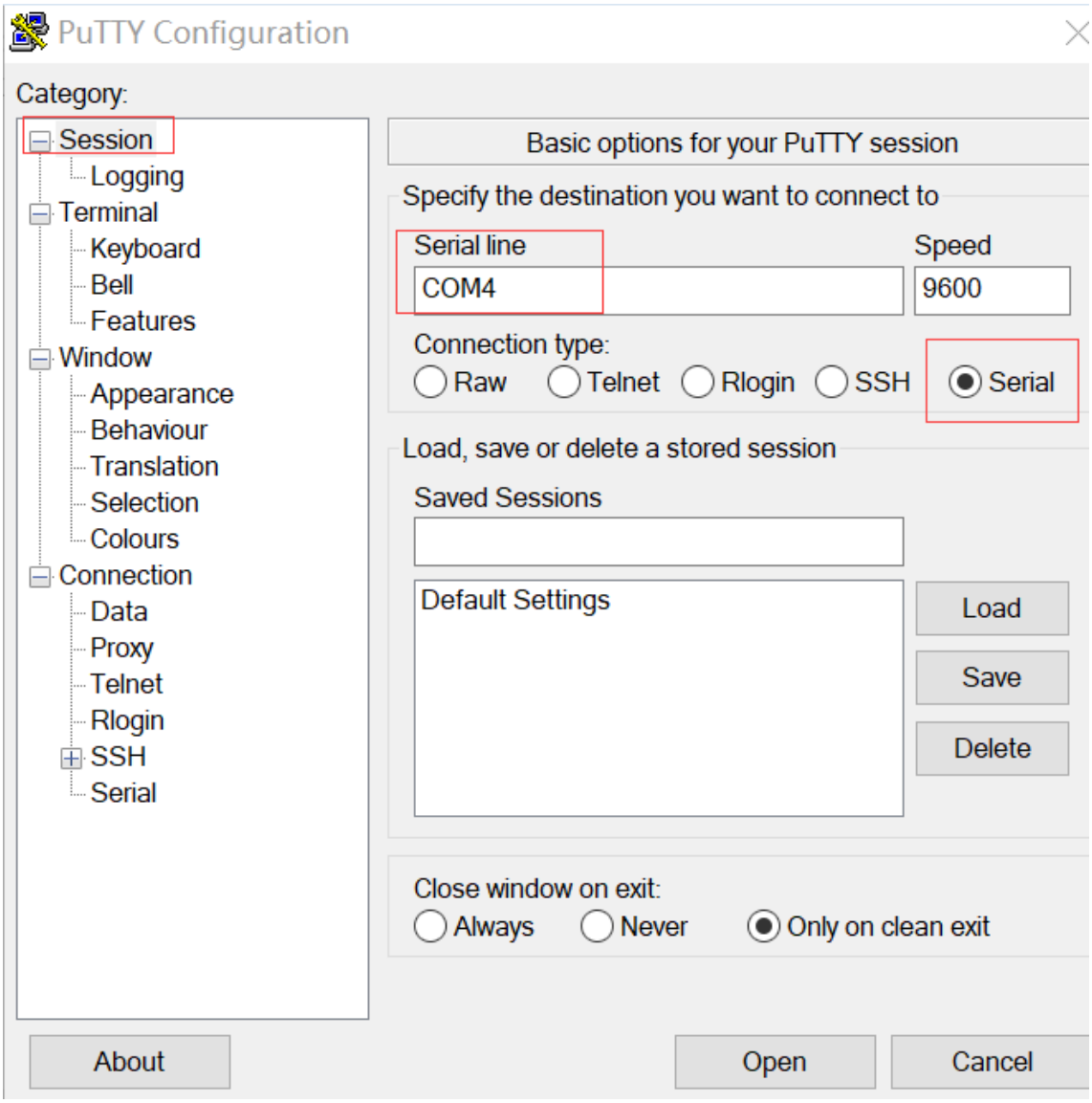
移动存储内容如下：

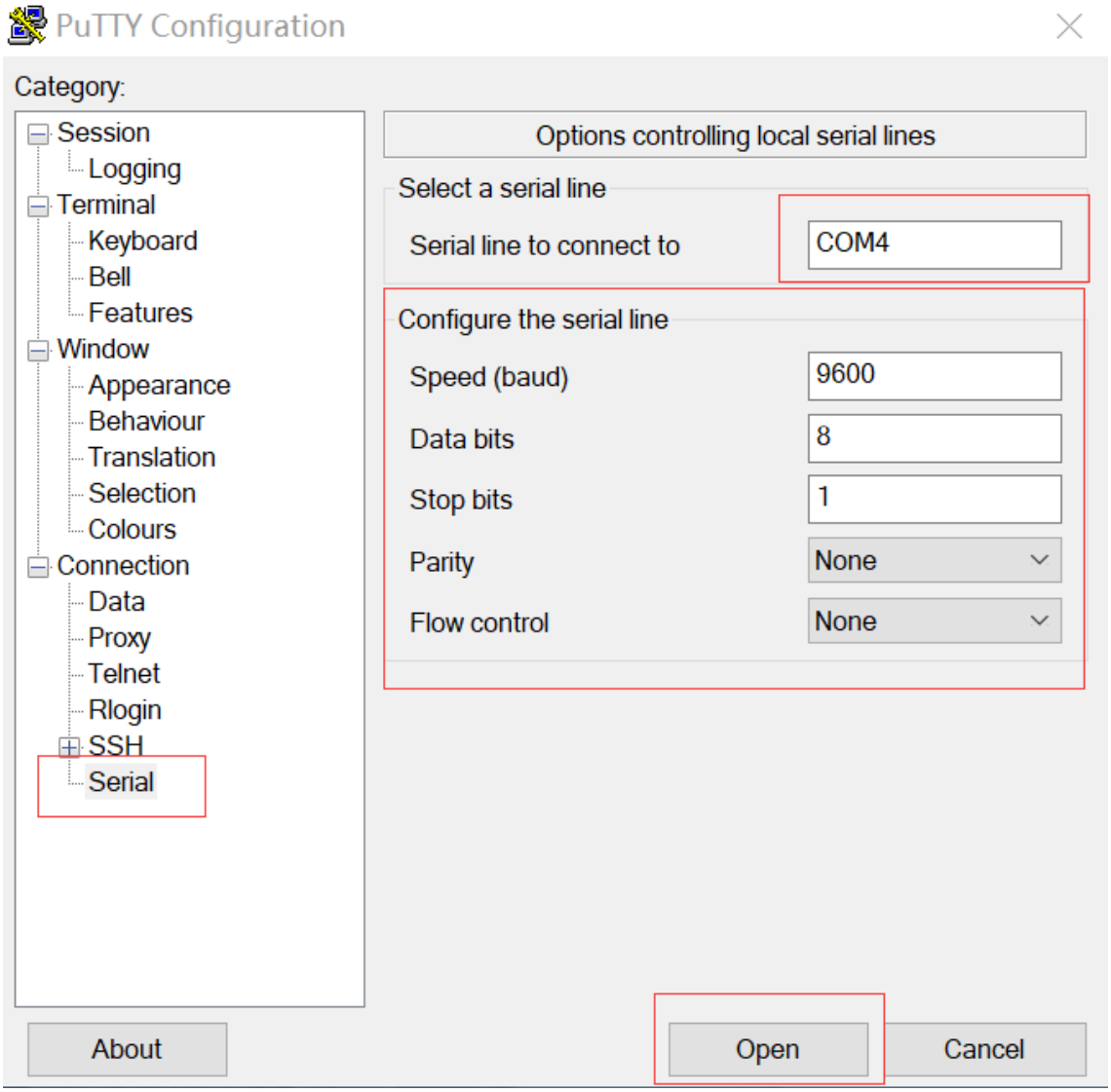


串口通过设备管理器查看序号：

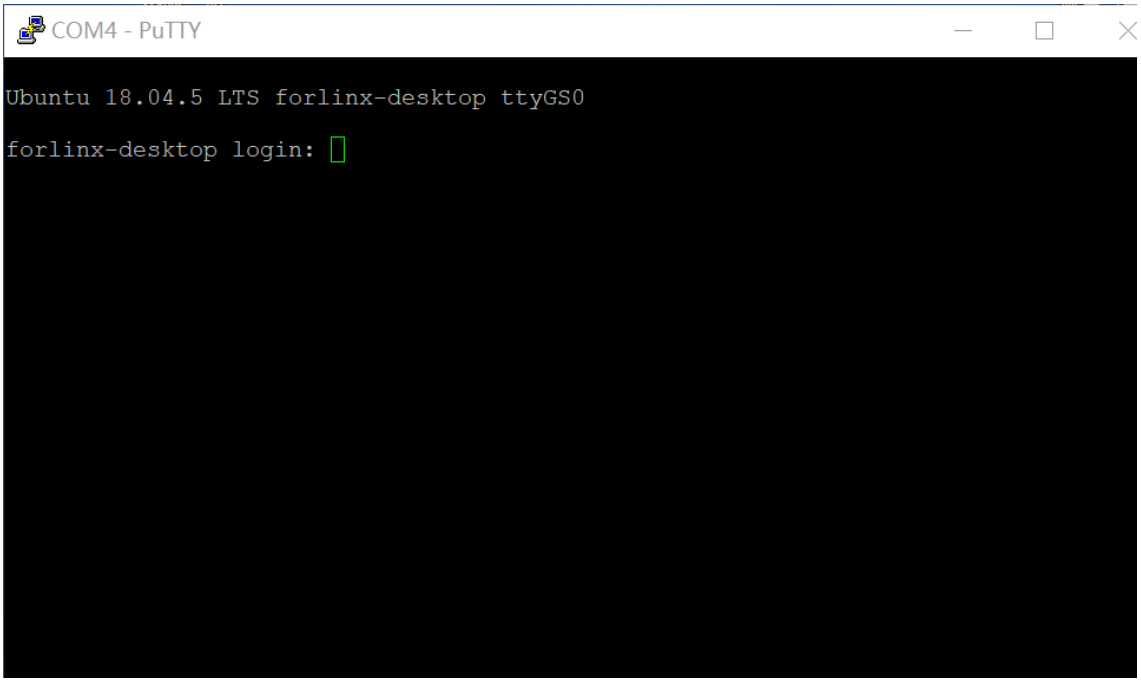


如上图所示串口编号为 COM4，通过 putty 打开该串口可以登录到 FCU3001，串口模式必须设置为 8N1，即 8bit 数据位，无校验位，1 结束位（READ-usb-dev-mode.txt 中对串口 8N1 解释有误）。

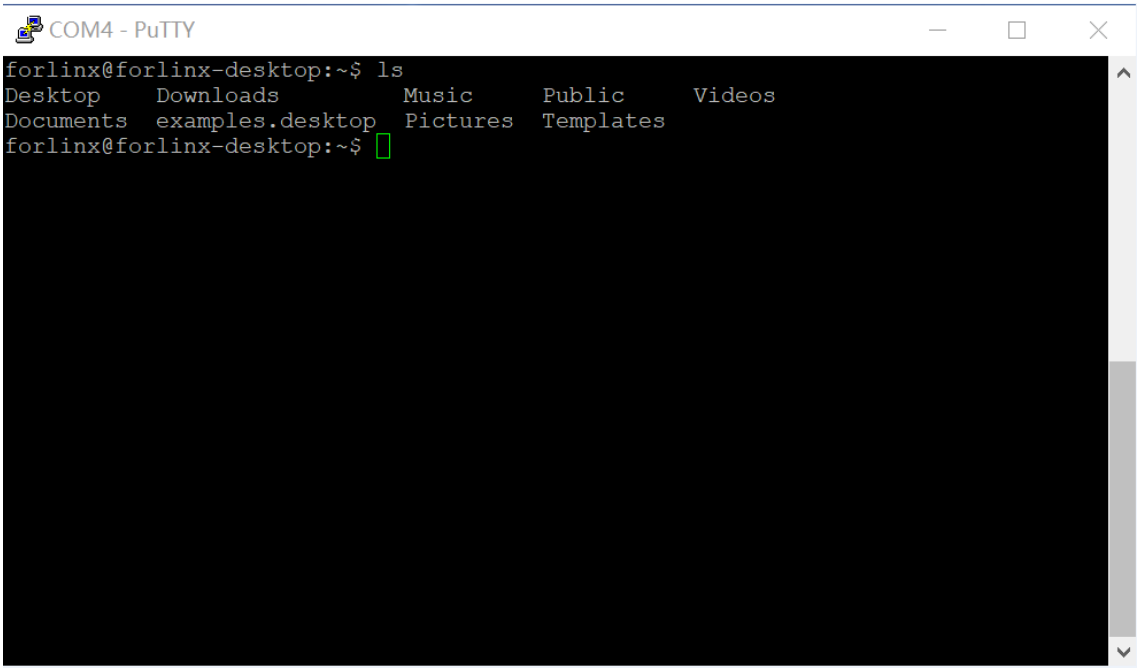

PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Serial
Basic options for your PuTTY session
Specify the destination you want to connect to
Serial line
Speed
COM4
9600
Connection type:
Raw
Telnet
Rlogin
SSH
Serial
Load, save or delete a stored session
Saved Sessions
Default Settings
Load
Save
Delete
Close window on exit:
Always
Never
Only on clean exit
About
Open
Cancel



点击 Open 后，提示输入用户名和密码：



输入用户名和密码，成功登录到系统内：



3.2.3 USB 摄像头测试

📖 **说明：本产品支持 USB 摄像头：**Webcam C270。

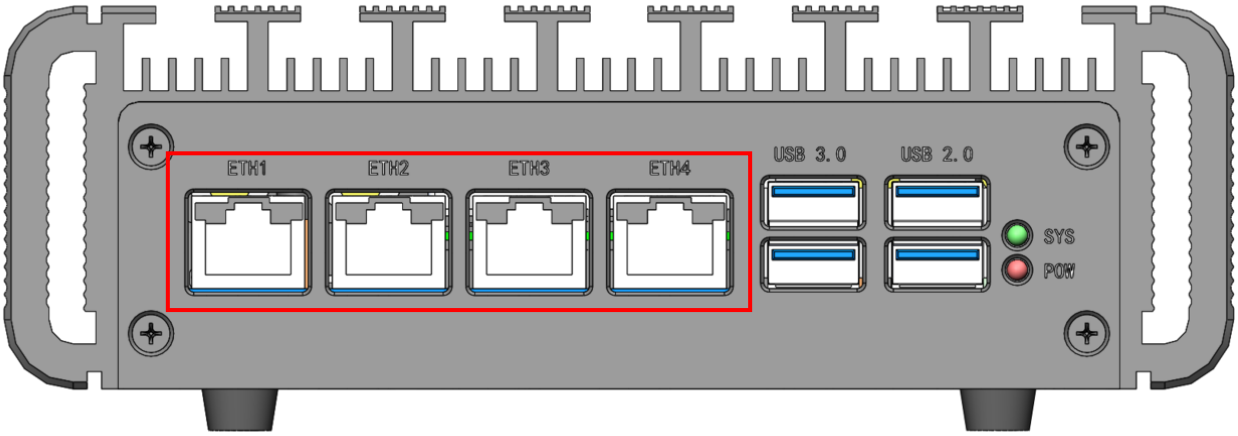
具体使用方法参考 5.6 UVC 摄像头

3.3 有线网络

FCU3001 内部集成五口网络交换，对外有四个千兆以太网，任意网口均可连接外网或者设备，对内有一个网口和交换通信。所有有线网络配置均可通过 ubuntu 界面配置，配置方法同 PC 机。本章节只给出终端命令行配置方法。

🕮 **说明：**

- **本节示例在飞凌的网络环境下进行测试，实际使用中网络环境会有差异，请用户按照自己实际网络环境进行配置。**
- **FCU3001 网卡名为 eth0，首次开机无默认 IP。**



