飞腾教育开发板用户手册 V2.6

天乾 C216F

文件修订记录

版本	日期	修订描述	修订人
V0.1	2022.1.10~2022.1.26	初始版本	\
V1.0	2022.1.27	发布版本	\
V2.0	2022.2.24	修改支持系统和相应 demo	\
V2.1	2022.2.28	增加固件、系统升级和开发 demo	\
V2.2	2022.3.2	删除麒麟内核编译部分	\
V2.3	2022.3.7	增加 WIFI 操作	\
V2.4	2022.3.9	增加 WIFI 型号、debian 安装方法和 MNN 推理	\
V2.5	2022.4.8	修改文档错误	\

目录

一、 飞腾教育开发板套件 ..... - 5 -
1.1 飞腾教育开发板配置 ..... - 5 -
1.2 硬件规格 ..... - 5 -
1.3 飞腾教育开发板正面图 ..... - 6 -
1.4 飞腾教育开发板 PCB 规格 ..... - 7 -
1.5 系统组件 ..... - 7 -
二、 配件安装 ..... - 8 -
2.1 安装 MXM 显卡 ..... - 8 -
2.2 安装硬盘 ..... - 9 -
三、 启动飞腾教育开发板 ..... - 11 -
3.1 开发板串口连接说明 ..... - 11 -
3.2 使用 SSH 登录开发板 ..... - 13 -
3.3 开发板关机流程 ..... - 13 -
四、 外置接口 ..... - 15 -
4.1 前面板功能接口 ..... - 15 -
4.2 后面板功能接口 ..... - 15 -
4.3 RJ45 网口 ..... - 15 -
4.4 电源接口 ..... - 16 -
4.5 RTC 实时时钟 ..... - 16 -
4.6 存储接口 ..... - 17 -
4.7 WIFI 模块 ..... - 18 -
五、 BIOS 在线升级和系统安装 ..... - 21 -
5.1 BIOS 升级固件 ..... - 21 -
5.2 启动盘刻录 ..... - 22 -
5.3 DEBIAN 系统安装 ..... - 24 -
5.4 银河麒麟系统安装 ..... - 33 -
5.5 TIGERVNC 局域网远程桌面 ..... - 35 -
5.6 PYTHON 更新 3.7 ..... - 37 -
5.7 GCC 更新 9.4.0 ..... - 38 -
六、 使用开发板在线开发程序 ..... - 41 -

6.1 搭建 QT 环境 ..... - 41 -
6.2 搭建 OPENCV 环境 ..... - 43 -
6.3 安装 MINICOM ..... - 45 -
6.4 在线开发 TCP 程序 ..... - 47 -
6.5 在线开发 CANNY 实例 ..... - 57 -
6.6 在线开发 LAPLACE 实例 ..... - 59 -
6.7 在线开发 SOBEL 实例 ..... - 61 -
6.8 FT2000/4 平台实时时钟实例 ..... - 63 -
七、 MNN 推理引擎 ..... - 64 -
7.1 MNN 介绍 ..... - 64 -
7.2 MNN 运行实例 ..... - 64 -
八、 常见问题解决 ..... - 66 -
7.1 调试串口没有打印信息！ ..... - 66 -
7.2 自动获取 IP 地址报错！ ..... - 66 -
7.3 MXM 显卡无法显示图像？ ..... - 66 -
7.4 SSH 无法远程登录问题？ ..... - 66 -
九、 支持与服务 ..... - 67 -
十、 免责声明 ..... - 67 -
十一、 版权公告 ..... - 67 -

一、 飞腾教育开发板套件

1.1 飞腾教育开发板配置

序号	货物名称	类别	配置	数量
1	天 乾 C216F 飞腾教育开发板套件（整机形态，卧式云终端）	主板	》CPU：FT-2000 V2，4 核，2.6GHz 》内存：2 个 SODIMM 插槽，单条最大支持 16GB DDR4，支持 DDR4-1600/DDR4-2666 》存储：支持 SATA、M.2 和 mSATA 存储 》显示：1 个 MXM 插槽 》接口：6 个 USB3.0 接口，1 个 HDMI 接口，1 个 DP 接口、2 个千兆网口，1 个音频接口 》电源：DC19V 》尺寸：195x170mm 》固件：国产自主可控 BIOS 》工作温度：0~40°C	x1
2		内存	标配 8GB DDR4 笔记本内存条	x1
3		硬盘	标配 64GB mSATA 硬盘	x1
4		显卡	支持 MXM 显卡	\
5		散热器	CPU 和 GPU 散热器	x1
6		适配器	标配电源适配器	x1
7		机箱	透明亚克力上下面板	x1
8		包材	产品包装	x1
9		合格证	产品合格证	x1
10		保修卡	产品合保修卡	X1
11		操作系统	银河麒麟桌面 V10 版试用版(预装在硬盘中)	\

注：①银河麒麟试用版系统对开发板使用无太大影响。②Ubuntu18.04和Debian10系统暂不支持，正在适配中。

1.2 硬件规格

- CPU：FT2000/4，集成4个FTC663内核，兼容64位ARMv8指令集，集成4MB二级缓存，4MB三级缓存，主频2.6GHz
- 内存：两个SODIMM插槽，单条最大支持16GB DDR4，支持DDR4-1600/DDR4-2666
- 3.5mm音频接口：2个，1个Mic输入、1个headphone输出
- USB 3.0：6个，包括前置2个，后置4个
- HDMI输出接口：1个
- DP输出接口：1个
- RJ45网口：2个，支持10M/100M/1000M模式，支持自适应网络
- MXM接口：1个
- 支持SoftAP：支持Wake on WLAN功能
- M.2接口：1个，支持2242规格的nvme硬盘，支持PCIe x4(Gen3)信号，支持m.2接口的wifi模块(PCIE协议)（RTL8821CE wifi麒麟系统自带驱动）；
- mSATA接口：1个，支持SATA3协议
- SATA接口：1个，支持SATA3协议
- 板载RTC芯片：1个
- DC接口：输入电压19V
- 飞腾教育开发板整体尺寸：199.4mm x 182mm

注：①硬盘为通用标准品，无特殊情况，市面上通用硬盘均可支持；②wifi 模块取决于驱动

1.3 飞腾教育开发板正面图

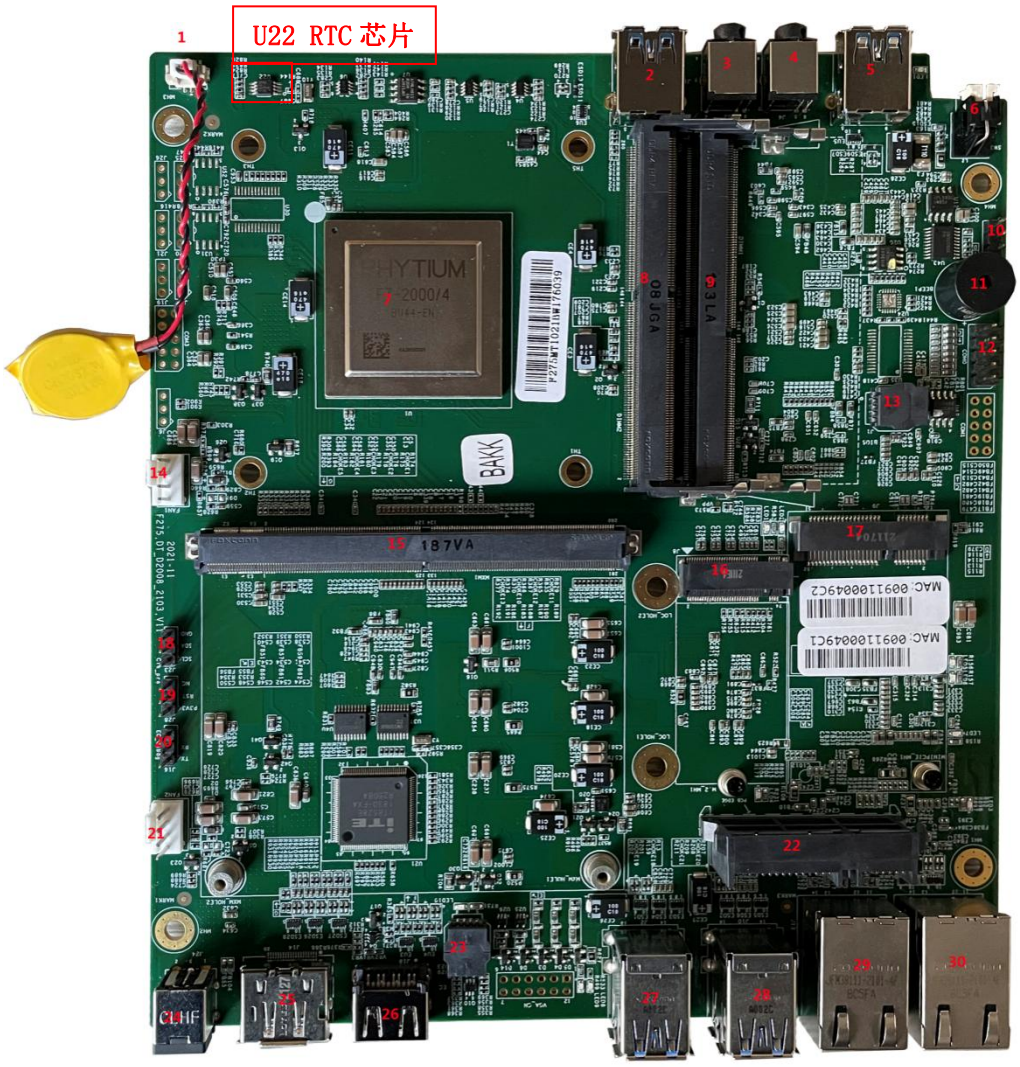


图1-1

功能接口说明：

序号	描述	序号	描述	序号	描述
1	电池接口	11	蜂鸣器	21	Fan 接口
2	前置 USB	12	RS232 串口	22	SATA 接口+供电
3	Audio	13	BIOS IC socket	23	EC IC Socket
4	Mic	14	Fan 接口	24	DC 电源接口
5	前置 USB	15	MXM 显卡插槽	25	DP 接口
6	电源开关	16	Nvme 接口	26	HDMI 接口
7	CPU	17	mSATA 接口	27	后置 USB 接口
8	DDR4 插槽	18	时钟 debug 口	28	后置 USB 接口
9	DDR4 插槽	19	时钟 debug 口	29	网口
10	CPU Debug 串口（TTL）	20	EC_debug 串口（TTL）	30	网口

1.4 飞腾教育开发板PCB规格

下图1-3所示为飞腾教育开发板PCB规格图，整体尺寸为195mm x 170mm

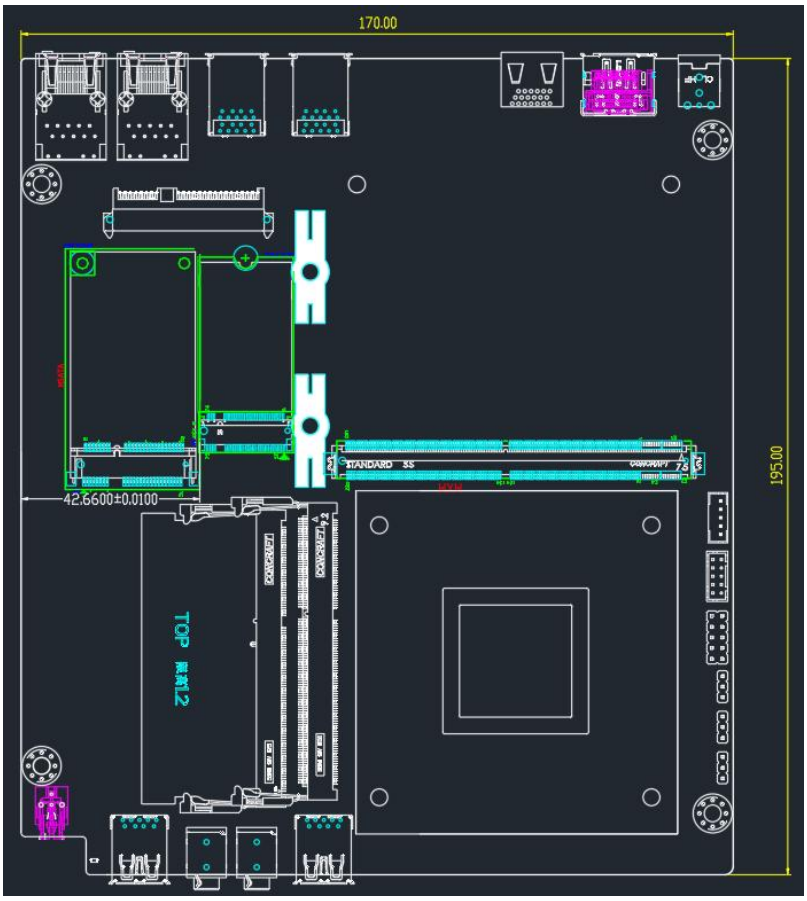


图1-3

1.5 系统组件

飞腾教育开发板支持以下系统组件：

- 操作系统：银河麒麟(kylin-Phytium-FT2000-4 4.4.131-20200710.kylin.desktop-generic)
- Linux内核：4.4.131-20210727.kylin.desktop-generic
- OpenCV：3.4.14
- 编译工具链gcc/g++：5.4.0
- python：2.7/3.5

二、 配件安装

2.1 安装MXM显卡

15为MXM接口，该接口主要用来连接MXM显卡，飞腾教育开发板已经适配多种型号的MXM显卡，开发板支持AMD R5 230显卡，也可以支持AMD HD8570显卡。

以下是MXM显卡安装方法（开发板发货时，散热片和散热风扇已配套安装好，如果拆卸过MXM显卡，需按此步骤完成安装）：

1） 断开电源后将开发板上面的亚克力外壳去掉

2） 将显卡插入MXM接口插槽，注意需要让显卡稍稍翘起，然后轻轻用力将其推入，如下图2-1所示

3） 使用配发的螺丝从背面向上锁住固定住显卡，左右螺丝都需要安装

4） 安装散热风扇，同时把风扇电源线连接到开发板FAN1(14)接口，防止显卡过热烧坏！！！

5） 将刚刚卸掉的亚克力外壳重新安装到开发板上

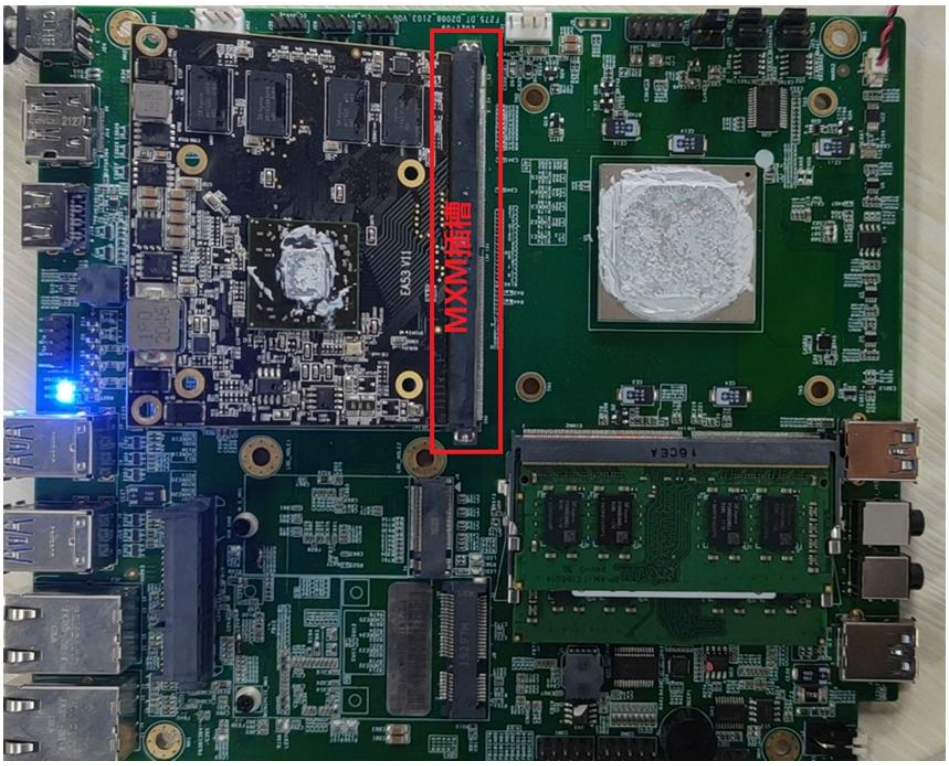


图2-1

2.2 安装硬盘

开发板有丰富的存储接口，实际使用时，可以任选其一。包括SATA接口，M.2接口，以及mSATA，分别为22、16、17；其中 M.2支持2242标准的NVMe盘，支持PCIE x4协议；mSATA接口支持SATA协议的mSATA盘；SATA接口支持2.5寸的标准SSD盘；