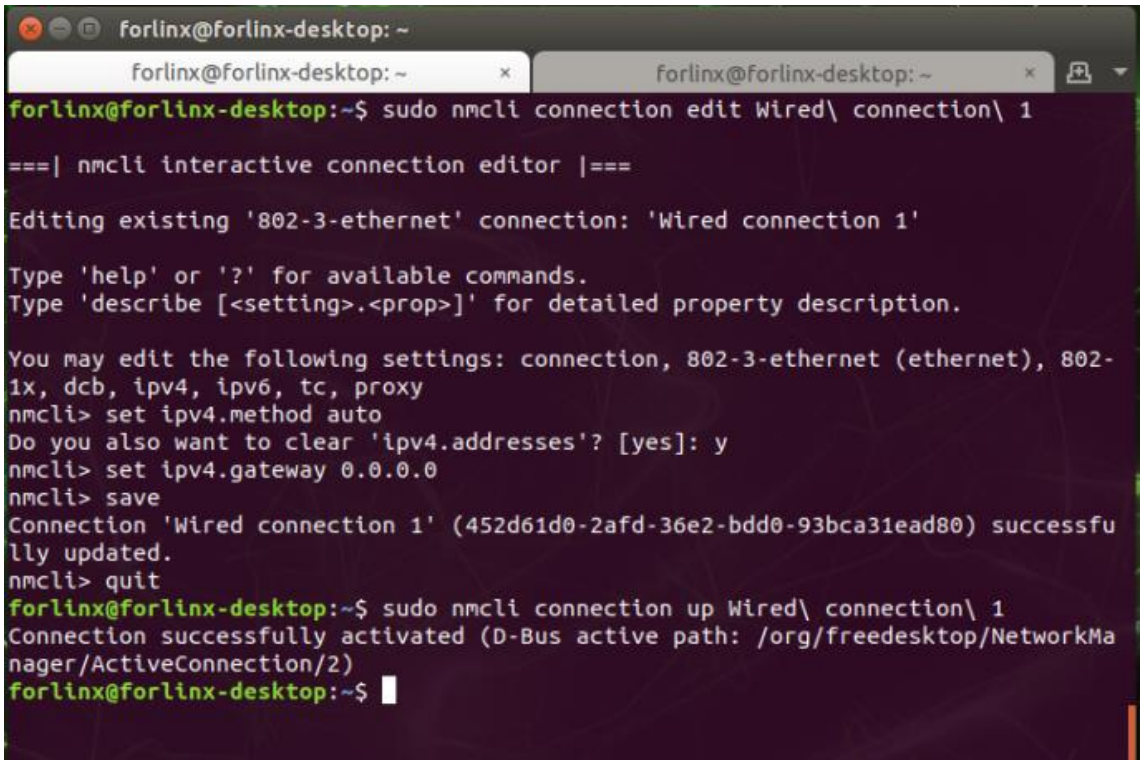
3.3.1 IPV4 动态 IP

如果您的 FCU3001 与路由器连接，且路由器支持 DHCP 自动 IP 地址分配可以在超级终端里面输入命令：

```
forlinx@forlinx-desktop:~$ sudo nmcli connection edit Wired\ connection\ 1

===| nmcli interactive connection editor |===

Editing existing '802-3-ethernet' connection: 'Wired connection 1'

Type 'help' or '?' for available commands.
Type 'describe [<setting>.<prop>]' for detailed property description.

You may edit the following settings: connection, 802-3-ethernet (ethernet), 802-1x, dcb, ipv4, ipv6, tc, proxy
nmcli> set ipv4.method auto
Do you also want to clear 'ipv4.addresses'? [yes]: y
nmcli> set ipv4.gateway 0.0.0.0
nmcli> save
Connection 'Wired connection 1' (452d61d0-2afd-36e2-bdd0-93bca31ead80) successfully updated.
```

```
nmcli> quit
forlinx@forlinx-desktop:~$ sudo nmcli connection up Wired\ connection\ 1
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/2)
forlinx@forlinx-desktop:~$
```

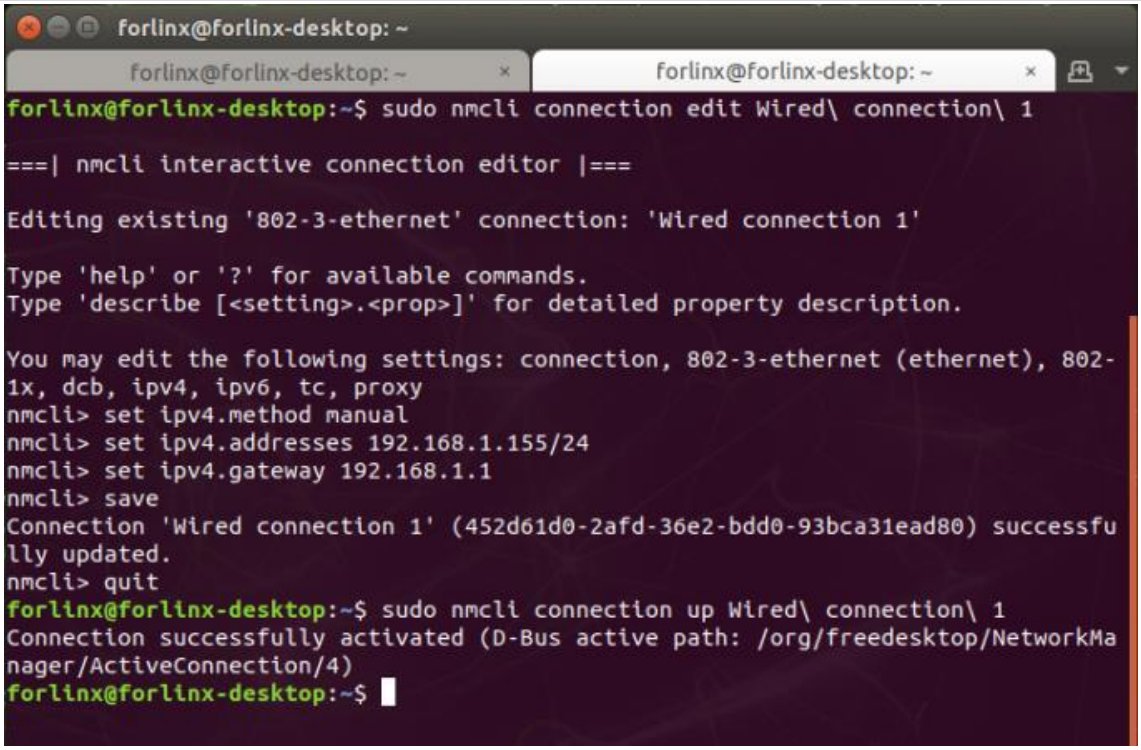
如果之前为静态 IP，会提示 Do you also want to clear 'ipv4.addresses'? [yes]:，输入 y 即可



3.3.2 IPV4 静态 IP

FCU3001 打开终端，依次输入如下命令即可实现静态 IP：

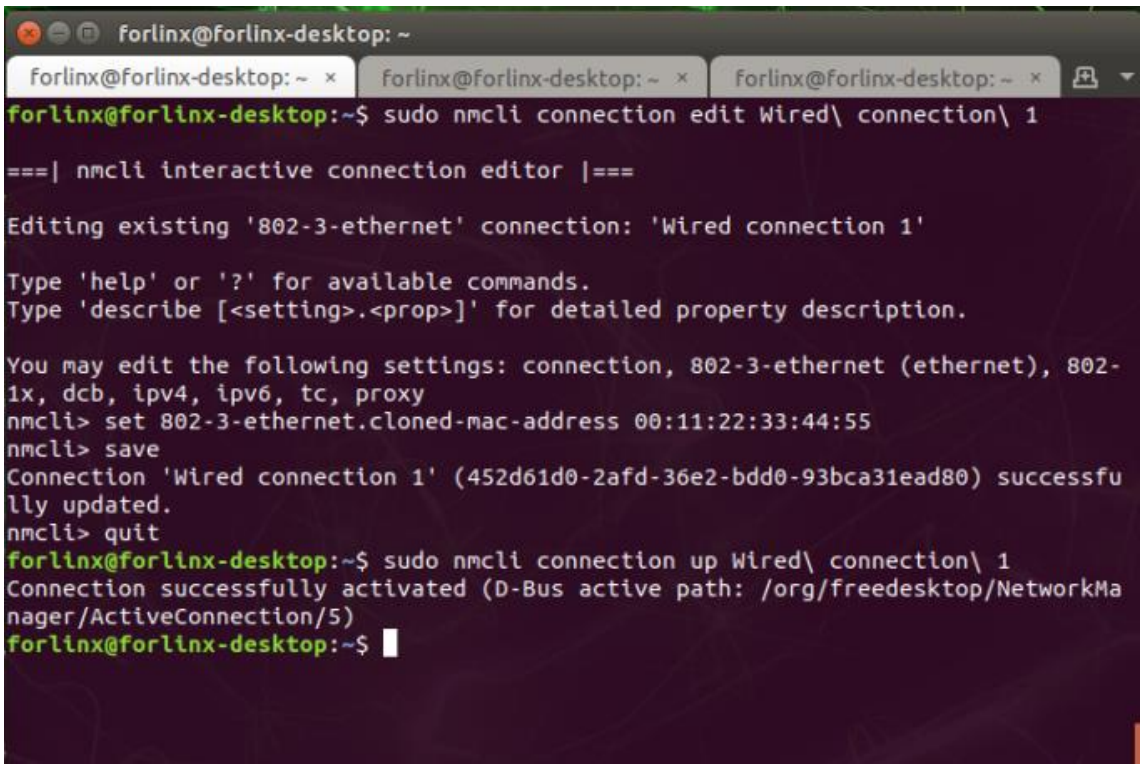
```
forlinx@forlinx-desktop:~$ sudo nmcli connection edit Wired\ connection\ 1

===| nmcli interactive connection editor |===

Editing existing '802-3-ethernet' connection: 'Wired connection 1'

Type 'help' or '?' for available commands.
Type 'describe [<setting>.<prop>]' for detailed property description.

You may edit the following settings: connection, 802-3-ethernet (ethernet), 802-1x, dcb, ipv4, ipv6, tc, proxy
nmcli> set ipv4.method manual
nmcli> set ipv4.addresses 192.168.1.155/24
nmcli> set ipv4.gateway 192.168.1.1
nmcli> save
Connection 'Wired connection 1' (452d61d0-2afd-36e2-bdd0-93bca31ead80) successfully updated.
nmcli> quit
forlinx@forlinx-desktop:~$ sudo nmcli connection up Wired\ connection\ 1
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/4)
forlinx@forlinx-desktop:~$
```



3.3.3 改变 MAC 地址

```
forlinx@forlinx-desktop:~$ sudo nmcli connection edit Wired\ connection\ 1
```

```
===| nmcli interactive connection editor |===

Editing existing '802-3-ethernet' connection: 'Wired connection 1'

Type 'help' or '?' for available commands.
Type 'describe [<setting>.<prop>]' for detailed property description.

You may edit the following settings: connection, 802-3-ethernet (ethernet), 802-1x, dcb, ipv4, ipv6, tc, proxy
nmcli> set 802-3-ethernet.cloned-mac-address 00:11:22:33:44:55
nmcli> save
Connection 'Wired connection 1' (452d61d0-2afd-36e2-bdd0-93bca31ead80) successfully updated.
nmcli> quit
forlinx@forlinx-desktop:~$ sudo nmcli connection up Wired\ connection\ 1
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/5)
```



3.3.4 IPV6

IPv6 设置原理与 IPv4 相同，只需将属性名由 ipv4 改为 ipv6，地址按照 ipv6 格式。

3.3.5 SSH 远程登录 FCU3001

默认已经开启 ssh 服务，在配置好 IP 之后可以通过 PC 远程登录到 FCU3001。

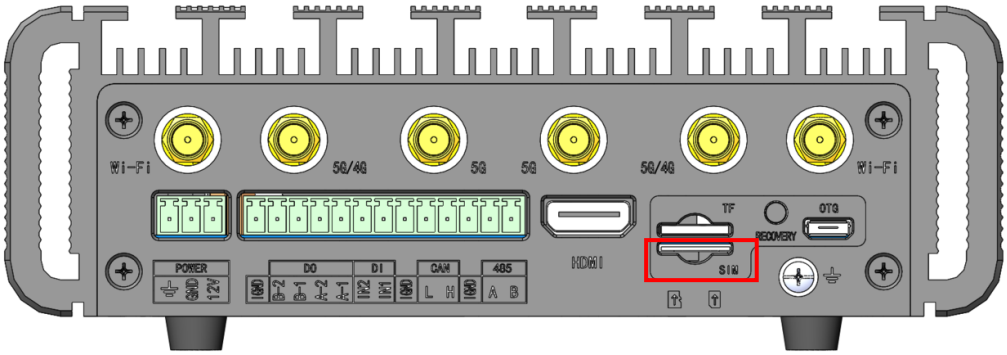
3.4 无线网络

3.4.1 4G/5G 上网

📖 **说明：**

- **目前 FCU3001 支持移远 EM05-CE 4G 模组，移远 RM500Q-GL 5G 模组和移远 RM500U-CN 5G 模组。**

确保 FCU3001 内部安装了 4G/5G 模块，并连接了 4G/5G 天线，插入 SIM 卡。SIM 卡扣位置如图红框。



2.9.1 4G/5G 识别判断

➢ **移远 EM05-CE 模块**

可以在 FCU3001 的控制台输入一下命令看移远 EM05-CE 4G 模块是否识别成功。

```
forlinx@ubuntu:~$ lsusb
Bus 002 Device 003: ID 2c7c: 0125
Bus 002 Device 002: ID 04b4:6500 Cypress Semiconductor Corp.
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 002: ID 04b4:6502 Cypress Semiconductor Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

使用 ifconfig 命令可以看到移远 EM05-CE 5G 模块产生的节点名称为 usb1。

➢ **移远 RM500Q-GL 模块**

可以在 FCU3001 的控制台输入一下命令看移远 RM500Q-GL 5G 模块是否识别成功。

```
forlinx@ubuntu:~$ lsusb
Bus 002 Device 003: ID 2c7c:0800
Bus 002 Device 002: ID 04b4:6500 Cypress Semiconductor Corp.
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 002: ID 04b4:6502 Cypress Semiconductor Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

使用 ifconfig 命令可以看到移远 RM500Q-GL 5G 模块模块产生的节点名称为 usb1。

- **移远 RM500U-CN 模块**

  可以在 FCU3001 的控制台输入一下命令看移远 RM500U-CN 5G 模块是否识别成功。

```
forlinx@ubuntu:~$ lsusb
Bus 002 Device 003: ID 2c7c:0900
Bus 002 Device 002: ID 04b4:6500 Cypress Semiconductor Corp.
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 002: ID 04b4:6502 Cypress Semiconductor Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

  使用 ifconfig 命令可以看到移远 RM500U-CN 5G 模块产生的节点名称为 usb1。

2.9.2 4G/5G 上网测试

- **移远 EM05-CE 模块**

  以插入移动 SIM 卡测试 4G 上网为例，测试是否连接外网，运行如下指令自动给模块上电，并拨号：

```
forlinx@ubuntu:~$ sudo fltest_4g5g.sh
[sudo] password for forlinx:

Pruduct is EM05-CE

Start connecting to the network ...
udhcpc: started, v1.27.2
udhcpc: sending discover
udhcpc: sending select for 10.5.167.145
udhcpc: lease of 10.5.167.145 obtained, lease time 7200
cmp: EOF on /tmp/tmp.HNaVBpoVBZ which is empty
EM05-CE dhcp Complite !!!
```

  运行如下命令，测试连通情况：

```
forlinx@forlinx-desktop:~$ ping -I usb1 www.forlinx.com
PING s-526319.gotocdn.com (211.149.226.120) from 192.168.2.103 usb1: 56(84) bytes of data.
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=1 ttl=50 time=40.5 ms
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=2 ttl=50 time=43.3 ms
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=3 ttl=50 time=38.7 ms
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=4 ttl=50 time=38.1 ms
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=5 ttl=50 time=44.8 ms
^C
--- s-526319.gotocdn.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 38.186/41.141/44.813/2.575 ms
```

- **移远 RM500Q-GL 模块**

以插入移动 SIM 卡测试 5G 上网为例，测试是否连接外网，运行如下指令自动给模块上电，并拨号：

```
forlinx@ubuntu:~$ sudo fltest_4g5g.sh
[sudo] password for forlinx:

Pruduct is RM500Q-GL

Start connecting to the network ...
udhcpc: started, v1.27.2
udhcpc: sending discover
udhcpc: sending select for 10.33.98.72
udhcpc: lease of 10.33.98.72 obtained, lease time 7200
cmp: EOF on /tmp/tmp.tRzCS3BieN which is empty
RM500Q-GL dhcp Complite !!!
```

运行如下命令，测试连通情况：

```
forlinx@forlinx-desktop:~$ ping -I usb1 www.forlinx.com
PING s-526319.gotocdn.com (211.149.226.120) from 192.168.2.103 usb1: 56(84) bytes of data.
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=1 ttl=50 time=40.5 ms
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=2 ttl=50 time=43.3 ms
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=3 ttl=50 time=38.7 ms
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=4 ttl=50 time=38.1 ms
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=5 ttl=50 time=44.8 ms
^C
--- s-526319.gotocdn.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 38.186/41.141/44.813/2.575 ms
```

- **移远 RM500U-CN 模块**

以插入移动 SIM 卡测试 5G 上网为例，测试是否连接外网，运行如下指令自动给模块上电，并拨号：

```
forlinx@ubuntu:~$ sudo fltest_4g5g.sh
[sudo] password for forlinx:

Pruduct is RM500U-CN

Start connecting to the network ...
udhcpc: started, v1.27.2
udhcpc: sending discover
udhcpc: sending select for 10.13.36.54
udhcpc: lease of 10.13.36.54 obtained, lease time 86400
cmp: EOF on /tmp/tmp.Boe9B5XaN1 which is empty
RM500U-CN dhcp Complite !!!
```

运行如下命令，测试连通情况：

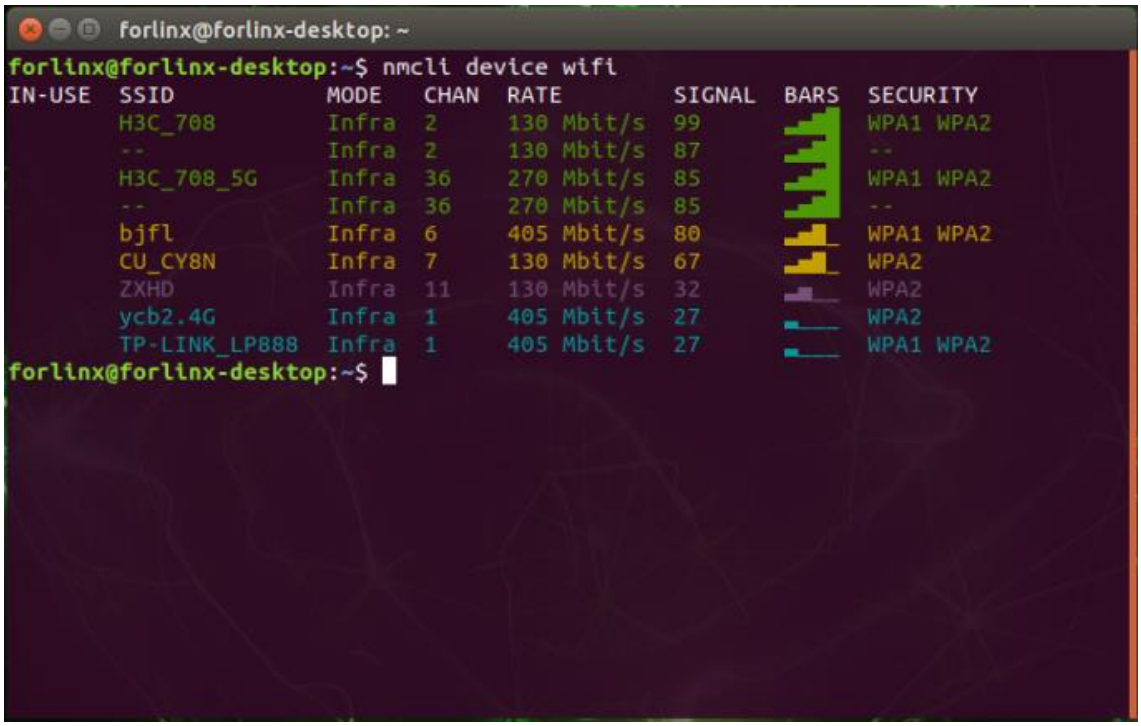
```
forlinx@forlinx-desktop:~$ ping -I usb1 www.forlinx.com
PING s-526319.gotocdn.com (211.149.226.120) from 192.168.2.103 usb1: 56(84) bytes of data.
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=1 ttl=50 time=40.5 ms
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=2 ttl=50 time=43.3 ms
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=3 ttl=50 time=38.7 ms
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=4 ttl=50 time=38.1 ms
64 bytes from s-526319.gotocdn.com (211.149.226.120): icmp_seq=5 ttl=50 time=44.8 ms
^C
--- s-526319.gotocdn.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 38.186/41.141/44.813/2.575 ms
```

3.4.2 WIFI 上网

确保 FCU3001 内已经安装 Intel 3168NGW WIFI 模块。支持连接 5G AP。

3.4.2.1 查看附近 AP

```
forlinx@forlinx-desktop:~$ nmcli device wifi
```

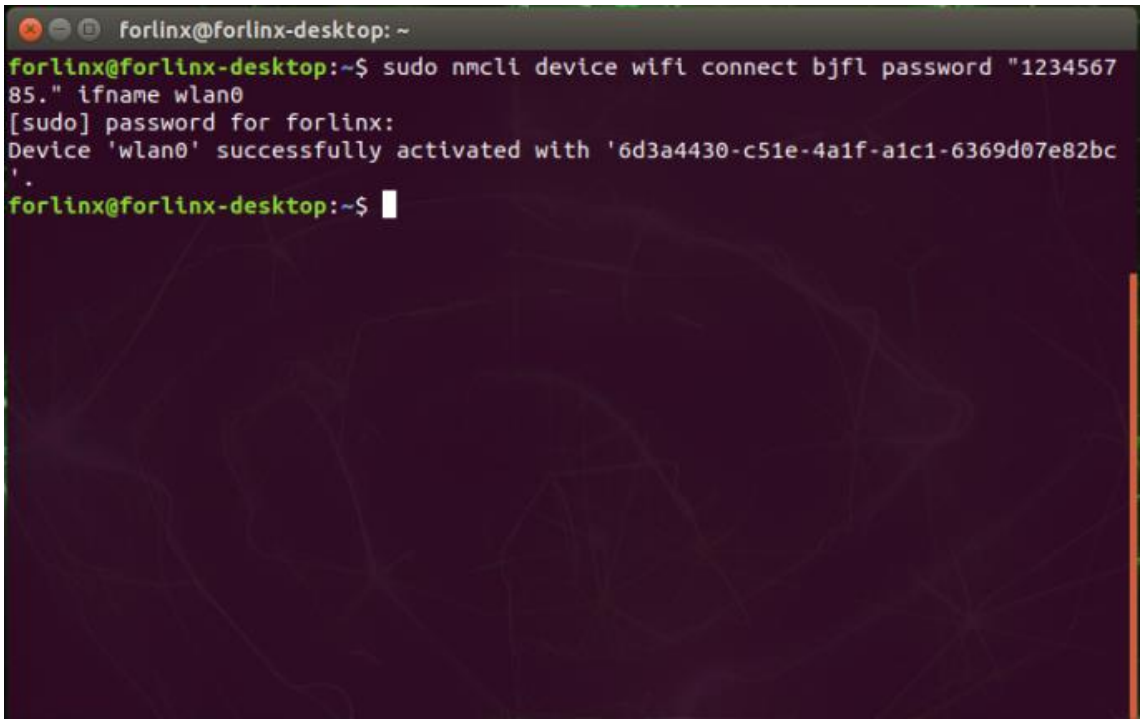


SSID 为附近的 AP 名称，BARS 为信号值

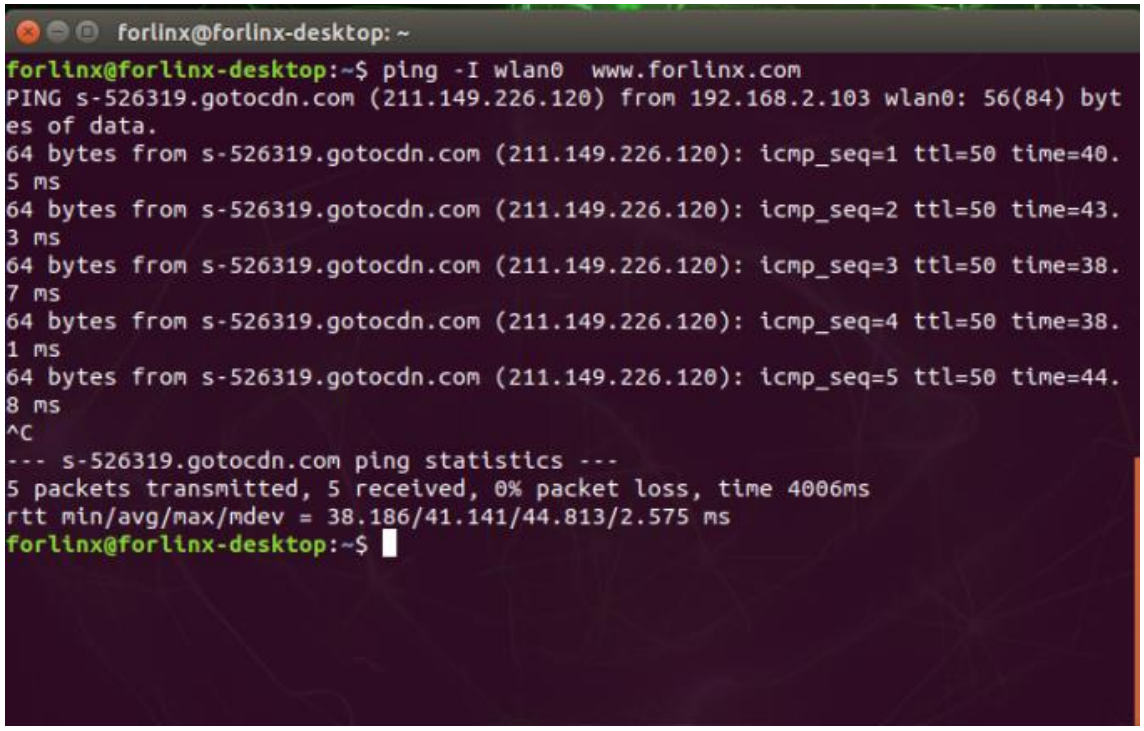
3.4.2.2 连接 AP

AP 名称为 bjfl ，密码为 123456785.

```
forlinx@forlinx-desktop:~$ sudo nmcli device wifi connect bjfl password "123456785." ifname wlan0
（需输入 sudo 密码）
```



测试连通性



3.4.3 AP 模式

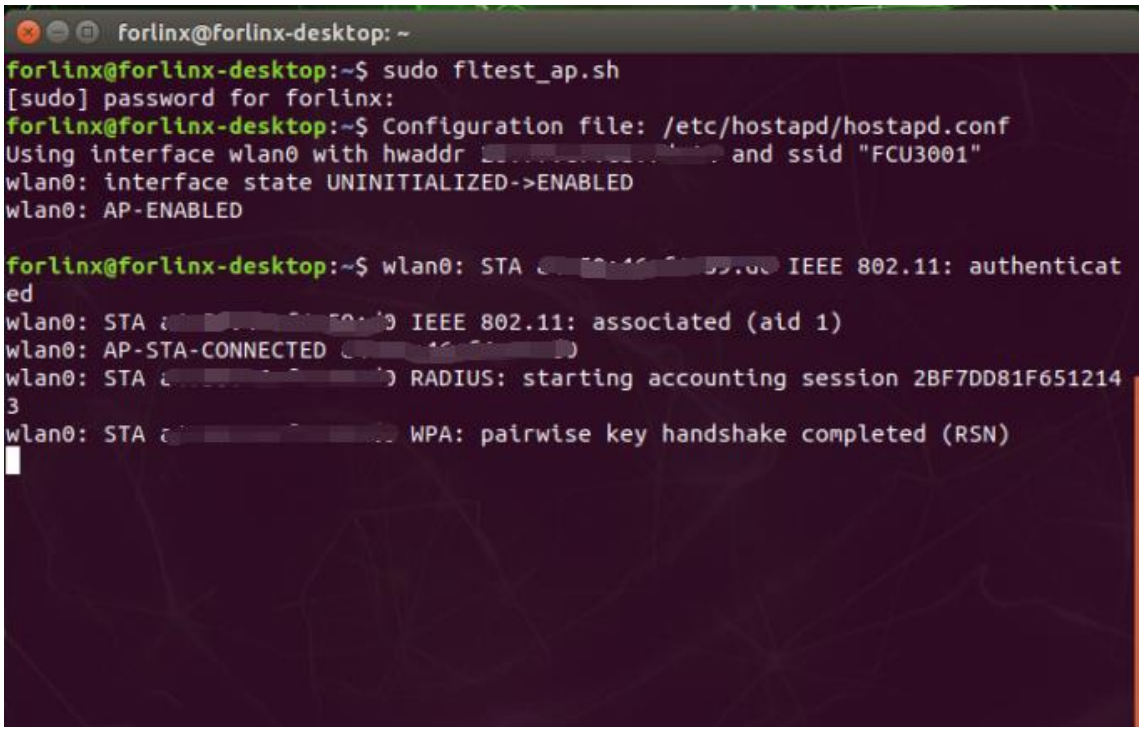
工作在 AP 模式下，手机等设备可以直接连接 FCU3001。如果 eth0 可以访问外网，那么通过该 AP 也可以访问外网。

如执行过 3.4.2 的命令连接过 AP，需要断开 AP 连接才能正常运行，断开命令如下,bjfl 为之前连接的 AP 名称：

```
forlinx@forlinx-desktop:~$ sudo nmcli connection down bjfl
```

执行 fltest_ap.sh，该脚本设置以太网 IP，配置网络防火墙，同时开启 AP：

```
forlinx@forlinx-desktop:~$ sudo fltest_ap.sh
```



手机等移动终端可以通过 WiFi 连接到 FCU3001 的 AP 热点，以下为热点名：FCU3001 密码：12345678 密码在/etc/hostapd/hostapd.conf 里可更改为其它。

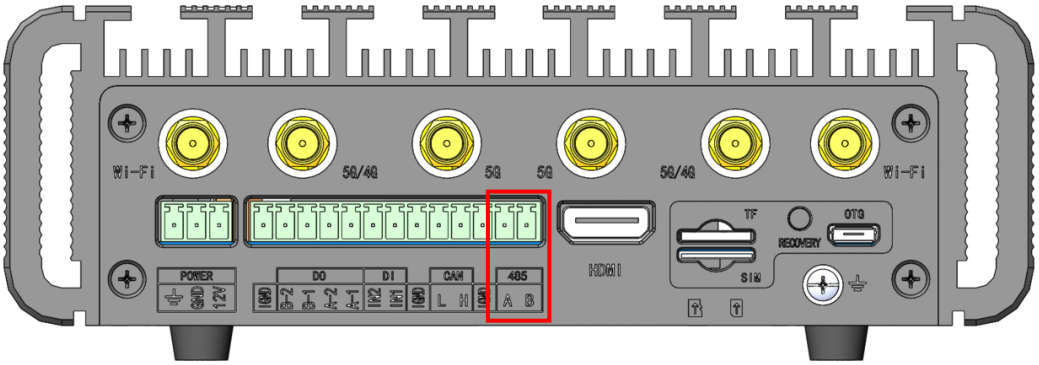
如果有设备连接 FCU3001，会打印如下信息：

```
wlan0: STA xx:xx:xx:xx:xx:xx IEEE 802.11: authenticated
wlan0: STA xx:xx:xx:xx:xx:xx IEEE 802.11: associated (aid 1)
wlan0: AP-STA-CONNECTED xx:xx:xx:xx:xx:xx
wlan0: STA xx:xx:xx:xx:xx:xx RADIUS: starting accounting session 2BF7DD81F6512143
wlan0: STA xx:xx:xx:xx:xx:xx WPA: pairwise key handshake completed (RSN)
```

（STA 的 MAC 根据实际设备变化，这里用 xx:xx:xx:xx:xx:xx 代替）

3.5 RS485

RS485 接口位置如图红框所示：



按照标记 A、B 和对端设备连接。

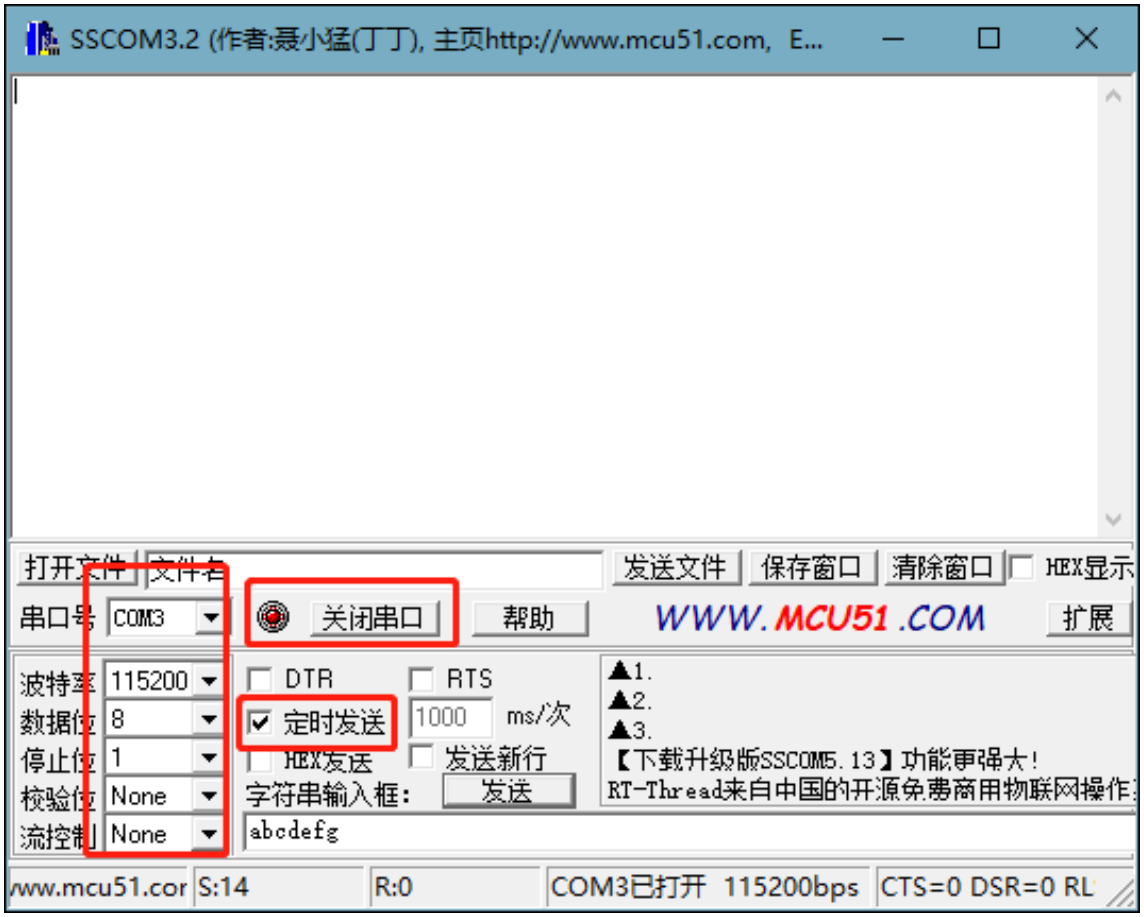
该接口在系统中对应的设备为/dev/ttyTHS0。如果是编程控制 RS485,方法同串口。

通过命令方式的连通性测试：

1、在 FCU3001 上使用自带命令 microcom 测试（也可以用 minicom）。