以下是mSATA/M.2安装方法：

1） 断开电源后将mSATA或者NVMe盘插入mSATA或者M.2接口插槽，稍微用力将其推入

2） 使用配发的螺丝固定住SSD，确保SSD不会脱落

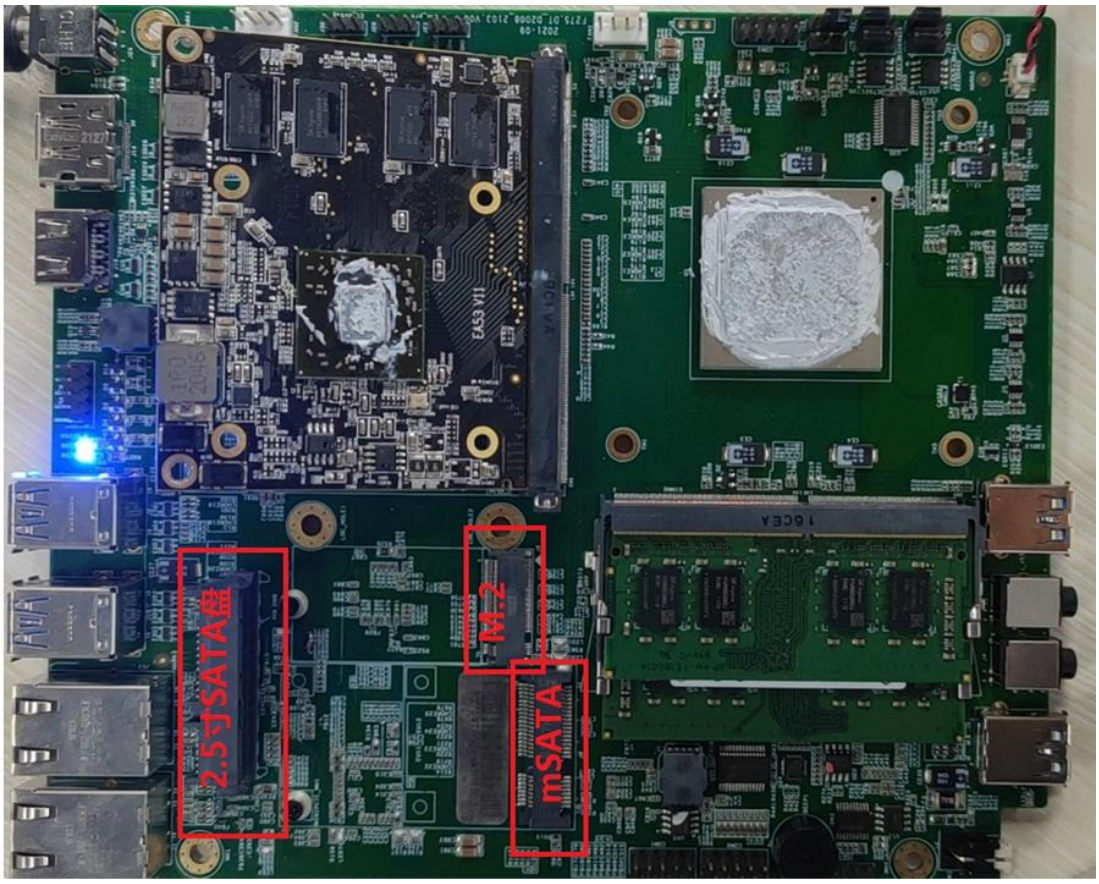


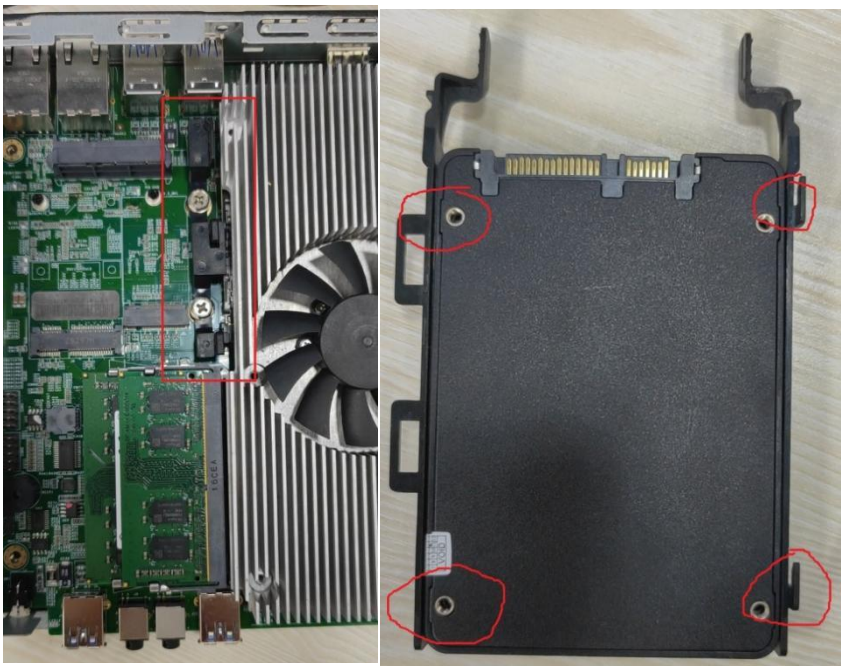
图2-2

2.5寸硬盘安装：

1） 先安装硬盘支架，如图所示，注意方向如图所示（本开发板默认采用mSATA硬盘，硬盘支架默认不发货）

2） 硬盘有特定支架，如图所示，注意安装

3） 然后如图所示将硬盘平放在托盘上进行安装，然后向上推动即可，安装OK后，会听到”咔”的一声



三、 启动飞腾教育开发板

3.1 开发板串口连接说明

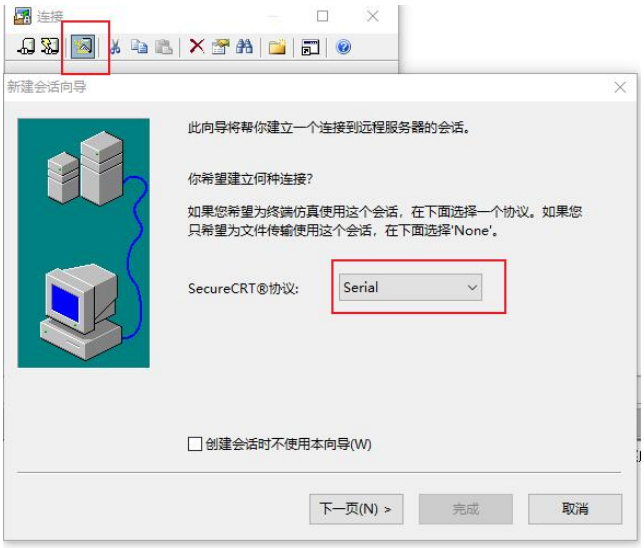
1） 准备笔记本、开发板、19V电源适配器（随机器配置）、USB转TTL调试串口线

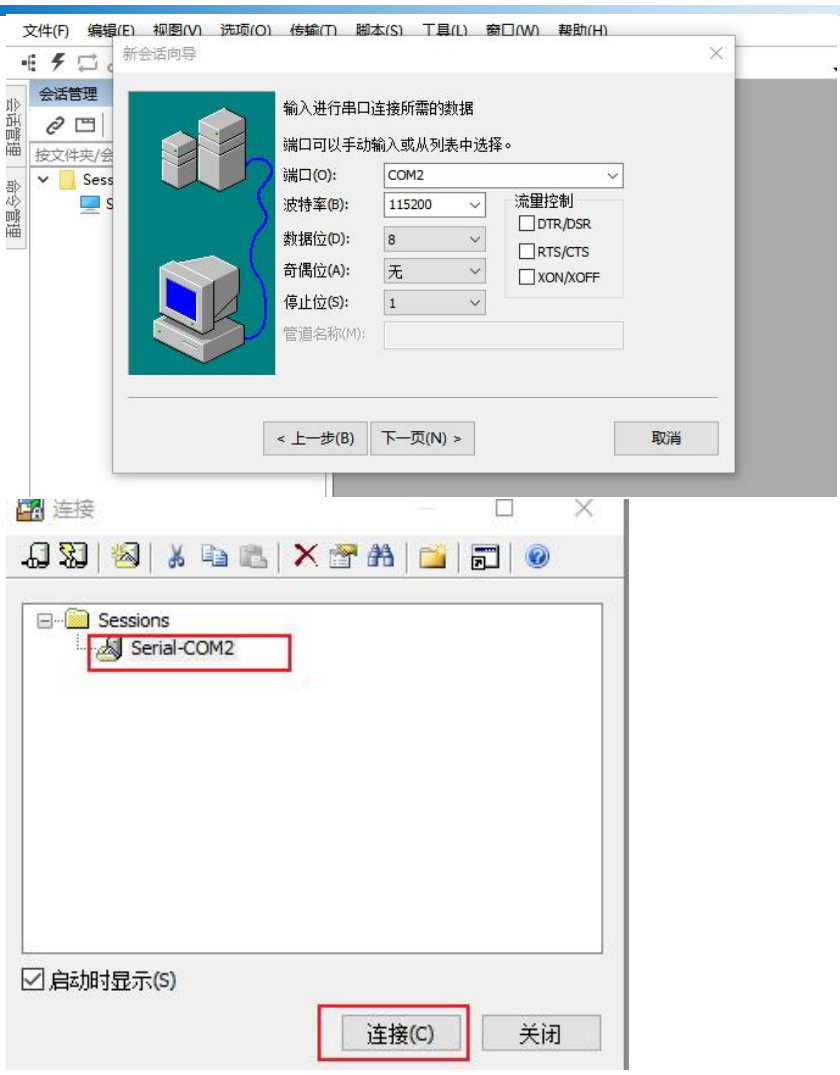
如图是CPU 3针Debug接口，支持TTL电平，波特率115200；连接顺序如图所示，如果没有输出，可以交换Tx和Rx线序。

2） 然后将USB转TTL调试串口线USB端连接到笔记本USB接口：

3） 笔记本安装 SecureCRT 工具，在电脑管理—>设备管理器—>通用串行总线控制器查看COM

4） 打开SecureCRT ，选择新建会话向导—协议（Serial）—>端口（com?）—>波特率（115200），然后下一步，创建完成后连接即可。




文件(F) 编辑(E) 视图(V) 选项(O) 传输(T) 脚本(S) 工具(L) 窗口(W) 帮助(H)
会话管理
新会话向导
输入进行串口连接所需的数据
端口可以手动输入或从列表中选择。
端口(O): COM2
波特率(B): 115200
数据位(D): 8
奇偶位(A): 无
停止位(S): 1
管道名称(M):
流量控制
DTR/DSR
RTS/CTS
XON/XOFF
< 上一步(B)
下一页(N) >
取消
连接
Sessions
Serial-COM2
启动时显示(S)
连接(C)
关闭