```
forlinx@forlinx-desktop:~$ sudo busybox microcom -s 115200 /dev/ttyTHS0
```

2、将 A、B 线连接到 USB 转 485 模块的引脚上。电脑端打开串口工具选择电脑识别的 COM 口，波特率 115200，数据位 8 位，停止位 1 位，无校验，无流控，1s 定时发送字符串 abcdefg，设置好参数后打开串口：



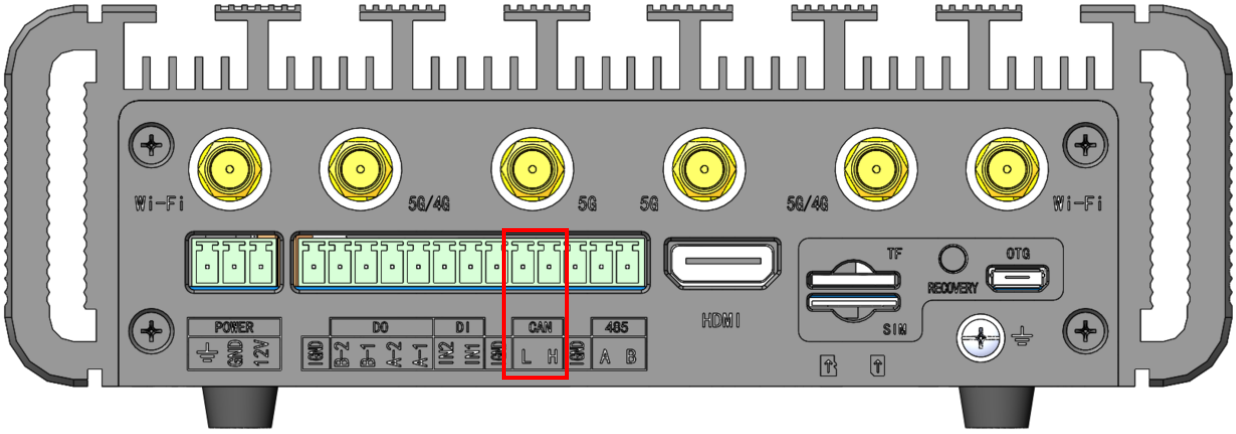
1、在 FCU3001 的 microcom 中输入字符，pc 工具能收到。用 PC 工具给 FCU3001 发送字符，microcom 中也能显示：

串口工具可接收到测试程序发送的数据。

microcosm 界面通过输入 ctrl+x 退出。

3.6 CAN

FCU3001 有一路 CAN 总线，按照 L、H 对应关系和对端设备连接。接口位置如下图：



连通性测试：

1、设置 can 服务如下：

```
forlinx@forlinx-desktop:~$ sudo modprobe mttcan
[sudo] password for forlinx:
forlinx@forlinx-desktop:~$ sudo ip link set can0 type can bitrate 125000
forlinx@forlinx-desktop:~$ sudo ifconfig can0 up
```

设置 can0 的 can 设备波特率为 125000

2、FCU3001 发送：

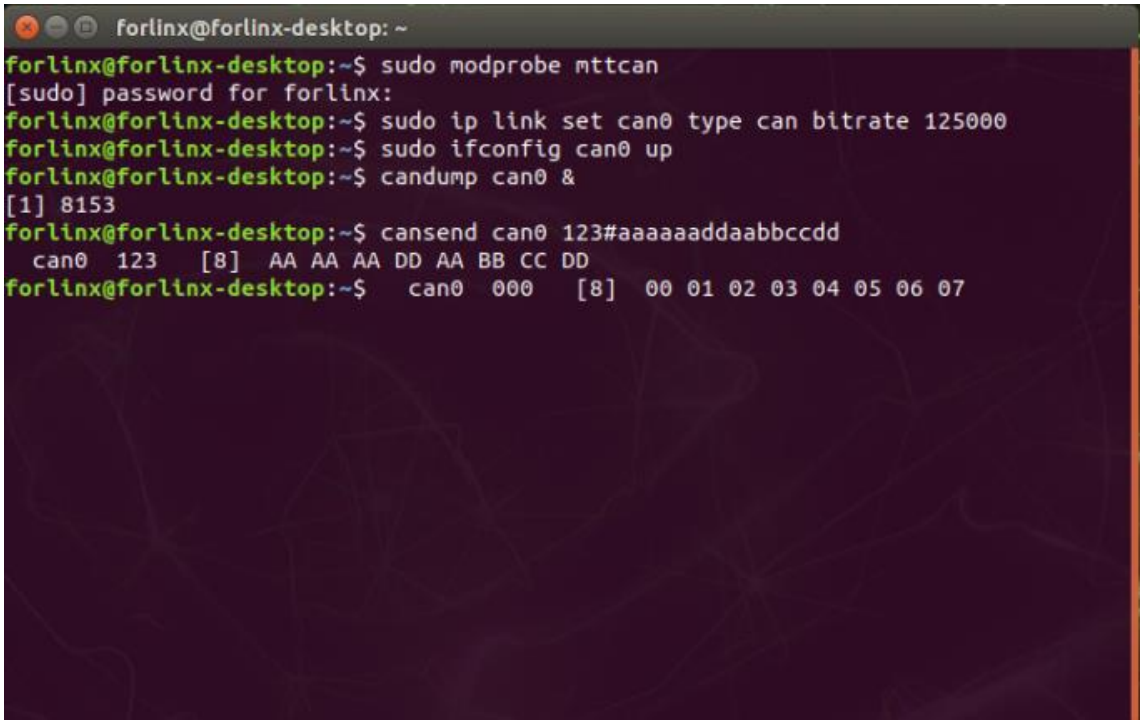
```
forlinx@forlinx-desktop:~$ cansend can0 123#aaaaaaddaabbccdd
```

3、FCU3001 接收：

```
forlinx@forlinx-desktop:~$ candump can0 &
```

CAN0 收到数据：

```
can0  000   [8]  00 01 02 03 04 05 06 07
```

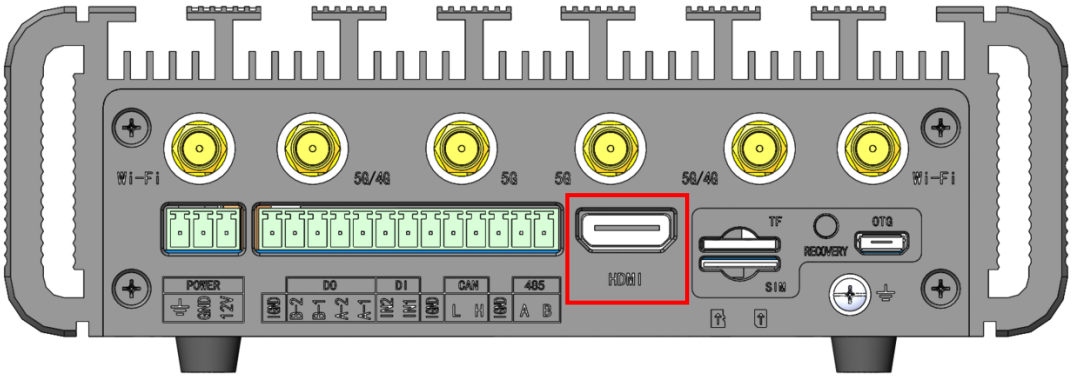


PC 端：

序号	传输方向	时间标识	帧ID	帧格式	帧类型	数据长度	数据(HEX)
00000000	接收	2751.7783	0x00000123	数据帧	标准帧	0x08	aa aa aa dd aa bb cc dd
00000001	发送	无	0x00000000	数据帧	标准帧	0x08	00 01 02 03 04 05 06 07

3.7 HDMI

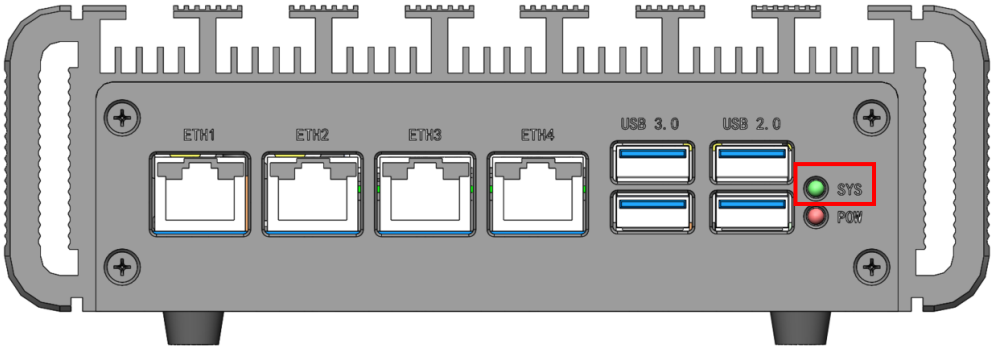
HDMI 接口如下图红框所示：



HDMI 支持自适应分辨率，最大支持到 4K 分辨率。

3.8 LED

FCU3001 仅有一个 LED 可供用户控制，位置如图：



1、点亮

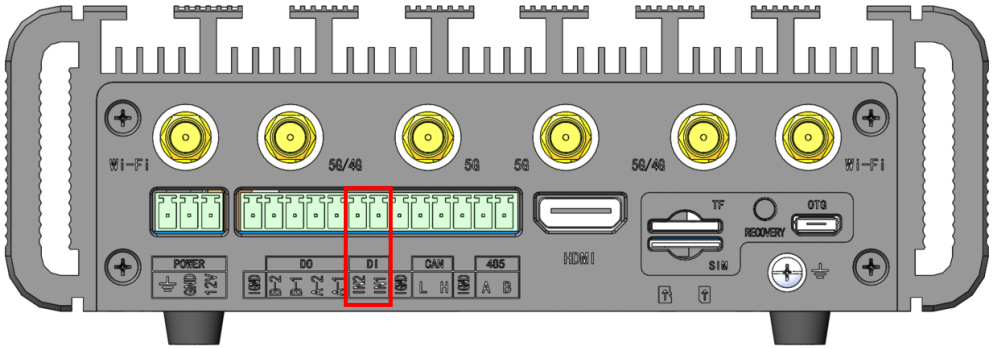
```
forlinx@forlinx-desktop:~$ fltest_led.sh on
```

2、熄灭

```
forlinx@forlinx-desktop:~$ fltest_led.sh off
```

3.9 DI

FCU3001 有两组数字量输入 IN1、IN2，公共端为 IGND。如下图



查询 DI1 状态

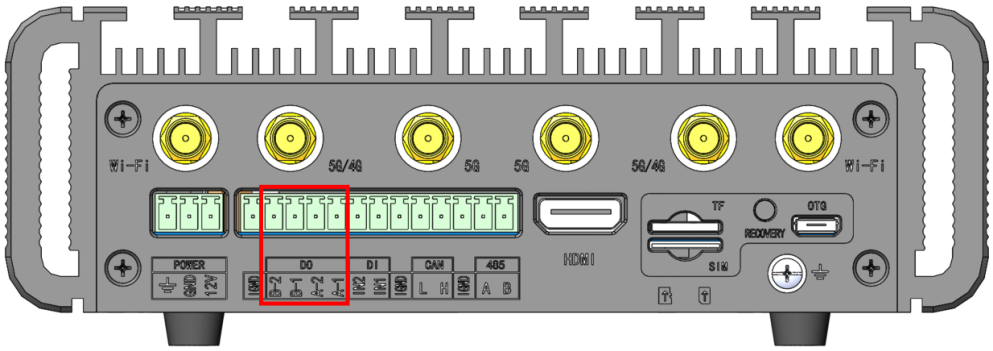
```
forlinx@forlinx-desktop:~$ fltest_di.sh IN1
IN1 state: 0
forlinx@forlinx-desktop:~$ fltest_di.sh IN1
IN1 state: 1
```

查询 DI2 状态

```
forlinx@forlinx-desktop:~$ fltest_di.sh IN2
IN1 state: 0
forlinx@forlinx-desktop:~$ fltest_di.sh IN2
IN1 state: 1
```

3.10 DO

FCU3001 有两组数字量输出，分别为 DO1、DO2



DO1 两个引脚：A-1、A-2
DO2 两个引脚：B-1、B-2

DO1 闭合：A-1 和 A-2 通

```
forlinx@forlinx-desktop:~$ fltest_do.sh DO1 on
```

DO1 断开：A-1 和 A-2 断

```
forlinx@forlinx-desktop:~$ fltest_do.sh DO1 off
```

DO2 闭合：B-1 和 B-2 通

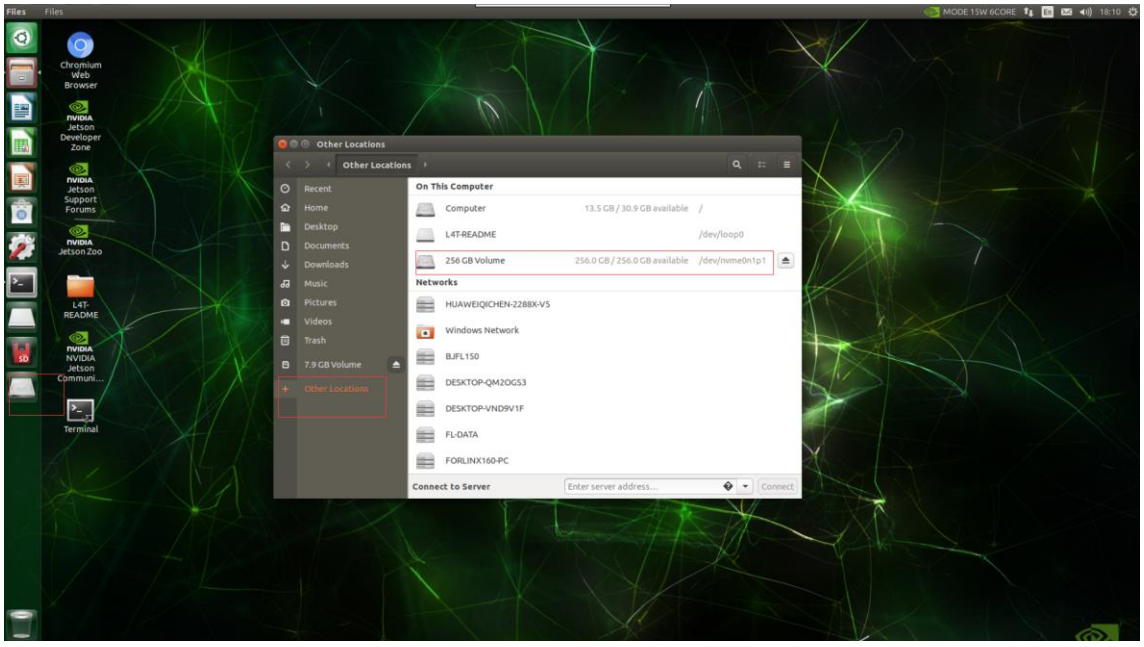
```
forlinx@forlinx-desktop:~$ fltest_do.sh DO2 on
```

DO2 断开：B-1 和 B-2 断

```
forlinx@forlinx-desktop:~$ fltest_do.sh DO2 off
```

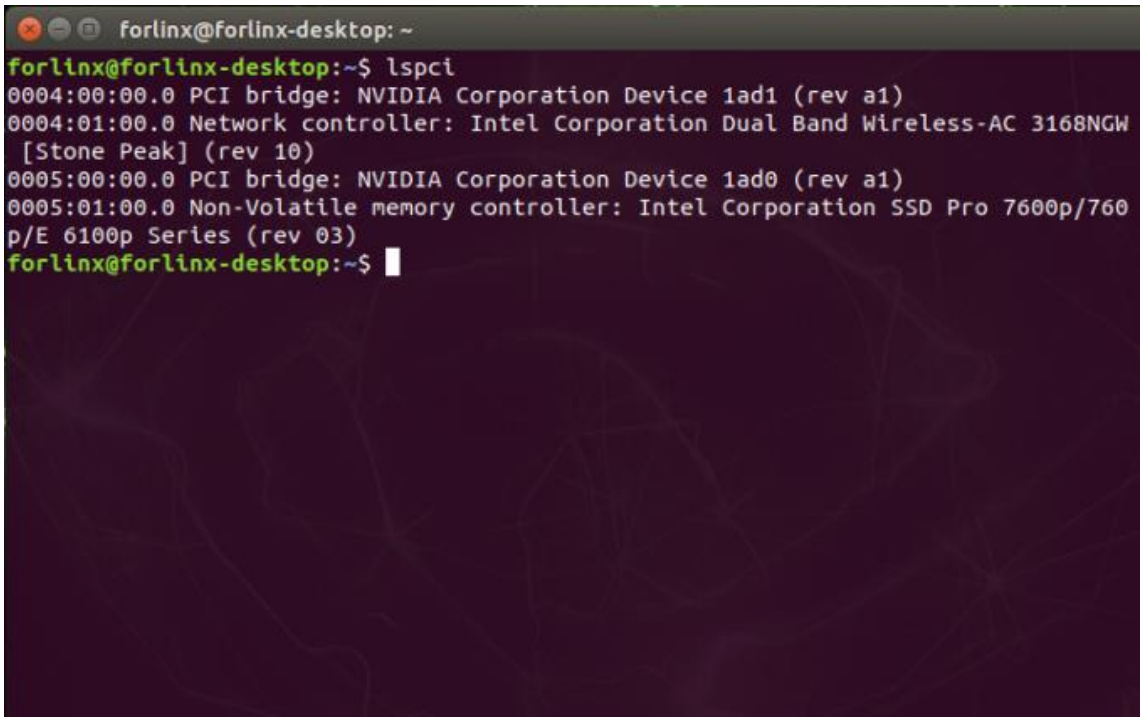
3.11 NVME

如果 FCU3001 内部安装了 NVME 固态硬盘，系统启动后会自动挂载，在图像桌面和命令行均可操作。验证过的型号：intel SSD 760p



命令行查看型号：

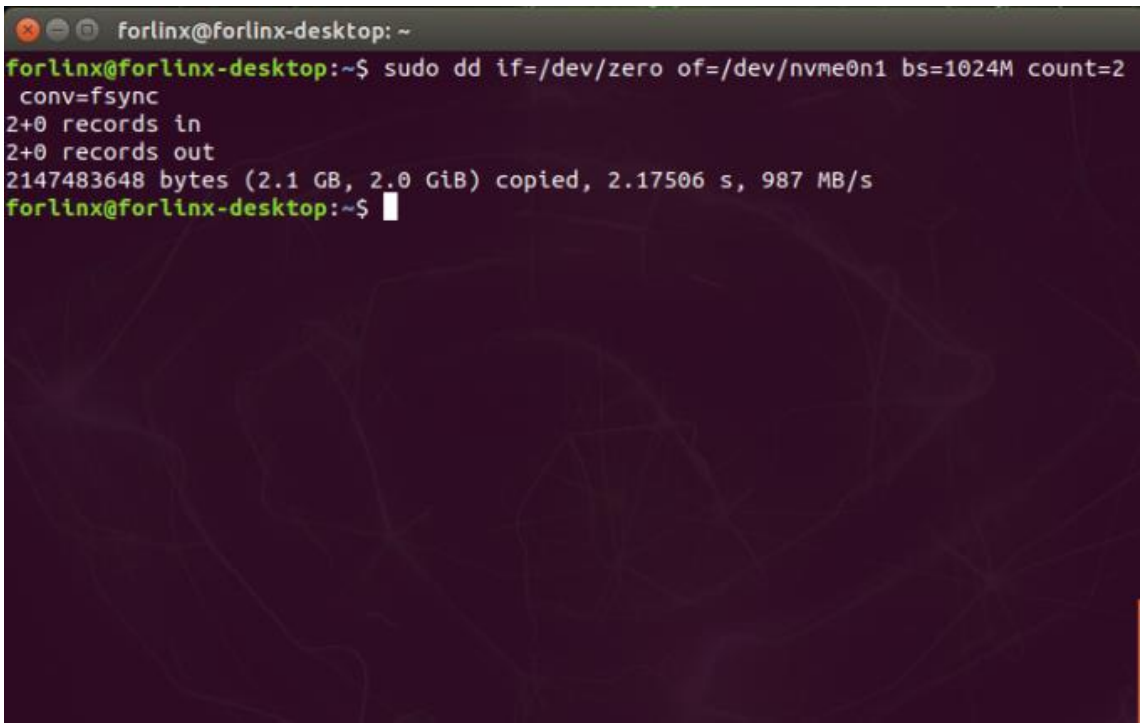
```
forlinx@forlinx-desktop:~$  lspci
0005:00:00.0 PCI bridge: NVIDIA Corporation Device 1ad0 (rev a1)
0005:01:00.0 Non-Volatile memory controller: Intel Corporation SSD Pro 7600p/760p/E 6100p Series (rev 03)
```



命令行速度测试（速度测试会损坏硬盘内数据，做好备份或者文件形式测试）：

写入：