3.2 使用SSH登录开发板

我们不仅可以使用串口登录开发板，还可以使用SSH登录开发板。

如遇ssh各项配置正确，但无法连接，需要打补丁包

下载链接：https://pan.baidu.com/s/1GLNPPB5MVGwRvOy3-xHUYw

提取码：jbhd

安装步骤：dpkg -i ksc-defender_1.1.0-6kord_arm64.deb

安装后重启：reboot

以下是使用SSH登录开发板的步骤，其中用户名为root，密码Test@123。

1）Windos （cmd）输入如下命令：

```
C:\Users\bxxx> ssh root@192.168.0.122
```

2）再次输入密码，完成登录

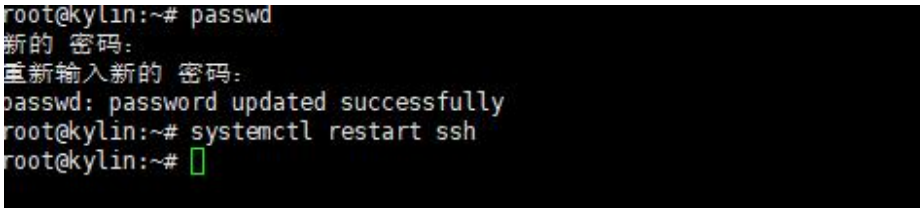
下图详细登录信息

注意：请根据开发板的实际网络地址进行登录。

报错分析：用户名、密码、网络地址都输入正确的情况下，报如下错误信息，该如何解决？

解决方案如下所示：

1） 修改root密码，重启ssh



3.3 开发板关机流程

开发板关机流程如下所示：

1） 终端输入poweroff命令，然后等待5秒钟直到所有程序关闭

```
root@kylin:~# poweroff
```

2) 拔掉19V电源适配器，使得开发板断电

四、 外置接口

4.1 前面板功能接口

下图4-1所示为飞腾教育开发板前面板功能接口。

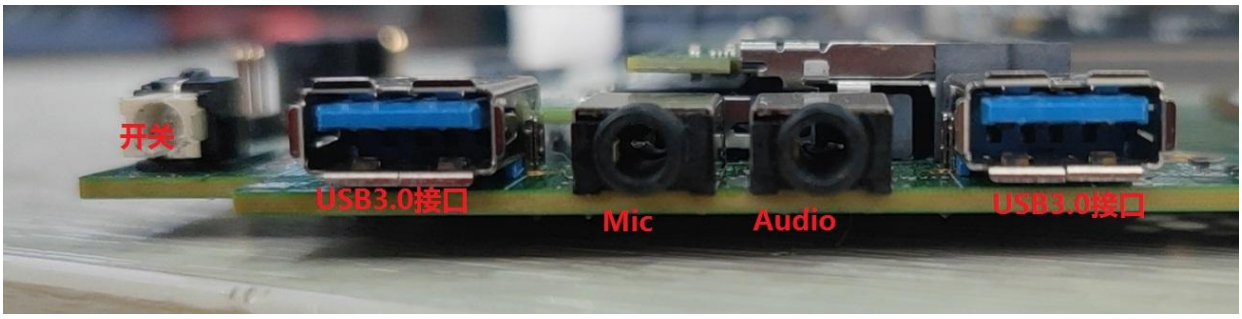


图4-1

功能说明：

1） 开关：在关机状态，短按开机，并且在上电完成后，会有滴一声，表明主板上电完成；在启动中，短按重启；主板会重新开始上电；如果长按4s，蜂鸣器会有一声长响，表明主板被强制断电关机；

2） USB接口：支持USB3.0

3） Mic：支持3.5mm录音输入

4） Audio：支持3.5mmHeadphone输出

4.2 后面板功能接口

下图4-2所示为飞腾教育开发板后置功能接口

图4-2

功能说明：

1） DC，主板电源接口，输入19V

2） DP接口，显示接口，DP插座

3） HDMI接口，显示接口，HDMI插座

4） USB3.0，双层USB3.0接口

5） 网口：RJ45接口，千兆电口

4.3 RJ45网口

参见《1.3 飞腾教育开发板正面图》中的功能接口说明，29、30为开发板RJ45网口，支持

10M/100M/1000M模式，其中30为enaphyt4i0开发板网口测试：

1） 打开enaphyt4i0

root@kylin:~# ifconfig enaphyt4i0 up

2） 查看enaphyt4i0 IP地址

root@kylin:~# ifconfig enaphyt4i0

3） Ping百度官网

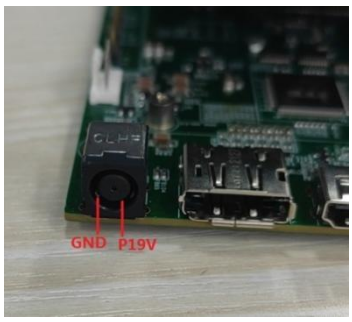
root@kylin:~# ping www.baidu.com

4） 关闭enaphyt4i0

root@kylin:~# ifconfig enaphyt4i0 down

4.4 电源接口

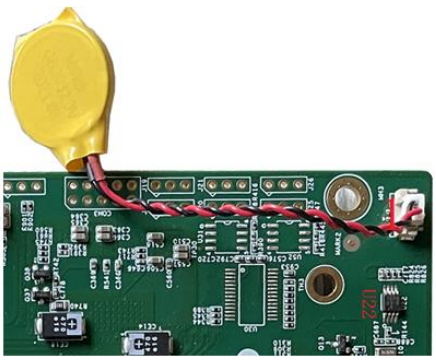
参见《1.3 飞腾教育开发板正面图》中的功能接口说明，24为开发板电源接口，仅支持19V电源输入，管脚定义如下图所示：



注意：开发板连接全功率负载测试时，请使用19V/4.7A及以上电源适配器，全功率负载测试包括：LTP、Spec、MEMtest等。

4.5 RTC实时时钟

参见《1.3 飞腾教育开发板正面图》中的功能接口说明，U22为开发板RTC芯片，1为钮扣电池座，在开发板掉电时为RTC芯片供电。



实时时钟测试：

1） 查看系统时间

2） root@kylin:~# date

2022年 02月 18日 星期五 16:16:33 CST

3） 查看RTC时间

```
root@kylin:~# hwclock -f /dev/rtc0
2022年02月18日 星期五 16时23分22秒  .445021 seconds
```

4） 设置系统时间

```
root@kylin:~# date -s "2021-03-10 11:55:00"
Wed 10 Mar 2021 11:55:00 AM CST
root@kylin:~# date
Wed 10 Mar 2021 11:55:10 AM CST
```

5） 同步系统时间到RTC芯片

```
root@kylin:~# hwclock -w -f /dev/rtc0
root@kylin:~# hwclock -f /dev/rtc0
Wed 10 Mar 2021 11:56:00 AM CST
```

4.6 存储接口

参见《1.3 飞腾教育开发板正面图》中的功能接口说明，开发板正面三个存储接口，分别为16、17和22，M.2接口内部集成PCIe x4(Gen3)信号。

参考本文2.2章，将存储设备安装到对应接口

以下是SSD挂载测试：

1） SSD分区查询

```
root@kylin:~# fdisk -l /dev/sdb1
```

2） 挂载SSD分区

```
root@kylin:~# mount /dev/sdb1 /mnt
```

3） 拷贝数据

从开发板拷贝Linux内核日志到SSD

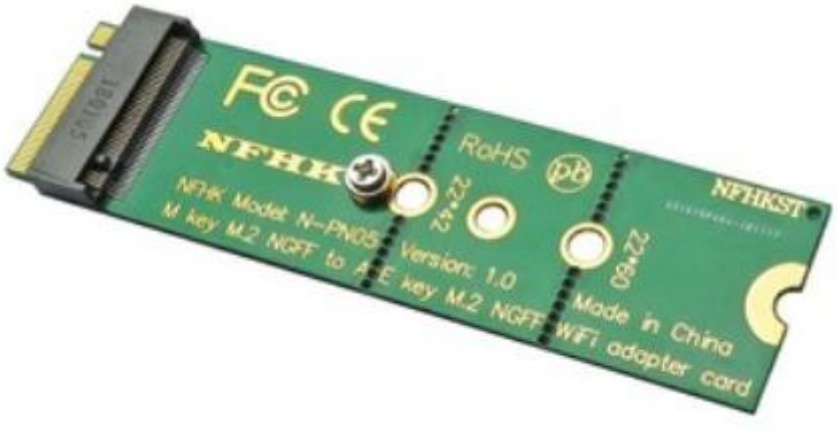
```
root@kylin:~# dmesg > kernel_log
root@kylin:~# cp kernel_log /mnt/
root@kylin:~# sync
```

4） 卸载分区

```
root@kylin:~# umount /mnt
```

4.7 WIFI模块

可采用RTL8821CE M.2 wifi,银河麒麟系统下自带驱动，更新内核即可；因主板为M.2 Key M插槽，M.2 wifi一般为Key A或A+E，需要将M.2 wifi模块插在M Key转NGFF转接卡上使用。



1） 升级内核

下载上传内核包 5.4.18-51内核-arm.tar.gz

链接：https://pan.baidu.com/s/1jsAS81eG2tZb8NOzGw3jzA

提取码：kbb1

tar -xvf 5.4.18-51内核-arm.tar.gz

2） 进到文件夹 cd arm

3） 安装内核包

dpkg -i *.deb

4） 重启 reboot

5） 进入系统后，连接wifi，有两种方式连接

a） 第一种方式：图形化连接

b） 第二种方式：命令行连接

```
nmcli device wifi connect ssid password wifi_passwd
```

6） 连接后ifconfig可以查看IP地址

```
enaphyt4i0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.159  netmask 255.255.255.0  broadcast 192.168.2.255
        inet6 fe80::bccf:a0cb:2509:7b99  prefixlen 64  scopeid 0x20<link>
        ether 00:07:3e:90:02:02  txqueuelen 1000  (以太网)
        RX packets 15119  bytes 1850769 (1.8 MB)
        RX errors 0  dropped 70  overruns 0  frame 0
        TX packets 5568  bytes 560999 (560.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 9  base 0x4000

enaphyt4i1: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 00:07:3e:90:02:03  txqueuelen 1000  (以太网)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 10  base 0x4000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (本地环回)
        RX packets 168  bytes 35787 (35.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 168  bytes 35787 (35.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp5s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.43.252  netmask 255.255.255.0  broadcast 192.168.43.255
        inet6 fe80::1364:baia:c840:cbc1  prefixlen 64  scopeid 0x20<link>
        inet6 2409:8954:38c8:664c:3e2a:2f19:68ac:7cc1  prefixlen 64  scopeid 0x0<global>
        inet6 2409:8954:38c8:664c:6cb6:e440:21f3:5b84  prefixlen 64  scopeid 0x0<global>
        ether ec:2e:98:56:be:b9  txqueuelen 1000  (以太网)
        RX packets 154  bytes 26810 (26.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 334  bytes 59086 (59.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@kylin-D2000:/home/kylin#
```

五、 BIOS在线升级和系统安装

5.1 BIOS升级固件

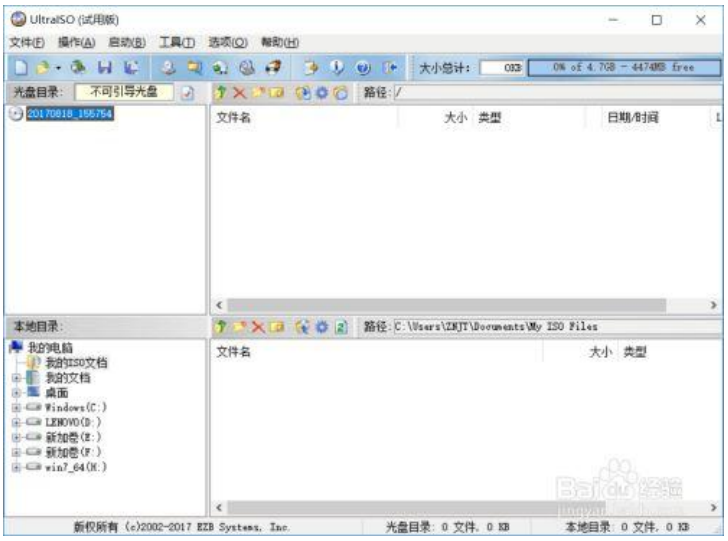
1） 在线升级方法：接入带固件的 U 盘，开机进 BIOS 引导界面按 F2 进入快速引导界面选择 Update BIOS 进入自动检索升级。

2） 拆卸主板 BIOS 芯片槽使用烧录工具进行更新，如下图，13 为 BIOS 芯片座子，BIOS 芯片在座子里。

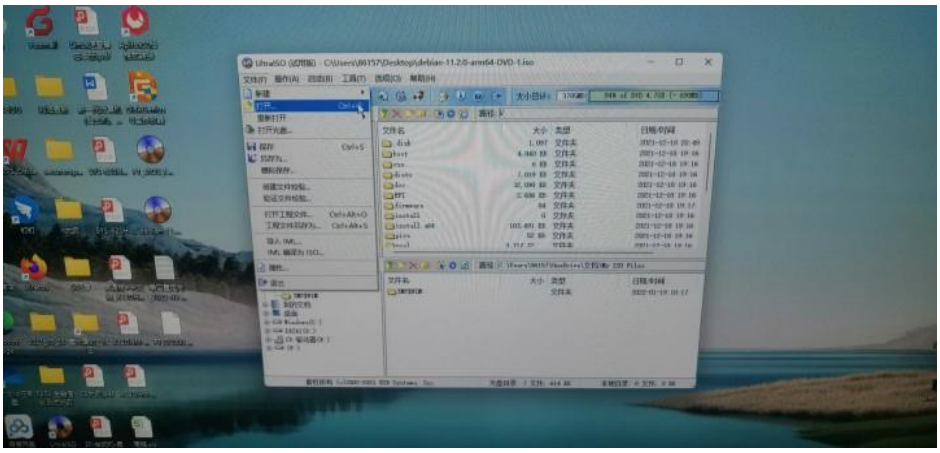
BIOS 如果没有损坏或厂家没有升级要求，不建议升级尝试。

5.2 启动盘刻录

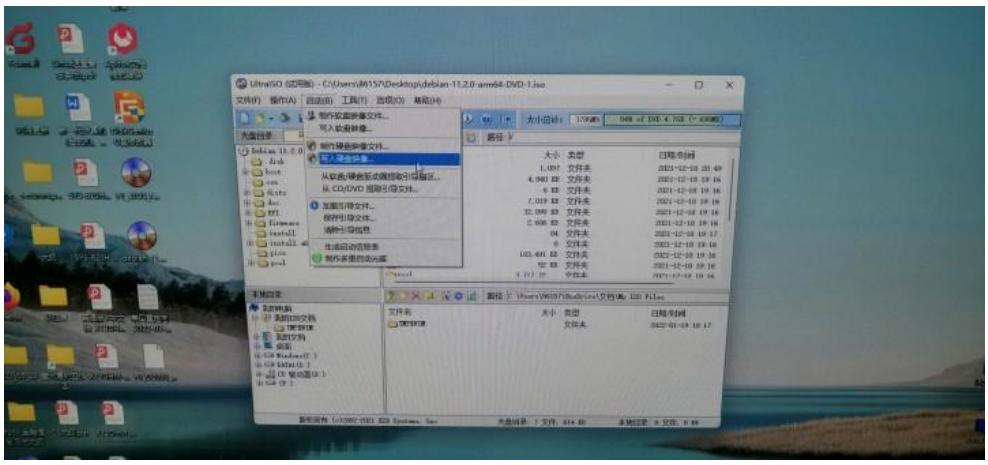
1） 打开 UltraISO 软件



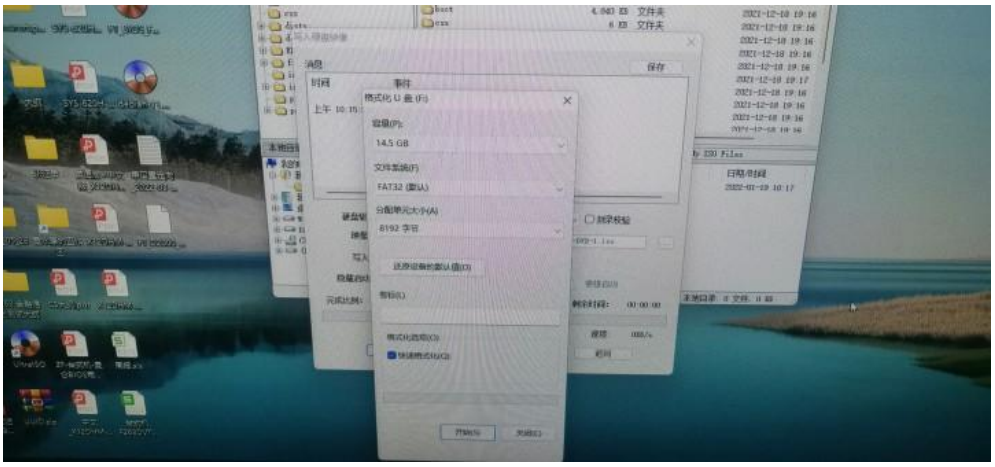
2） 菜单栏“文件”——“打开”---找到需要刻录的系统文件



3） 点击“启动”——“写入硬盘映像”



4） 点击格式化--将 U 盘格式化（注意文件系统格式和容量）



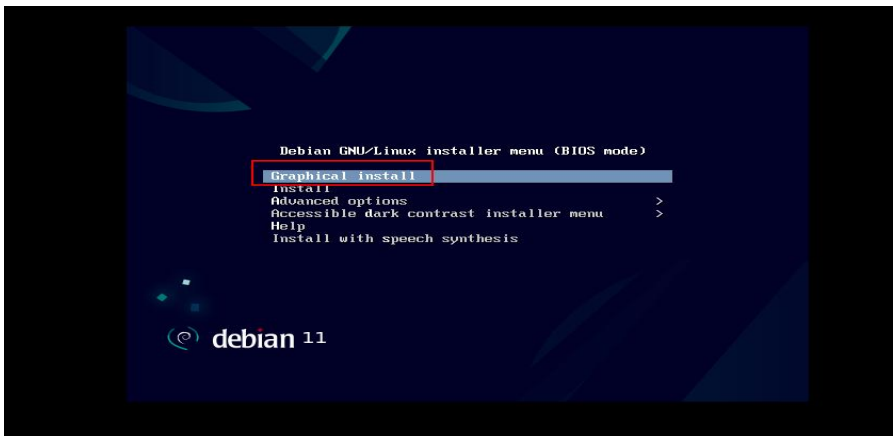
5） 结束后，点击“写入”，进行刻盘操作，等刻录完成，开机启动选择 U 盘启动即可。