```
forlinx@forlinx-desktop:~$ sudo dd if=/dev/zero of=/dev/nvme0n1 bs=1024M count=2 conv=fsync
2+0 records in
2+0 records out
2147483648 bytes (2.1 GB, 2.0 GiB) copied, 2.17506 s, 987 MB/s
```



读取（最好是重启系统后再测，能测到真实速度）：

```
forlinx@forlinx-desktop:~$  sudo dd if=/dev/nvme0n1 of=/dev/null bs=1M count=2048
[sudo] password for ubuntu:
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB, 2.0 GiB) copied, 2.06022 s, 1.0 GB/s
```

第四章 GPU Computing Applications & Deep Learning

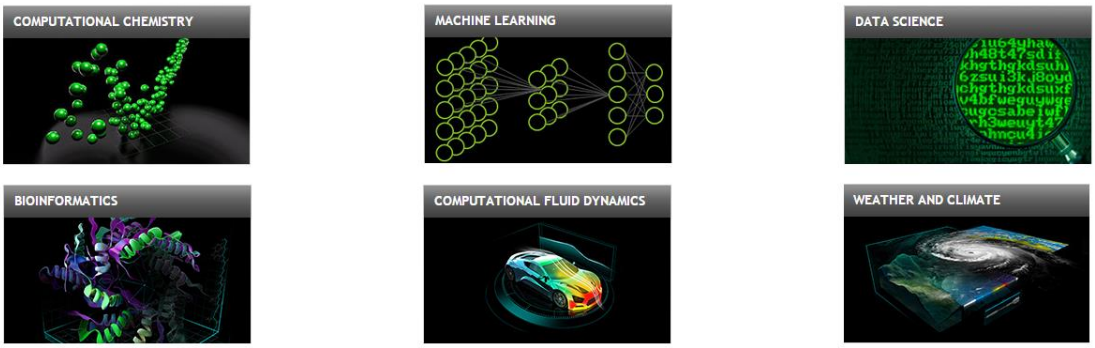
4.1 CUDA

CUDA® 是 NVIDIA 专为图形处理单元 (GPU) 上的通用计算开发的并行计算平台和编程模型。借助 CUDA，开发者能够利用 GPU 的强大性能显著加速计算应用。

在经 GPU 加速的应用中，工作负载的串行部分在 CPU 上运行，且 CPU 已针对单线程性能进行优化，而应用的计算密集型部分则以并行方式在数千个 GPU 核心上运行。使用 CUDA 时，开发者使用主流语言（如 C、C++、Fortran、Python 和 MATLAB）进行编程，并通过扩展程序以几个基本关键字的形式来表示并行性。

NVIDIA 的 CUDA 工具包提供了开发 GPU 加速应用所需的一切。CUDA 工具包中包含多个 GPU 加速库、一个编译器、多种开发工具以及 CUDA 运行环境。

CUDA 对多个领域中的应用实现了加速，从图像处理到深度学习、数字分析和计算科学。



CUDA 官方文档地址 https://docs.nvidia.com/cuda/#

在 FCU3001 上安装 CUDA。复制 FCU3001(ubuntu)用户资料\第四章 软件包\1 cuda_debs 目录到 U 盘中。复制完成后插到 FCU3001 上。

```
forlinx@forlinx-desktop:~$ sudo mount /dev/sda1    /mnt/
forlinx@forlinx-desktop:~$ cd /mnt/1\ cuda_debs/
forlinx@forlinx-desktop:/mnt/1 cuda_debs$ sudo dpkg -i    *.deb
```

运行 CUDA 中的图片降噪例程

```
forlinx@forlinx-desktop:~$ cd /usr/local/cuda-10.2/samples/
forlinx@forlinx-desktop:/usr/local/cuda-10.2/samples$ sudo make -j8
forlinx@forlinx-desktop:/usr/local/cuda-10.2/samples$ cd 3_Imaging/imageDenoising
forlinx@forlinx-desktop:/usr/local/cuda-10.2/samples/3_Imaging/imageDenoising$ ./imageDenoisin
g
CUDA ImageDenoising Starting...
```

```
[CUDA ImageDenoising]
Allocating host and CUDA memory and loading image file...
Loading ./data/portrait_noise.bmp...
BMP width: 320
BMP height: 408
BMP file loaded successfully!
Data init done.
Initializing GLUT...
OpenGL window created.
GPU Device 0: "Xavier" with compute capability 7.2

Creating GL texture...
Texture created.
Creating PBO...
PBO created.
Starting GLUT main loop...
Press [1] to view noisy image
Press [2] to view image restored with knn filter
Press [3] to view image restored with nlm filter
Press [4] to view image restored with modified nlm filter
Press [*] to view smooth/edgy areas [RED/BLUE] Ct's when a filter is active
Press [f] to print frame rate
Press [?] to print Noise and Lerp Ct's
Press [q] to exit
```

在连接到 FCU3001 的键盘上按 2 或 3 可以看到图像降噪效果，按 1 恢复到原图片。

4.2 cuDNN

NVIDIA CUDA® 深度神经网络库 (cuDNN) 是经 GPU 加速的深度神经网络基元库。cuDNN 可大幅优化标准例程（例如用于前向传播和反向传播的卷积层、池化层、归一化层和激活层）的实施。

世界各地的深度学习研究人员和框架开发者都依赖 cuDNN 实现高性能 GPU 加速。借助 cuDNN，研究人员和开发者可以专注于训练神经网络及开发软件应用，而不必花时间进行低层级的 GPU 性能调整。cuDNN 可加速广泛应用的深度学习框架，包括 Caffe2、Chainer、Keras、MATLAB、MxNet、PyTorch 和 TensorFlow。

在 FCU3001 上安装 cuDNN。将光盘资料内 FCU3001(ubuntu)用户资料\第四章 软件包\2 cudnn_debs 复制到 U 盘，复制完成后将 U 盘插到 FCU3001。

```
forlinx@forlinx-desktop:~$ sudo mount /dev/sda1    /mnt/
forlinx@forlinx-desktop:~$ cd /mnt/2\ cudnn_debs/
```

```
forlinx@forlinx-desktop:/mnt/2 cudnn_debs$ sudo dpkg -i  *.deb
forlinx@forlinx-desktop:/mnt/2 cudnn_debs$ sudo ln -s /usr/local/cuda-10.2/ /usr/local/cuda
```

运行卷积例程：

```
forlinx@forlinx-desktop:~$ cd /usr/src/cudnn_samples_v8/conv_sample/
forlinx@forlinx-desktop:/usr/src/cudnn_samples_v8/conv_sample$ sudo make
Linking agains cublasLt = true
CUDA VERSION: 10020
TARGET ARCH: aarch64
TARGET OS: linux
SMS: 30 35 50 53 60 61 62 70 72 75
/usr/local/cuda/bin/nvcc -ccbin g++ -I/usr/local/cuda/include
-I/usr/local/cuda/targets/aarch64-linux/include  -m64  -gencode
arch=compute_30,code=sm_30 -gencode arch=compute_35,code=sm_35 -gencode
arch=compute_50,code=sm_50 -gencode arch=compute_53,code=sm_53 -gencode
arch=compute_60,code=sm_60 -gencode arch=compute_61,code=sm_61 -gencode
arch=compute_62,code=sm_62 -gencode arch=compute_70,code=sm_70 -gencode
arch=compute_72,code=sm_72 -gencode arch=compute_75,code=sm_75 -gencode
arch=compute_75,code=compute_75 -o fp16_dev.o -c fp16_dev.cu
g++ -I/usr/local/cuda/include -I/usr/local/cuda/targets/aarch64-linux/include  -o fp16_emu.o -c
fp16_emu.cpp
g++ -I/usr/local/cuda/include -I/usr/local/cuda/targets/aarch64-linux/include  -o conv_sample.o
-c conv_sample.cpp
/usr/local/cuda/bin/nvcc -ccbin g++  -m64  -gencode arch=compute_30,code=sm_30
-gencode arch=compute_35,code=sm_35 -gencode arch=compute_50,code=sm_50 -gencode
arch=compute_53,code=sm_53 -gencode arch=compute_60,code=sm_60 -gencode
arch=compute_61,code=sm_61 -gencode arch=compute_62,code=sm_62 -gencode
arch=compute_70,code=sm_70 -gencode arch=compute_72,code=sm_72 -gencode
arch=compute_75,code=sm_75 -gencode arch=compute_75,code=compute_75 -o
conv_sample fp16_dev.o fp16_emu.o conv_sample.o -I/usr/local/cuda/include
-I/usr/local/cuda/targets/aarch64-linux/include -L/usr/local/cuda/lib64
-L/usr/local/cuda/targets/aarch64-linux/lib -lcublasLt -lcudart -lcublas -lcudnn -lstdc++ -lm
```

运行：

```
forlinx@forlinx-desktop:/usr/src/cudnn_samples_v8/conv_sample$ ./conv_sample
Executing: conv_sample
Using format CUDNN_TENSOR_NCHW (for INT8x4 and INT8x32 tests use
CUDNN_TENSOR_NCHW_VECT_C)
Testing single precision
====USER DIMENSIONS====
input dims are 1, 32, 4, 4
filter dims are 32, 32, 1, 1
output dims are 1, 32, 4, 4
```

```
====PADDING DIMENSIONS====
padded input dims are 1, 32, 4, 4
padded filter dims are 32, 32, 1, 1
padded output dims are 1, 32, 4, 4
Testing conv
^^^^ CUDA : elapsed = 2.83575 sec,
Test PASSED
Testing half precision (math in single precision)
====USER DIMENSIONS====
input dims are 1, 32, 4, 4
filter dims are 32, 32, 1, 1
output dims are 1, 32, 4, 4
====PADDING DIMENSIONS====
padded input dims are 1, 32, 4, 4
padded filter dims are 32, 32, 1, 1
padded output dims are 1, 32, 4, 4
Testing conv
^^^^ CUDA : elapsed = 0.000332832 sec,
Test PASSED
```

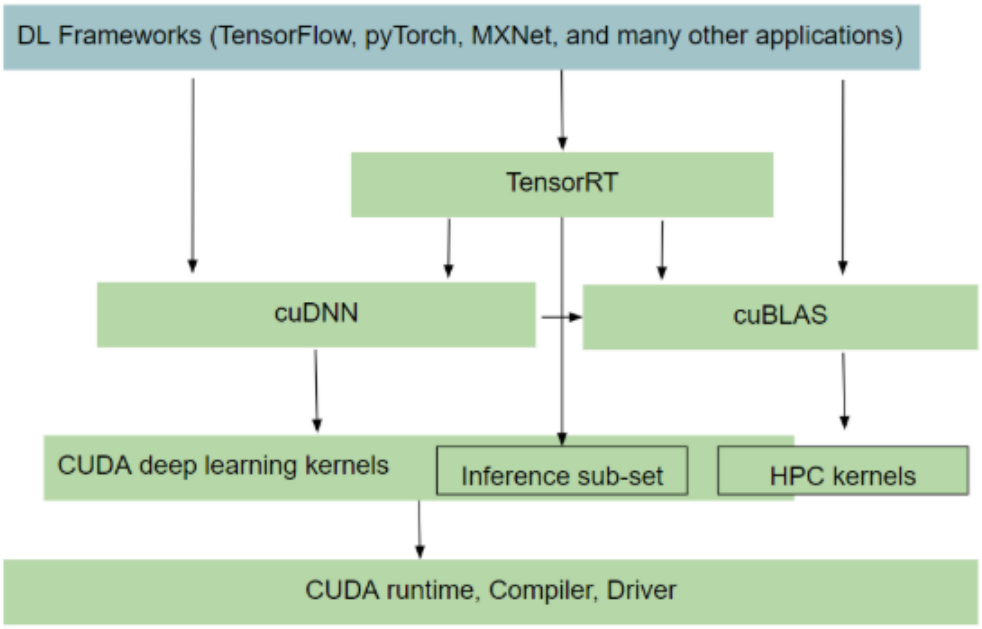
4.3 Tensorrt

NVIDIA TensorRT™ 是用于高性能深度学习推理的 SDK。此 SDK 包含深度学习推理优化器和运行时环境，可为深度学习推理应用提供低延迟和高吞吐量。

在推理过程中，基于 TensorRT 的应用程序的执行速度可比 CPU 平台的速度快 40 倍。借助 TensorRT，您可以优化在所有主要框架中训练的神经网络模型，精确校正低精度，并最终将模型部署到超大规模数据中心、嵌入式或汽车产品平台中。

TensorRT 以 NVIDIA 的并行编程模型 CUDA 为基础构建而成，可帮助您利用 CUDA-X 中的库、开发工具和技术，针对人工智能、自主机器、高性能计算和图形优化所有深度学习框架中的推理。

TensorRT 针对多种深度学习推理应用的生产部署提供 INT8 和 FP16 优化，例如视频流式传输、语音识别、推荐和自然语言处理。推理精度降低后可显著减少应用延迟，这恰巧满足了许多实时服务、自动和嵌入式应用的要求。



官方文档地址 https://docs.nvidia.com/deeplearning/tensorrt/index.html

在 FCU3001 上部署 Tensorrt（依赖于 CUDA，参考 4.1 CUDA，先安装 CUDA）

```
forlinx@forlinx-desktop:~$ sudo mount /dev/sda1    /mnt/
forlinx@forlinx-desktop:~$ cd /mnt/3\ tensorrt_debs/
forlinx@forlinx-desktop:/mnt/3 tensorrt_debs$ sudo dpkg -i    *.deb
```

运行手写字符识别测试程序：

```
forlinx@forlinx-desktop:~$ sudo tar xvf /usr/src/forlinx/mnist_data.tar.bz2 -C /usr/src/tensorrt/
forlinx@forlinx-desktop:~$ sudo tar xvf /usr/src/forlinx/mnist_pgm.tar.bz2 -C
/usr/src/tensorrt/data/mnist/
forlinx@forlinx-desktop:~$ cd /usr/src/tensorrt/samples/sampleMNIST
forlinx@forlinx-desktop:/usr/src/tensorrt/samples/sampleMNIST$ sudo make
../Makefile.config:10: CUDA_INSTALL_DIR variable is not specified, using /usr/local/cuda by
default, use CUDA_INSTALL_DIR=<cuda_directory> to change.
../Makefile.config:15: CUDNN_INSTALL_DIR variable is not specified, using /usr/local/cuda by
default, use CUDNN_INSTALL_DIR=<cudnn_directory> to change.
../Makefile.config:28: TRT_LIB_DIR is not specified, searching ../../lib, ../../lib, ../lib by default,
use TRT_LIB_DIR=<trt_lib_directory> to change.
if [ ! -d ../../bin/chobj/../common ]; then mkdir -p ../../bin/dchobj/../common; fi; :
Compiling: sampleMNIST.cpp
if [ ! -d ../../bin/chobj/../common ]; then mkdir -p ../../bin/dchobj/../common; fi; :
Compiling: ../common/sampleInference.cpp
if [ ! -d ../../bin/chobj/../common ]; then mkdir -p ../../bin/dchobj/../common; fi; :
```

```
Compiling: ../common/sampleOptions.cpp
^[[Cif [ ! -d ../../bin/chobj/../common ]; then mkdir -p ../../bin/dchobj/../common; fi; :
Compiling: ../common/logger.cpp
if [ ! -d ../../bin/chobj/../common ]; then mkdir -p ../../bin/dchobj/../common; fi; :
Compiling: ../common/getOptions.cpp
if [ ! -d ../../bin/chobj/../common ]; then mkdir -p ../../bin/dchobj/../common; fi; :
Compiling: ../common/sampleReporting.cpp
if [ ! -d ../../bin/chobj/../common ]; then mkdir -p ../../bin/dchobj/../common; fi; :
Compiling: ../common/sampleEngines.cpp
Linking: ../../bin/sample_mnist_debug
if [ ! -d ../../bin/chobj/../common ]; then mkdir -p ../../bin/chobj/../common; fi; :
Compiling: sampleMNIST.cpp
Linking: ../../bin/sample_mnist
# Copy every EXTRA_FILE of this sample to bin dir


forlinx@forlinx-desktop:/usr/src/tensorrt/samples/sampleMNIST$ cd /usr/src/tensorrt/bin/
forlinx@forlinx-desktop:/usr/src/tensorrt/bin$ ./sample_mnist --datadir=../data/mnist
&&&& RUNNING TensorRT.sample_mnist # ./sample_mnist --datadir=../data/mnist
[08/10/2021-12:56:22] [I] Building and running a GPU inference engine for MNIST
[08/10/2021-12:56:23] [I] [TRT]
[08/10/2021-12:56:23] [I] [TRT] ---------- Layers running on DLA:
[08/10/2021-12:56:23] [I] [TRT]
[08/10/2021-12:56:23] [I] [TRT] ---------- Layers running on GPU:
[08/10/2021-12:56:23] [I] [TRT] (Unnamed Layer* 9) [Constant], PWN((Unnamed Layer* 10)
[ElementWise], scale), conv1, pool1, conv2, pool2, ip1 + relu1, ip2, prob,
[08/10/2021-12:56:39] [I] [TRT] Detected 1 inputs and 1 output network tensors.
[08/10/2021-12:56:39] [I] Input:
@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@:.%@@@@@
@@@@@@@@@@@@@@@@%%%%* .@@@@@@
@@@@@@@@@@@@%-::.   =- :@@@@@@
@@@@@@@@@@@@+       -.    *@@@@@@
@@@@@@@@@@@=   -=@@#     @@@@@@@
@@@@@@@@@@@= .%@@%-   +@@@@@@@
@@@@@@@@@@@#. -@%:   +@@@@@@@@
@@@@@@@@@@@@- .*.   +@@@@@@@@@
@@@@@@@@@@@@=        -%@@@@@@@@@
@@@@@@@@@@@@+       :%@@@@@@@@@@
@@@@@@@@@@@@-     :@@@@@@@@@@@@
```

```
@@@@@@@@@@-      :@@@@@@@@@@@@@
@@@@@@@@%+   :- :@@@@@@@@@@@@@
@@@@@@@@*   +@* -@@@@@@@@@@@@@
@@@@@@@#   *@%. =@@@@@@@@@@@@
@@@@@@@= :@@+   +@@@@@@@@@@@@@
@@@@@@@= :@*   -@@@@@@@@@@@@@
@@@@@@@- -:   *@@@@@@@@@@@@@@
@@@@@@@+      =@@@@@@@@@@@@@@
@@@@@@@@+ :+@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

[08/10/2021-12:56:39] [I] Output:
0:
1:
2:
3:
4:
5:
6:
7:
8: **********
9:

&&&& PASSED TensorRT.sample_mnist # ./sample_mnist --datadir=../data/mnist
```

4.4 Hello AI World

将光盘资料内 FCU3001(ubuntu)用户资料\第四章 软件包\4 hello ai world 复制到 U 盘，复制完成后将 U 盘插到 FCU3001。（如果之前执行过 4.1 节、4.2 节内容，再执行本节安装可能导致空间不足问题，需要手工删除 4.1 节、4.2 节安装的包，或者参考第七章重新刷系统恢复到原始状态）

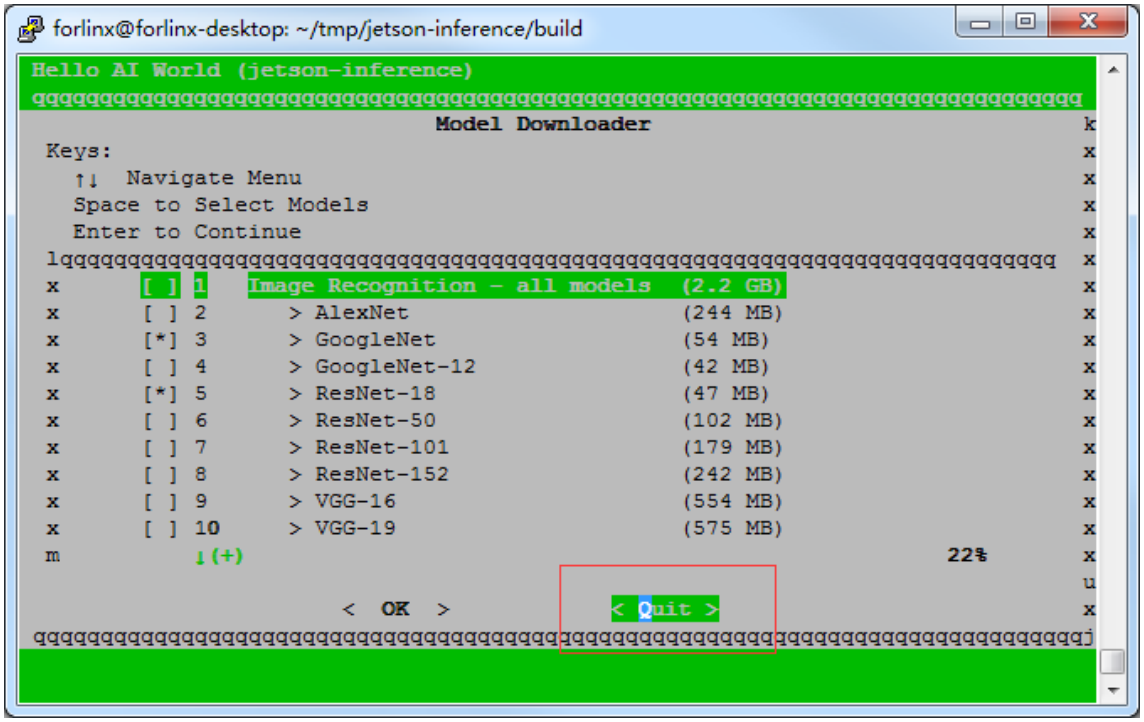
```
forlinx@forlinx-desktop:~$ sudo mount /dev/sda1 /mnt/
[sudo] password for forlinx:
forlinx@forlinx-desktop:~$ sudo dpkg -i /mnt/4\ hello\ ai\ world/interface_debs/*.deb

forlinx@forlinx-desktop:~$ tar xvf /mnt/4\ hello\ ai\ world/jetson-inference_with_model.tar.bz2
-C .
forlinx@forlinx-desktop:~$ cd jetson-inference
forlinx@forlinx-desktop:~/jetson-inference$ mkdir build
forlinx@forlinx-desktop:~/jetson-inference$ cd build
```

```
forlinx@forlinx-desktop:~/jetson-inference/build$ cmake ..
```

cmake 配置过程中会尝试通过网络更新安装包。如果没有连接网线，等待超时后自动进入后续步骤，不影响代码完整性。

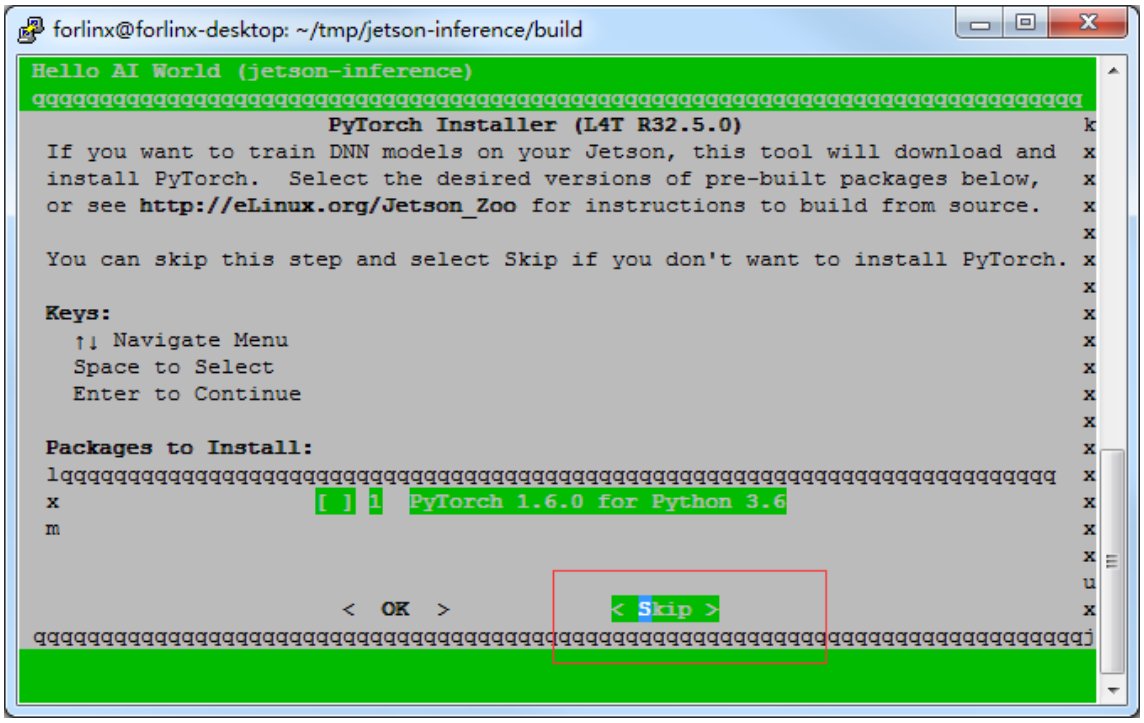
cmake 配置过程会弹出 Model Downloader 界面。如果需要用该工具下载模型需要联网，并且能够访问 google。默认安装包内已经预装了三个模型，满足测试。如需要其它模型可以后续自行下载，复制到系统即可。

所以下图通过键盘方向右键将光标移动至<Quit>，按回车后自动进行后续步骤。

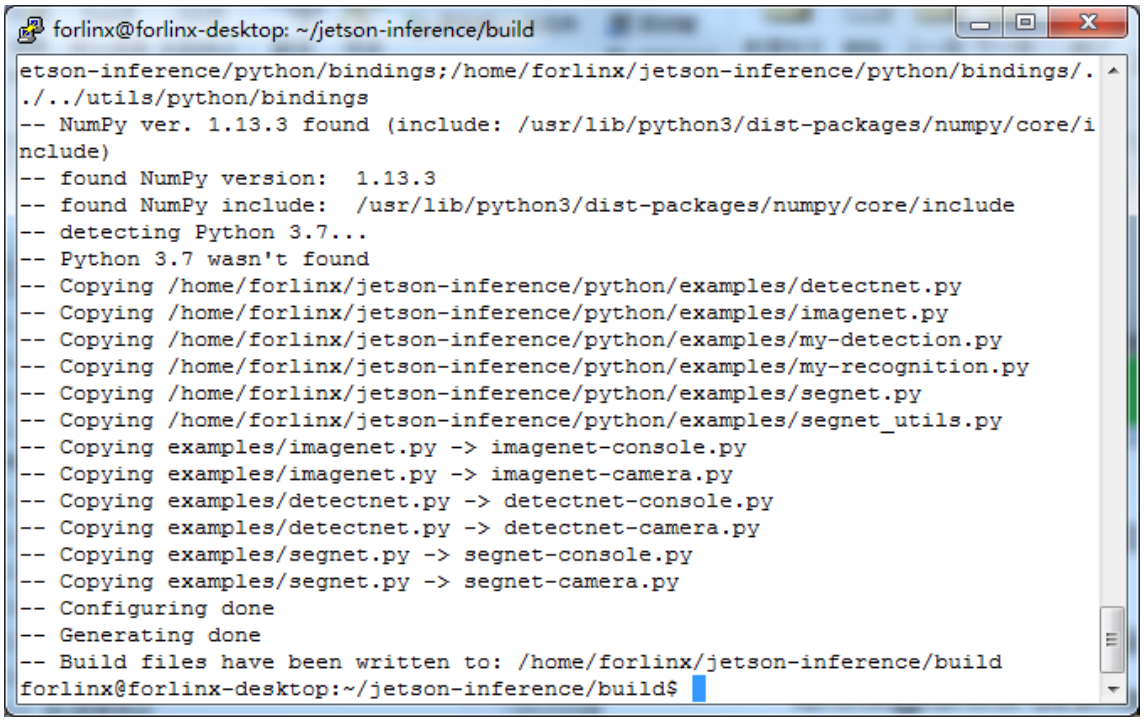


cmake 过程会弹出是否安装 PyTorch，同理通过键盘移动光标至<Skip>后输入回车继续后续步骤。

如果需要在 jetson 上训练模型可以选择安装，但一般情况下都是在 PC 或者云上训练模型。



配置完成后输出如下信息：



输入如下命令开始编译：

forlinx@forlinx-desktop:~/jetson-inference/build$ make -j8

输出如下信息代表编译正确且完成：

```
Scanning dependencies of target camera-capture_autogen
[ 87%] Automatic MOC for target camera-capture
[ 88%] Building CXX object examples/segnet/CMakeFiles/segnet-console.dir/segnet.cpp.o
[ 89%] Building CXX object python/bindings_python_3.6/CMakeFiles/jetson-inference-python-36.dir/PySegNet.cpp.o
[ 90%] Building CXX object python/bindings_python_3.6/CMakeFiles/jetson-inference-python-36.dir/PyInference.cpp.o
[ 92%] Building CXX object python/bindings_python_3.6/CMakeFiles/jetson-inference-python-36.dir/PyDetectNet.cpp.o
[ 92%] Building CXX object python/bindings_python_3.6/CMakeFiles/jetson-inference-python-36.dir/PyTensorNet.cpp.o
[ 93%] Building CXX object python/bindings_python_3.6/CMakeFiles/jetson-inference-python-36.dir/PyImageNet.cpp.o
[ 94%] Linking CXX executable ../../aarch64/bin/segnet-console
[ 94%] Linking CXX shared library ../../aarch64/lib/python/3.6/jetson_inference_python.so
[ 94%] Built target segnet-console
[ 94%] Built target camera-capture_autogen
Scanning dependencies of target camera-capture
[ 94%] Built target jetson-inference-python-36
[ 95%] Building CXX object tools/camera-capture/CMakeFiles/camera-capture.dir/captureWindow.cpp.o
[ 95%] Building CXX object tools/camera-capture/CMakeFiles/camera-capture.dir/camera-capture.cpp.o
[ 96%] Building CXX object tools/camera-capture/CMakeFiles/camera-capture.dir/controlClassify.cpp.o
[ 97%] Building CXX object tools/camera-capture/CMakeFiles/camera-capture.dir/controlDetection.cpp.o
[ 98%] Building CXX object tools/camera-capture/CMakeFiles/camera-capture.dir/camera-capture_autogen/mocs_compilation.cpp.o
[ 99%] Building CXX object tools/camera-capture/CMakeFiles/camera-capture.dir/controlWindow.cpp.o
[100%] Linking CXX executable ../../aarch64/bin/camera-capture
[100%] Built target camera-capture
forlinx@forlinx-desktop:~/jetson-inference/build$
```

安装：

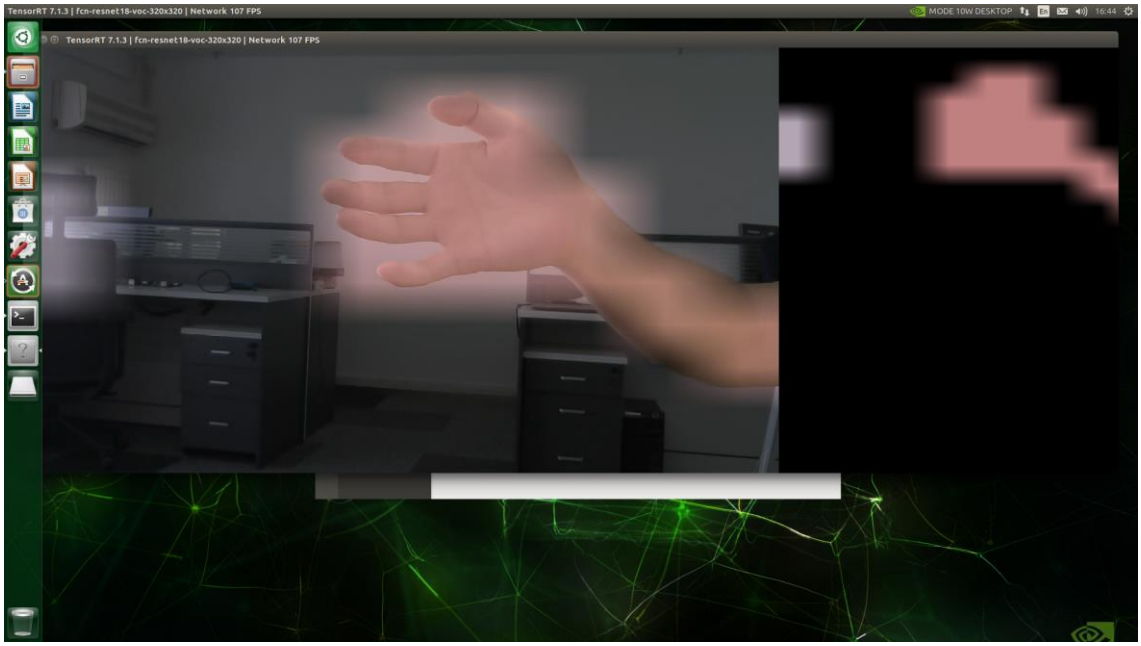
```
forlinx@forlinx-desktop:~/jetson-inference/build$ sudo make install
forlinx@forlinx-desktop:~/jetson-inference/build$ sudo ldconfig
```

运行测试程序：

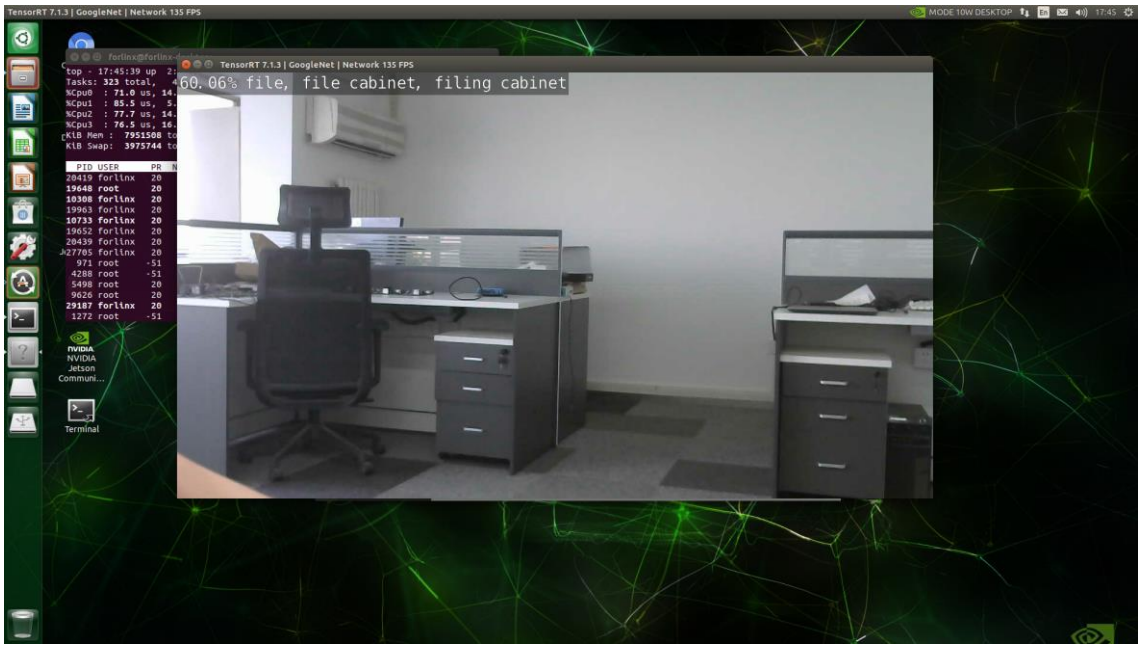
将 usb 摄像头插入到 FCU3001 usb 口，执行程序。第一次执行测试程序会创建 TensorRT 的 engine 缓存，需要数分钟时间，同一个模型文件只需要创建一次缓存。不指定模型的情况下使用预置模型，其它模型需自行下载并通过命令行指定。