5.3 debian系统安装

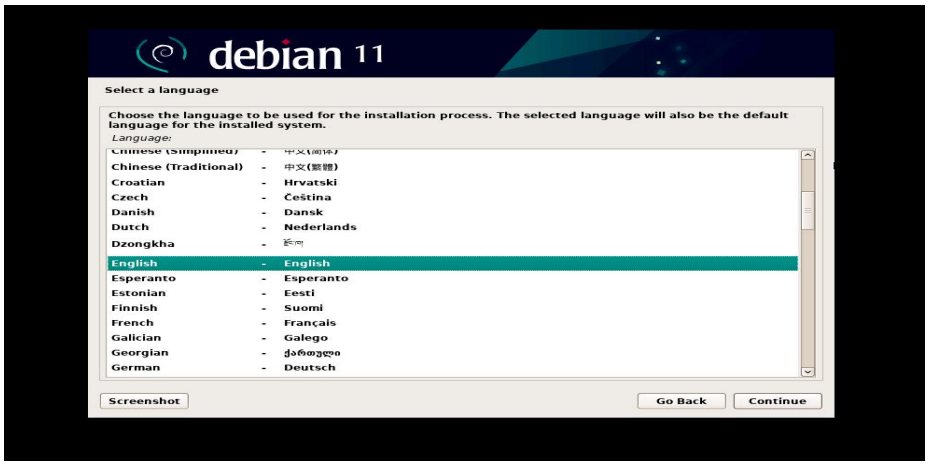
1） 接入刻录好的 U 盘，开机进入 BIOS 引导界面，按 F2 进入快速启动界面：

2） 选择 U 盘启动，进入 grub 菜单引导安装

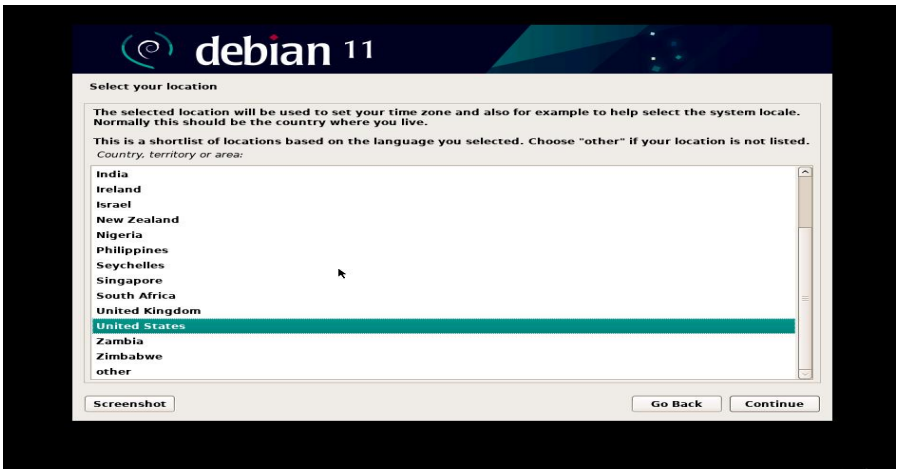
3） 进入安装界面，默认选择第一个图形化安装界面，回车



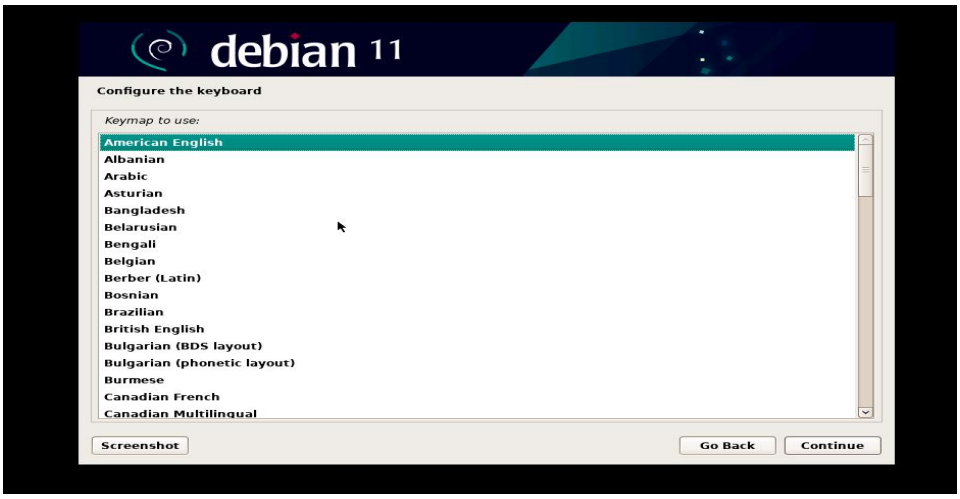
4） 选择语言：我这里选择 English 语言，然后点击 Continue



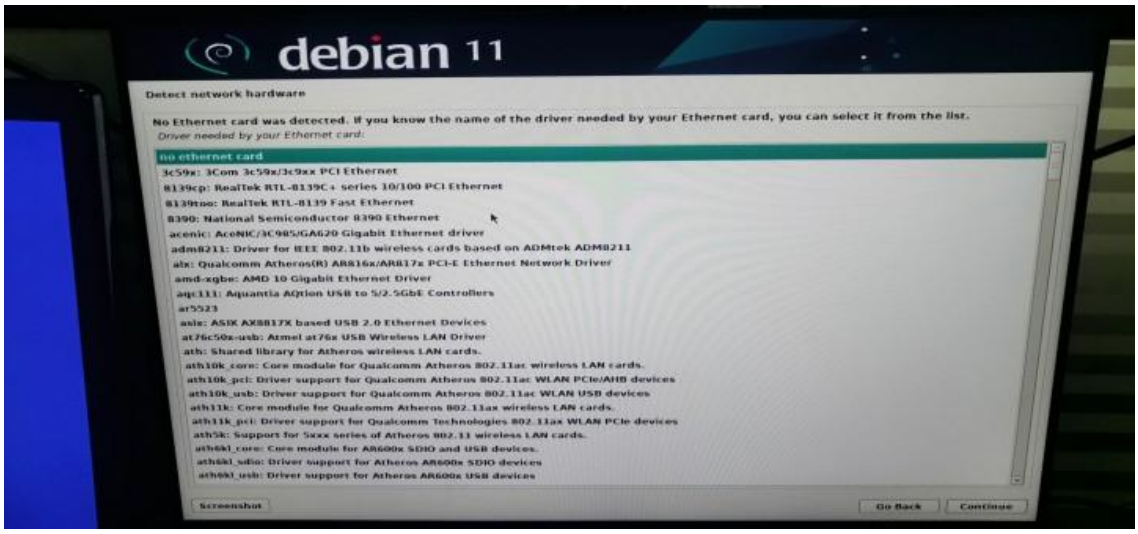
5） 选择所在位置：



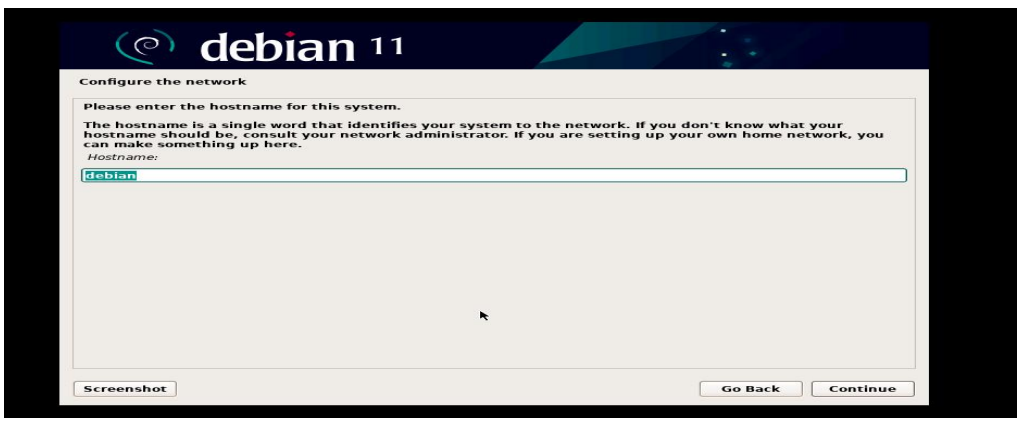
6） 键盘布局，默认 American English,然后点击 Continue



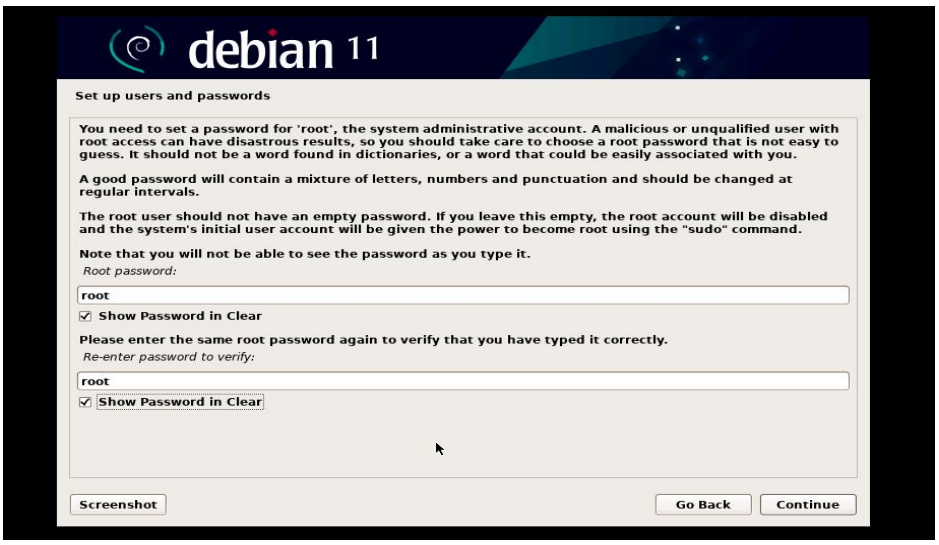
7） 选择配置网卡，没有就选择 no ethernet card 然后点击 Continue