```
forlinx@forlinx-desktop:~/jetson-inference/build$ detectnet-camera --camera=/dev/video0
```


```
forlinx@forlinx-desktop:~/jetson-inference/build$ segnet-camera --camera=/dev/video0
```



```
forlinx@forlinx-desktop:~/jetson-inference/build$ imagenet-camera --camera=/dev/video0
```



4.5 TensorFlow

TensorFlow 依赖于 CUDART，参考 4.1 节安装 CUDA，否则无法使用 GPU 加速。

本节提供网络安装方法。离线安装包在光盘中，在无网络条件的情况下自习手工安装。

```
forlinx@forlinx-desktop:~$ sudo apt-get update
forlinx@forlinx-desktop:~$ sudo apt-get install libhdf5-serial-dev hdf5-tools libhdf5-dev
zlib1g-dev zip libjpeg8-dev liblapack-dev libblas-dev gfortran

forlinx@forlinx-desktop:~$ sudo apt-get install python3-pip
```

```
forlinx@forlinx-desktop:~$ sudo pip3 install -U pip testresources setuptools==49.6.0

forlinx@forlinx-desktop:~$ sudo pip3 install -U numpy==1.19.4 future==0.18.2 mock==3.0.5
h5py==2.10.0 keras_preprocessing==1.1.1 keras_applications==1.0.8 gast==0.2.2 futures
protobuf pybind11

forlinx@forlinx-desktop:~$ sudo pip3 install --pre --extra-index-url
https://developer.download.nvidia.com/compute/redist/jp/v45 tensorflow
```

安装验证，成功载入 cudart 库并且无报错表示安装成功：

```
forlinx@forlinx-desktop:~$ python3
Python 3.6.9 (default, Jan 26 2021, 15:33:00)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow
2021-08-24 20:14:14.302159: I tensorflow/stream_executor/platform/default/dso_loader.cc:53]
Successfully opened dynamic library libcudart.so.10.2
>>> exit()
```

输入 exit()可退出 python3 命令行。

离线包所在路径为：
FCU3001(ubuntu)用户资料\第四章 软件包\5 TensorFlow

第五章 多媒体

FCU3001 具有非常优秀的视频编解码性能，特性参数如下表：

视频解码

Standard	Profile(s)	Resolution (Max Number of Streams)	Throughput (up to)
H.264	Baseline, Main, High,	(2x) 2160p60 | (4x) 2160p30 | (8x) 1080p60 | (16x) 1080p30	(2x) 560 MP/s
	High 444, High 444 Predictive, MVC (per view)	(1x) 2160p60 | (2x) 2160p30 | (4x) 1080p60 | (8x) 1080p30	(2x) 280 MP/s
			(Max Throughput half for YUV444 as compared to YUV420)
HEVC	Main, Main 10	(0x) 4320p30 | (3x) 2160p60 | (6x) 2160p30 | (12x) 1080p60 | (24x) 1080p30	(2x) 920 MP/s
	Main 444, Main 444 10, MV	(1x) 2160p60 | (3x) 2160p30 | (6x) 1080p60 | (12x) 1080p30	(2x) 460 MP/s
			(Max Throughput half for YUV444 as compared to YUV420)
VP9	Profile 0	(2x) 2160p60 | (4x) 2160p30 | (10x) 1080p60 | (20x) 1080p30	(2x) 700 MP/s

视频编码：

Standard	Profile(s)	Resolution (Max Number of Streams)	Throughput (up to)
H.265 (HEVC)	Main, Main 10	2160p30 (2) | 1080p60 (6) | 1080p30 (14)	(2x) 464 MP/s
	Main 4:4:4, Main 4:4:4 10, MV (per view)	1080p60 (2) | 1080p30 (6)	(2x) 232 MP/s
H.264	Baseline, Main, High	2160p30 (2) | 1080p60 (6) | 1080p30 (14)	(2x) 455 MP/s
	High 444, High 444 Predictive, MVC (per view)	1080p60 (2) | 1080p30 (6)	(2x) 227 MP/s
VP9	profile 0	2160p30 (2) | 1080p60 (4) | 1080p30 (8)	(2x) 278 MP/s

测试素材位于：

FCU3001(ubuntu)用户资料\第五章 素材\test_av

在 PC 端复制 test_av 目录到 U 盘后，将 U 盘插入 FCU3001，并执行手工挂载命令：

```
forlinx@forlinx-desktop:~$ sudo mount /dev/sda1 /mnt
```

该章节所有测试均基于/mnt 下内容执行（即 U 盘）。

5.1 视频解码

视频解码使用 nvidia 硬件加速

以光盘资料内视频为例：

H265

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 \
filesrc location=/mnt/test_av/video/1080p_30fps_h265.mp4 !  \
qtdemux name=demux demux.video_0 ! queue ! h265parse ! omxh265dec !  \
nvoverlaysink -e
```

H264

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 \
filesrc location=/mnt/test_av/video/1080p_60fps_h264.mp4 !  \
qtdemux name=demux demux.video_0 ! queue ! h264parse ! omxh264dec !  \
nvoverlaysink -e
```

VP8

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 \
filesrc location=/mnt/test_av/video/1080p_30fps_vp8.webm  !  \
matroskademux name=demux demux.video_0 ! queue ! omxvp8dec !  \
nvoverlaysink -e
```

VP9

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 \
filesrc location=/mnt/test_av/video/1080p_30fps_vp9.webm !  \
matroskademux name=demux demux.video_0 ! queue ! omxvp9dec !  \
nvoverlaysink -e
```

MPEG-4

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 \
filesrc location=/mnt/test_av/video/1080p_60fps_m4v.mp4 !  \
qtdemux  ! queue ! mpeg4videoparse ! nvv4l2decoder !  \
nvoverlaysink -e
```

其它参考：

10-bit H.265 Decode (NVIDIA Accelerated Decode)

```
gst-launch-1.0 filesrc location=<filename_10bit.mkv> ! \
matroskademux ! h265parse ! omxh265dec ! nvvidconv ! \
'video/x-raw(memory:NVMM), format=(string)NV12' ! \
nvoverlaysink -e
```

12-bit H.265 Decode (NVIDIA Accelerated Decode)

```
gst-launch-1.0 filesrc location=<filename_12bit.mkv> ! \
matroskademux ! h265parse ! omxh265dec ! nvvidconv ! \
'video/x-raw(memory:NVMM), format=(string)NV12' ! \
nvoverlaysink -e
```

5.2 视频编码

细节参数可查看具体 gst 插件帮助文档

H.264 Encode

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 videotestsrc !    \
'video/x-raw, format=(string)I420, width=(int)640, \
height=(int)480' ! omxh264enc ! \
'video/x-h264, stream-format=(string)byte-stream' ! h264parse ! \
qtmux ! filesink location=test.mp4 -e
```

H.265 Encode

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 videotestsrc !    \
'video/x-raw, format=(string)I420, width=(int)640, \
height=(int)480' ! omxh265enc ! filesink location=test.h265 -e
```

10-bit H.265 Encode

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 videotestsrc !    \
'video/x-raw, width=(int)1920, height=(int)1080, framerate=(fraction)30/1'    ! \
omxh265enc ! matroskamux ! \
filesink location=test_10bit.mkv -e
```

VP9 Encode

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 videotestsrc ! \
'video/x-raw, format=(string)I420, width=(int)640, \
height=(int)480' ! omxvp9enc ! matroskamux ! \
filesink location=test.mkv -e
```

MPEG-4 Encode (软编)

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 videotestsrc ! \
'video/x-raw, format=(string)I420, width=(int)640, \
height=(int)480' ! avenc_mpeg4 ! qtmux ! \
filesink location=test.mp4 -e
```

5.3 JPEG

JEPG

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 \
filesrc location=/mnt/test_av/image/test.jpg ! nvjpegdec !     \
imagefreeze ! xvimagesink -e
```

Image Encode

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 videotestsrc num-buffers=1 ! \
'video/x-raw, width=(int)640, height=(int)480, \
```

```
format=(string)I420' ! nvjpegenc ! filesink location=test.jpg -e
```

5.4 音频解码（软解）

使用光盘资料内 test_av 目录下文件播放：

仅播放视频中的音频

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 filesrc
location=/mnt/test_av/video/1080p_30fps_h265.mp4 ! \
qtdemux name=demux demux.audio_0 !     \
queue ! avdec_aac ! audioconvert ! alsasink -e
```

播放 MP3 文件

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 filesrc location=/mnt/test_av/audio/test.mp3 ! \
mpegaudioparse !     avdec_mp3 ! audioconvert ! alsasink -e
```

其它参考格式如下：

AAC Decode (OSS Software Decode)

```
gst-launch-1.0 filesrc location=<filename.mp4> ! \
   qtdemux name=demux demux.audio_0 ! \
   queue ! avdec_aac ! audioconvert ! alsasink -e
```

AMR-WB Decode (OSS Software Decode)

```
gst-launch-1.0 filesrc location=<filename.mp4> ! \
   qtdemux name=demux demux.audio_0 ! queue ! avdec_amrwb ! \
   audioconvert ! alsasink -e
```

AMR-NB Decode (OSS Software Decode)

```
gst-launch-1.0 filesrc location=<filename.mp4> ! \
   qtdemux name=demux demux.audio_0 ! queue ! avdec_amrnb ! \
   audioconvert ! alsasink -e
```

MP3 Decode (OSS Software Decode)

```
gst-launch-1.0 filesrc location=<filename.mp3> ! mpegaudioparse ! \
   avdec_mp3 ! audioconvert ! alsasink -e
```

5.5 音频编码（软编）

AAC Encode

```
forlinx@forlinx-desktop:~$    gst-launch-1.0 audiotestsrc ! \
'audio/x-raw, format=(string)S16LE,
layout=(string)interleaved, rate=(int)44100, channels=(int)2' ! \
voaacenc ! qtmux ! filesink location=test.mp4 -e
```

AMR-WB Encode

```
forlinx@forlinx-desktop:~$  gst-launch-1.0 audiotestsrc ! \
'audio/x-raw, format=(string)S16LE, layout=(string)interleaved, \
rate=(int)16000, channels=(int)1' ! voamrwbenc ! qtmux ! \
filesink location=test.mp4 -e
```

5.6 UVC 摄像头

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 v4l2src device="/dev/video0" ! \
"video/x-raw, width=640, height=480, format=(string)YUY2" ! \
xvimagesink -e
```

5.7 CUDA

CUDA 视频后处理

```
forlinx@forlinx-desktop:~$ gst-launch-1.0 filesrc  \
location=/mnt/test_av/video/1080p_30fps_h265.mp4 !  \
qtdemux name=demux demux.video_0 ! queue ! h265parse !  omxh265dec !  \
videocuda ! nveglglessink max-lateness=-1 -e
```

第六章 杂项

6.1 看门狗参考

看门狗提供了在程序跑飞情况下的硬件复位功能，使用看门狗需要 sudo 权限。

复位测试：

执行 60s 复位命令：。

```
forlinx@forlinx-desktop:~$ sudo fltest_wdt /dev/watchdog settimeout 60          //执行60s 复位命令
```

又例如执行 10s 复位命令：

```
forlinx@forlinx-desktop:~$ sudo fltest_wdt /dev/watchdog settimeout 10          //执行10s 复位命令
```

打印信息如下：

```
forlinx@forlinx-desktop:~$ sudo fltest_wdt /dev/watchdog0 settimeout 10
[sudo] password for forlinx:
reboot after 10
reboot after 9
reboot after 8
reboot after 7
reboot after 6
reboot after 5
reboot after 4
reboot after 3
reboot after 2
reboot after 1
```

守护模式，1 秒周期喂狗，不发生重启。如该程序退出 10 内重启（10 表示 10 秒）

```
forlinx@forlinx-desktop:~$ sudo fltest_wdt   /dev/watchdog keepalive 10
```

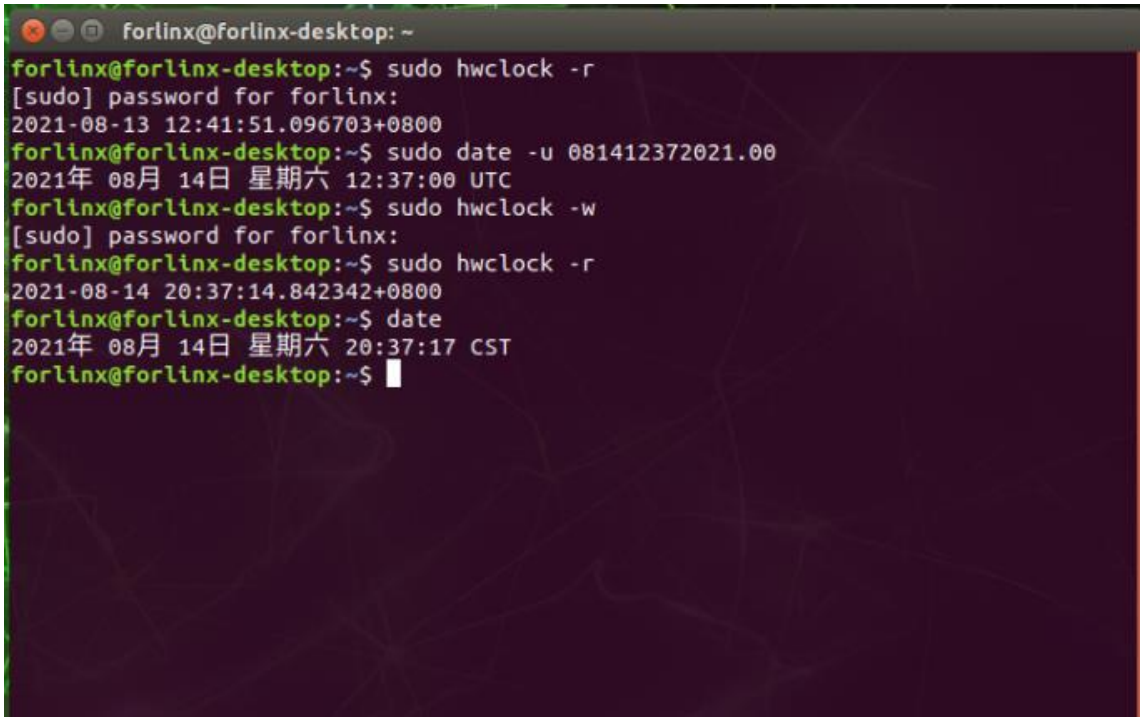
6.2 RTC 时钟驱动测试

⚠ **注意：确保板子上已经安装了纽扣电池，并且电池电压正常**

RTC 测试，主要通过使用 date 和 hwclock 工具设置软、硬件时间，测试当板子断电再上电的时候，软件时钟读取 RTC 时钟是否同步

```
forlinx@forlinx-desktop:~$ date -u 072216162021.00          //设置软件时间
```

```
forlinx@forlinx-desktop:~$ sudo hwclock –r                    #读取硬件RTC时间
[sudo] password for forlinx:
2021-08-13 12:41:51.096703+0800
forlinx@forlinx-desktop:~$ sudo date -u 081412372021.00       #设置新系统时间
2021年 08月 14日 星期六 12:37:00 UTC
forlinx@forlinx-desktop:~$ sudo hwclock –w                    #写入硬件RTC
[sudo] password for forlinx:
forlinx@forlinx-desktop:~$ sudo hwclock –r                    #读取RTC时间
2021-08-14 20:37:14.842342+0800
forlinx@forlinx-desktop:~$ date                               #读取系统时间
2021年 08月 14日 星期六20:37:17 CST
```