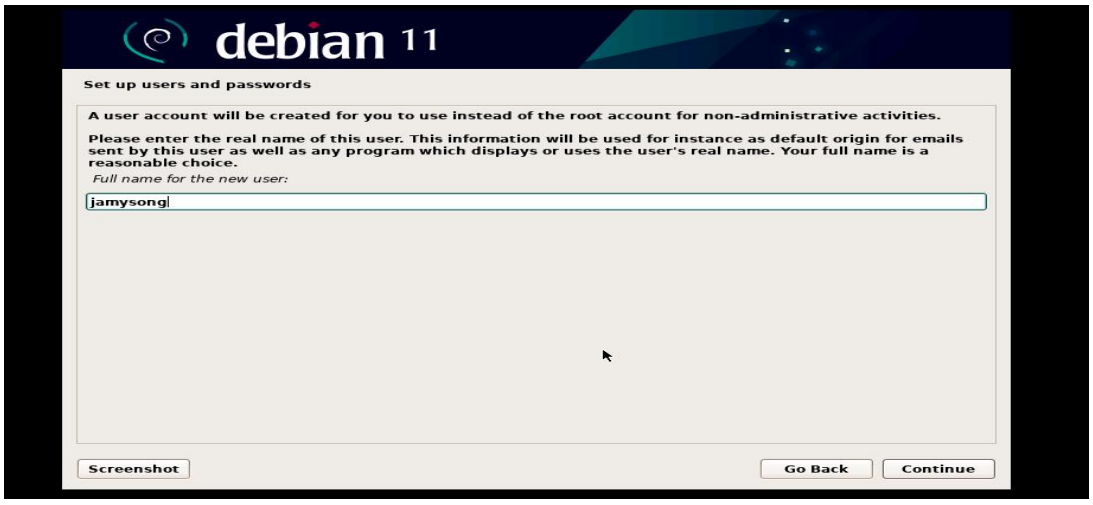
8） 设置主机名：默认主机名(hostname)为 debian，点击 Continue

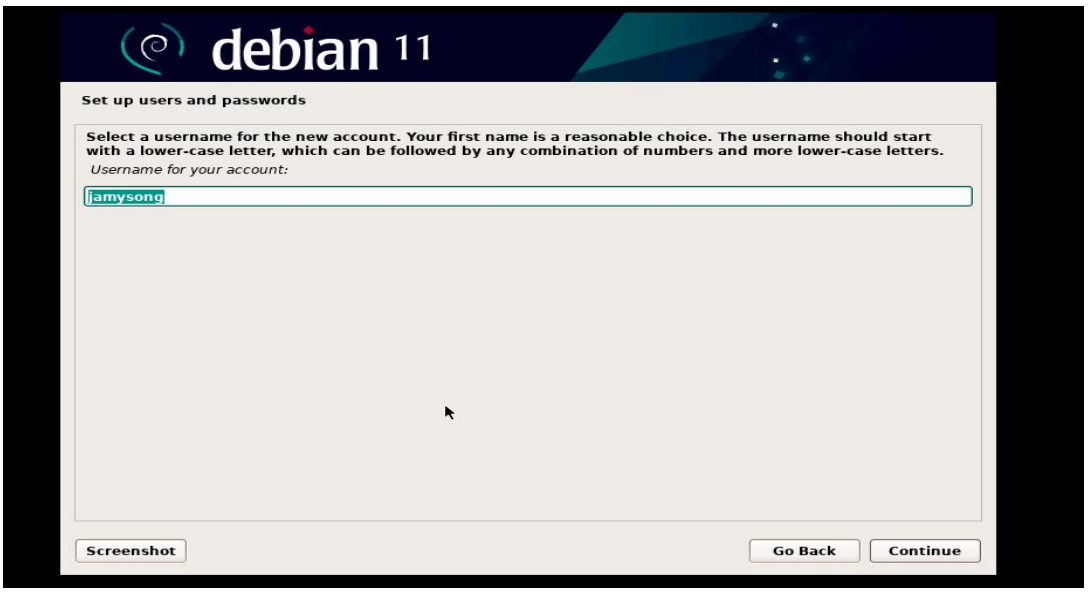


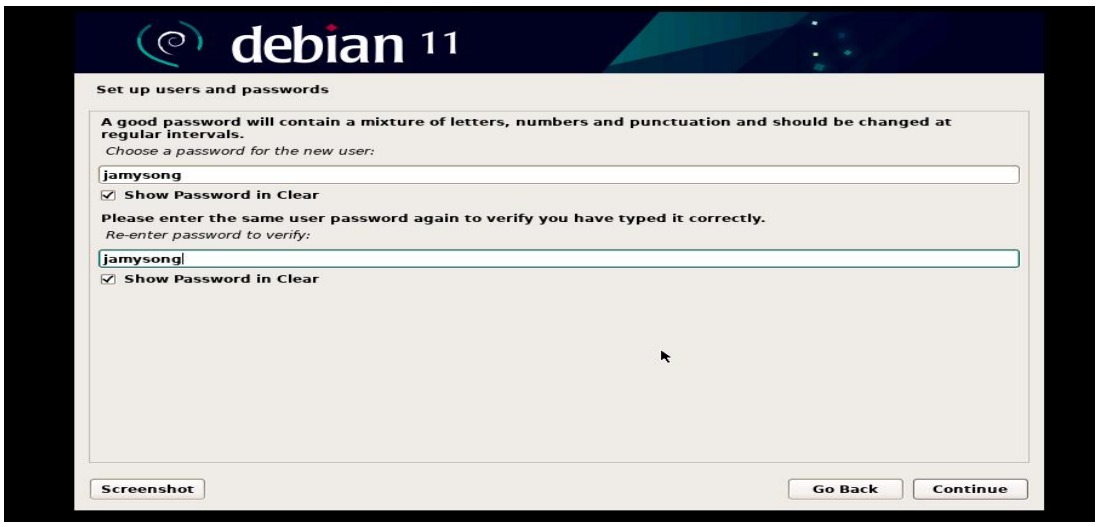
9） 设置 root 账号及密码



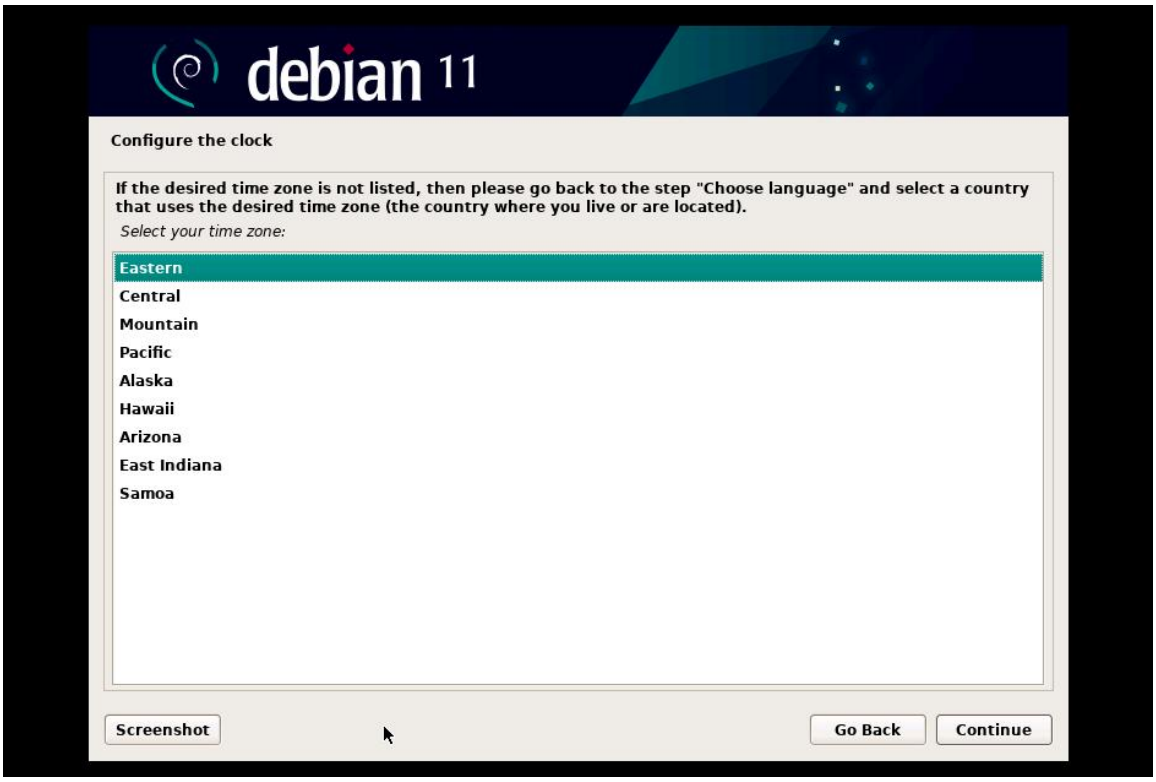
10）设置用户名及密码


debian 11
Set up users and passwords
A user account will be created for you to use instead of the root account for non-administrative activities.
Please enter the real name of this user. This information will be used for instance as default origin for emails sent by this user as well as any program which displays or uses the user's real name. Your full name is a reasonable choice.
Full name for the new user:
jamysong
Screenshot
Go Back
Continue


debian 11
Set up users and passwords
Select a username for the new account. Your first name is a reasonable choice. The username should start with a lower-case letter, which can be followed by any combination of numbers and more lower-case letters.
Username for your account:
jamysong
Screenshot
Go Back
Continue