然后给板子断电再上电，进入系统后读取系统时间，可以看到时间已经同步。

注意：系统默认开启NTP服务，在有外网访问权限的情况下会自动对时。如需关闭，需执行 sudo systemctl disable systemd-timesyncd.service

6.3 CPU 调频

一般情况下无需修改，按照默认即可。以下举例将默认调频模式改为高性能模式：

1、当前内核中支持的所有 cpufreq governor 类型：

```
forlinx@forlinx-desktop:~$ cat
/sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors
```

```
interactive conservative ondemand userspace powersave performance schedutil
```

2、改为高性能模式

先创建脚本 performance.sh，内容为：

```
echo performance > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

```
forlinx@forlinx-desktop:~$ echo "echo performance >
/sys/devices/system/cpu/cpu0/cpufreq/scaling_governor" > performance.sh
forlinx@forlinx-desktop:~$ cat performance.sh
echo performance > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
forlinx@forlinx-desktop:~$ chmod +x performance.sh
forlinx@forlinx-desktop:~$ sudo ./performance.sh
```

6.4 温度

1、查看 CPU 当前温度值：

```
forlinx@forlinx-desktop:~$ cat /sys/class/thermal/thermal_zone0/temp
57000                                          //温度值即为 57℃（57000/1000）
```

2、查看 GPU 当前温度值

```
forlinx@forlinx-desktop:~$ cat /sys/class/thermal/thermal_zone1/temp
57000                                          //温度值 57℃
```

6.5 开机自启设置

在文件系统的/etc/autorun.sh 为设置应用开机自启的脚本，用户可以在该脚本中加入自己需要开机自启动的程序。

使用 systemctl status autorun 可以查看服务运行状态：

```
forlinx@forlinx-desktop:~$ systemctl status autorun
● autorun.service - auto run something
   Loaded: loaded (/lib/systemd/system/autorun.service; enabled; vendor preset: enabled)
   Active: inactive (dead) since Thu 2021-08-26 18:08:39 CST; 5min ago
 Main PID: 5571 (code=exited, status=0/SUCCESS)

8 月  26 18:08:39 forlinx-desktop systemd[1]: Started auto run something.
8 月  26 18:08:39 forlinx-desktop autorun.sh[5571]: autorun script, put your code below
```

6.6 logo 替换

6.6.1 第一阶段 logo

刷机包内提供 logo 制作工具，只支持 16、24、32bit 的 bmp 图片格式，不能用索引图。

🗁 FCU3001(ubuntu)用户资料\第七章 刷机工具\OTG 刷机\Linux_for_Tegra.tar.bz2

1. 将 Linux_for_Tegra.tar.bz2 从光盘资料中复制到 PC 机的 ubuntu 中，解压

```
ubuntu@ubuntu:~$ tar xvf Linux_for_Tegra.tar.bz2
ubuntu@ubuntu:~$ cd Linux_for_Tegra/tools/bmp-splash/
```

2. 复制 800x480 分辨率的 bmp 文件到该目录，命名为 logo480.bmp
   复制完成后当前目录下文件为：
   bmp-blob-README.txt　BMP_generator_L4T.py　config.file　config_file.example
   genbmpblob_L4T.sh　logo480.bmp
3. 执行制作命令

```
ubuntu@ubuntu:~$ OUT=$PWD ./genbmpblob_L4T.sh
t194 ./config.file ./BMP_generator_L4T.py /usr/bin/lz4c my-bmp.blob
BMP IMAGE INFO     : ./logo480.bmp nvidia 480
1+0 records in
1+0 records out
1 byte (1 B) copied, 6.9902e-05 s, 14.3 kB/s
1+0 records in
1+0 records out
1 byte (1 B) copied, 5.5702e-05 s, 18.0 kB/s
1+0 records in
1+0 records out
1 byte (1 B) copied, 5.6097e-05 s, 17.8 kB/s
1+0 records in
1+0 records out
1 byte (1 B) copied, 5.4529e-05 s, 18.3 kB/s
./genbmpblob_L4T.sh: Success! bmp.blob is in 'my-bmp.blob'.
```

   完成后会在该目录下生成 my-bmp.blob

4. 复制 my-bmp.blob 到 fcu3001，写入到存储中（举例假设复制到了 fcu3001 的 mnt 目录下）

```
forlinx@forlinx-desktop:~$ sudo dd if=/mnt/my-bmp.blob of=/dev/mtdblock0 bs=1K seek=15488
forlinx@forlinx-desktop:~$ sync
```

5. 重新上电

6.6.2 登录 logo

登录 logo 位于 FCU3001 文件系统内/usr/share/backgrounds/NVIDIA_Login_Logo.png，将 NVIDIA_Login_Logo.png 替换为合适的图片即可。

登录输入用户名和密码，敲回车即可显示该图片。

6.6.3 桌面背景

FCU 桌面背景位于/usr/share/backgrounds/NVIDIA_Wallpaper.jpg，替换为合适的 png 图片即可。

第七章 重新刷机

在 FCU3001 原系统被破坏的情况下可以参考该章节重新刷机。FCU3001 目前支持 U 盘和 OTG 两种刷机方式。在用户资料/Linux/刷机工具文件夹中提供了 OTG 和 U 盘的刷机工具，用户可选择任意一种方式进行刷机。

FCU3001 如果被损坏了启动引导软件（QSPI 内）即 U 盘刷机不能正常刷机的情况下，需要用 OTG 刷机。

不可将第三方镜像刷入系统，否则 U 盘刷机功能失效。用 OTG 刷机将飞凌镜像再次刷入，可恢复支持 U 盘刷机。

4.1 U 盘刷机

4.1.1 制作刷机 U 盘

在 PC 机上将 U 盘格式化为 FAT32 格式。

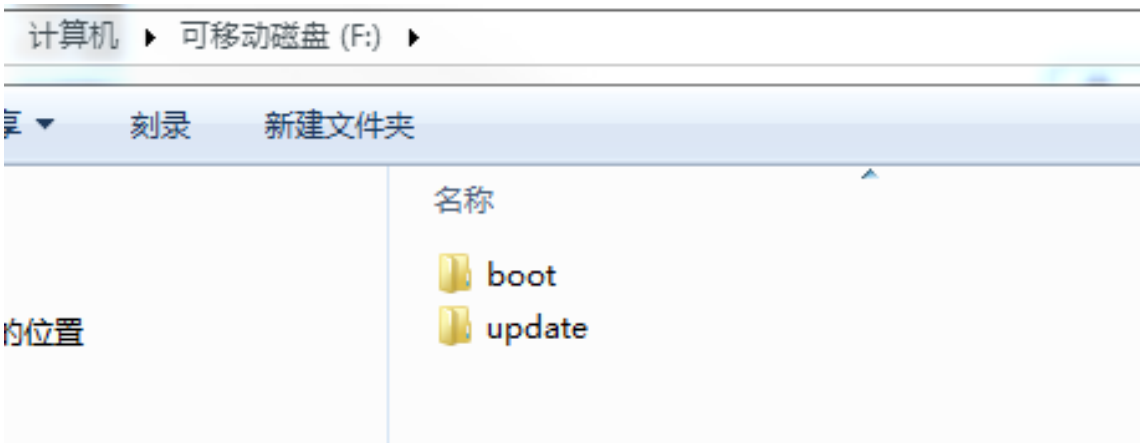
应保证文件系统为第一个分区，即分区号为 1。一般情况下默认操作均第一个分区，但使用第三方格式工具或者 linux 上手工指定过分区号均可能导致文件系统分区号不是 1，这种情况下不能用作刷机 U 盘，需要重新手工在 linux 格式化。典型情况是使用 Universal Usb Installer 处理过的 U 盘，在 win 上右键格式化后不能支持刷机功能，需要删掉分区重新创建。

4.1.2 拷贝刷机镜像

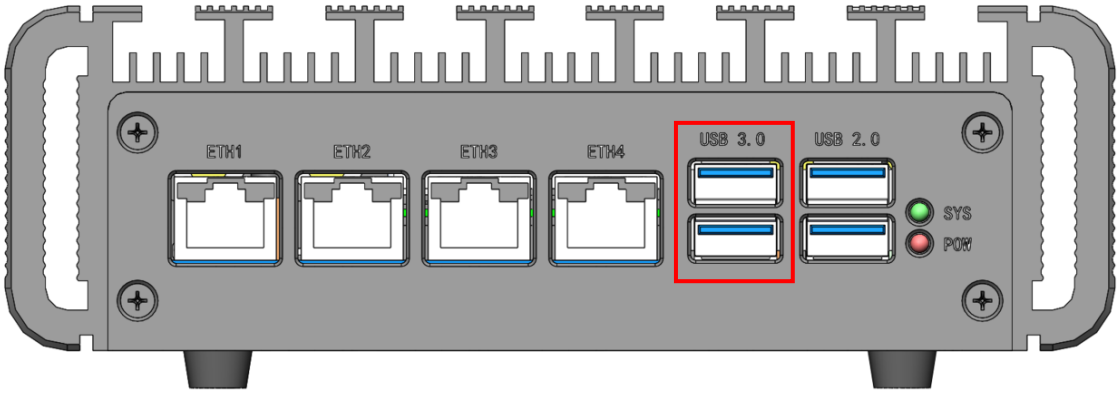
刷机工具内含有刷机镜像，位于：

🗁 用户资料\第七章 刷机工具\U 盘刷机

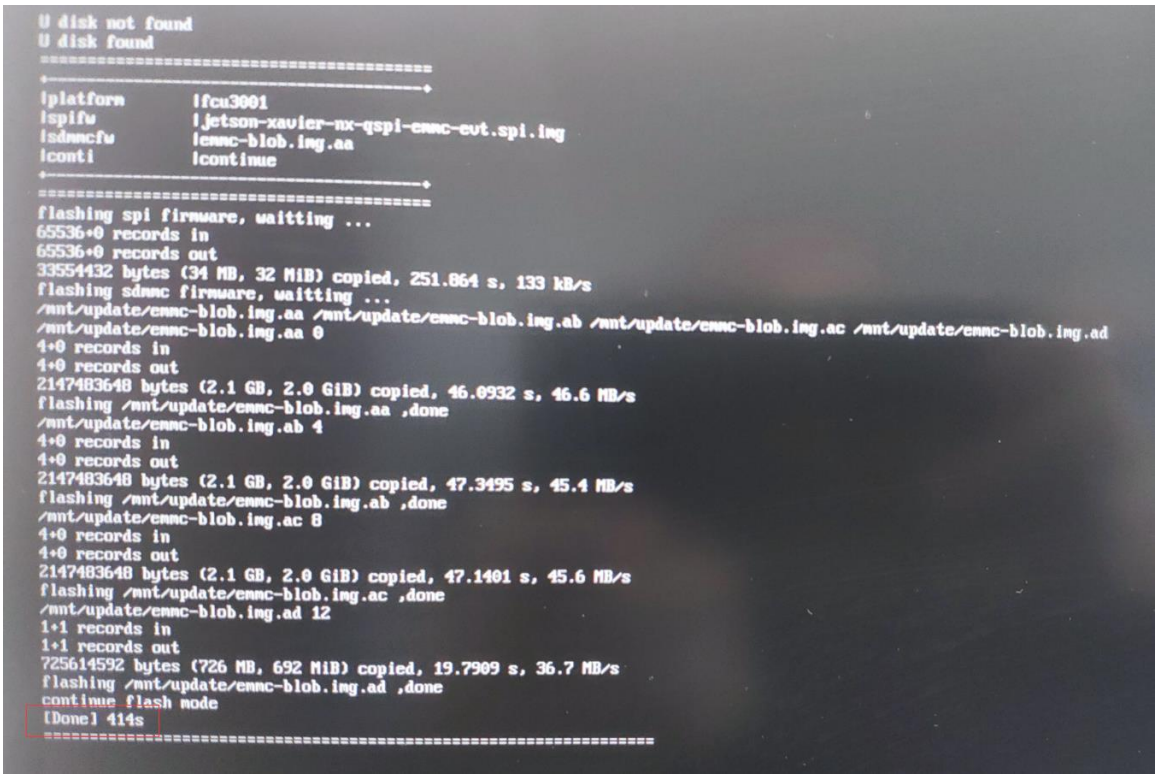
将该目录下的 boot 和 update 文件夹拷贝到 U 盘内。



将 U 盘插入 USB3.0 的任意口，对 FCU3001 上电，自动开始刷机。数分钟后提示刷机完成。断电，拔下 U 盘，再上电。参考第一章首次启动重新配置系统。



整个刷机过程如下图，出现[Done]表示传输完成。刷机过程中红色LED常亮；刷机完成后红色LED闪烁。



刷机完成后，拔出U盘，重新上电（不拔出U盘重新上电会再次进入刷机）。

4.1.2 OTG 刷机

在 U 盘更新系统无法使用的情况下，可以使用 USB 线缆通过 PC 机刷机。需要在 ubuntu 下进行，ubuntu18.04 为最佳。可以在 win 上装虚拟机 ubuntu，避免装真机。

刷机包位于:

🗁 FCU3001(ubuntu)用户资料\第七章 刷机工具\OTG 刷机\Linux_for_Tegra.tar.bz2

1. 将 Linux_for_Tegra.tar.bz2 从光盘资料中复制到 PC 机的 ubuntu 中，解压

```
ubuntu@ubuntu:~$ tar xvf Linux_for_Tegra.tar.bz2
```

2. 按下 Recovery 按键后，重新给 FCU3001 上电，上电后松开 Recovery 按键。
3. 将 OTG 接口通过 Micro USB 线缆连接至 PC 机。
4. 在 PC 端执行

```
ubuntu@ubuntu:~$ cd Linux_for_Tegra
ubuntu@ubuntu:~/Linux_for_Tegra $ sudo ./fcu3001_flash.sh
```

刷机完成后，FCU3001 自动重启进入系统，参考“第二章 首次开机”做初始化设置。

声明

本手册版权归保定飞凌嵌入式技术有限公司所有。未经本公司的书面许可，任何单位和个人无权以任何形式复制、传播、转载本手册的任何部分，违者将被追究法律责任。

保定飞凌嵌入式有限公司所提供的所有服务内容旨在协助客户加速产品的研发进度，在服务过程中所提供的任何程序、文档、测试结果、方案、支持等资料和信息，都仅供参考，客户有权不使用或自行参考修改，本公司不提供任何的完整性、可靠性等保证，若在客户使用过程中因任何原因造成的特别的、偶然的或间接的损失，本公司不承担任何责任。

更多帮助

注意事项与维护

- 请勿带电插拔核心板及外围模块！
- 请遵循所有标注在产品上的警示和指引信息。
- 请保持本产品干燥。如果不慎被任何液体泼溅或浸润，请立刻断电并充分晾干
- 使用中注意本产品的通风散热，避免温度过高造成元器件损坏。
- 请勿在多尘、脏乱的环境中使用或存放本产品。
- 请勿将本产品应用在冷热交替环境中，避免结露损坏元器件。
- 请勿粗暴对待本产品，跌落、敲打或剧烈晃动都可能损坏线路及元器件。
- 请勿使用有机溶剂或腐蚀性液体清洗本产品。
- 请勿自行修理、拆卸本公司产品，如产品出现故障请及时联系本公司进行维修。
- 擅自修改或使用未经授权的配件可能损坏本产品，由此造成的损坏将不予以保修。

资料的更新

产品相关资料会不断的完善更新，本手册内容亦然如此；当您在使用这些内容时，请确保其为最新状态。

飞凌嵌入式产品资料更新通知采用微信公众号推送，敬请关注！

资料获取

1. 请登录飞凌官方论“bbs.witech.com.cn”→“开发板资料下载”选择对应平台下载；
2. 下载前请阅读《资料下载说明》：http://bbs.witech.com.cn/thread-67932-1-1.html。

售后服务政策

1. 如产品使用过程中出现硬件故障可根据售后服务政策进行维修；
2. 服务政策：参见官方网站 www.forlinx.com 售后服务说明；

送修地址

1. 地 址：河北省保定市高开区向阳北大街 2699 号飞凌嵌入式技术有限公司新楼五层售后维修部
2. 联系人：售后维修部
3. 电 话：0312-3102650-952、953
4. 邮编：071000
5. 邮寄须知：建议使用顺丰、圆通或韵达，且不接收任何到付。

技术支持范围

1. 本公司产品的软、硬件资源提供情况咨询；
2. 本公司产品的软、硬件手册使用过程中遇到的问题；

技术讨论范围

1. 源码的修改以及理解；
2. 操作系统如何移植；
3. 用户自行修改以及开发中遇到的软硬件问题；

注：以上三点虽不属于技术支持范围，但我公司会尽力为用户提供帮助，如仍然没能解敬请谅解。

技术支持方式

1. 电话：0312-3119192
2. 论坛：bbs.witech.com.cn
3. 邮箱：

   Linux 技术支持： linux@forlinx.com

   Android 技术支持： android@forlinx.com

   硬件技术支持： hardware@forlinx.com
4. 知识库：bbs.witech.com.cn/kb

技术支持时间

1. 周一至周五：上午 9:00—11:30；

   下午 13:30—17:00；
2. 公司按照国家法定节假日安排休息，在此期间无法提供技术支持，请将问题发送至邮箱或论坛技术支持区，我们会在工作日尽快给您回复。