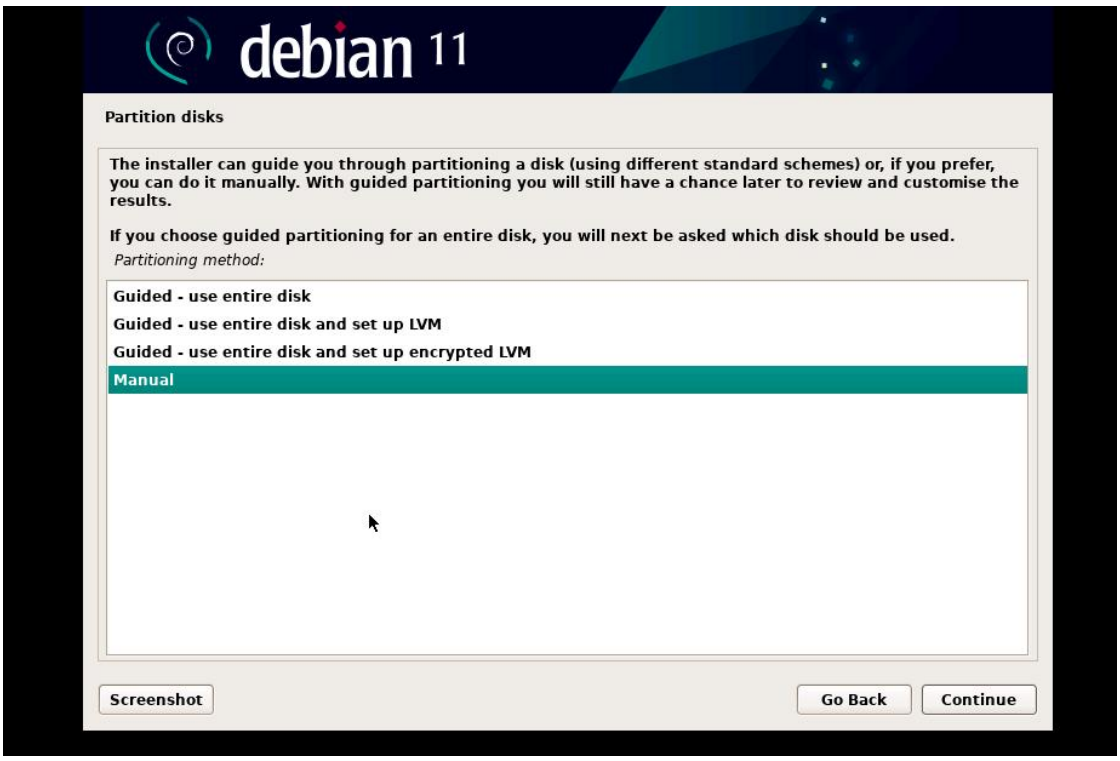

debian 11
Set up users and passwords
A good password will contain a mixture of letters, numbers and punctuation and should be changed at regular intervals.
Choose a password for the new user:
jamysong
Show Password in Clear
Please enter the same user password again to verify you have typed it correctly.
Re-enter password to verify:
jamysong
Show Password in Clear
Screenshot
Go Back
Continue

11）设置时区



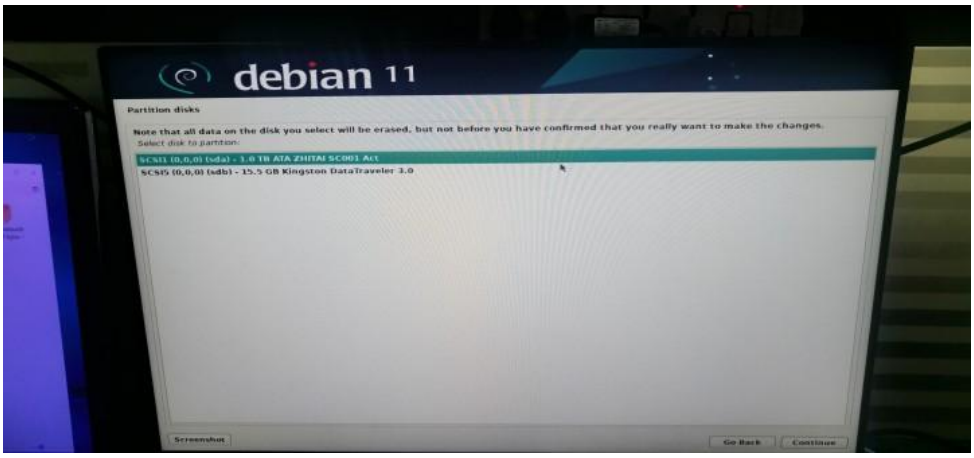
12) 磁盘分区

选择 Manual,进行手动磁盘分区模式，如果你不是太熟悉手动磁盘分区，可以选择第一个 Guided-use entire disk，这样就可以自动分区了。

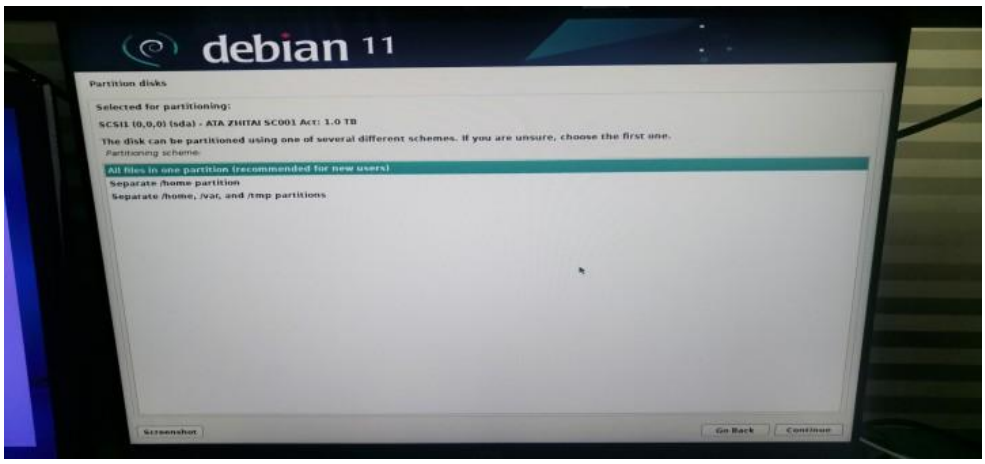


13）选择好自动分区之后，开始安装

①选择安装磁盘位置，然后点击 Continue

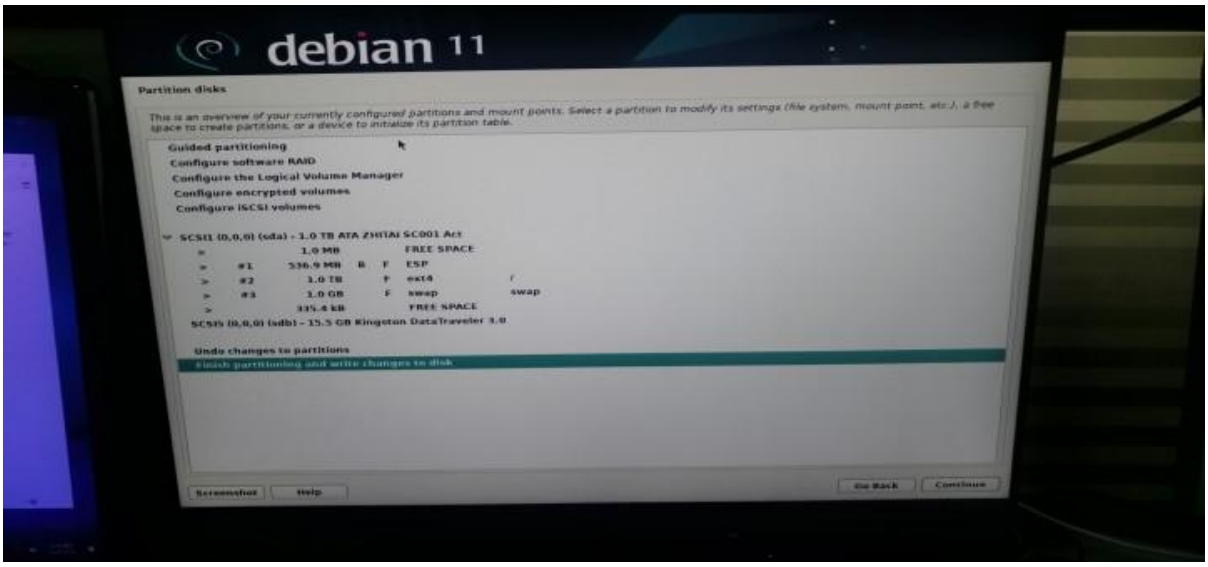


②提示新用户安装推荐方法，然后点击 Continue

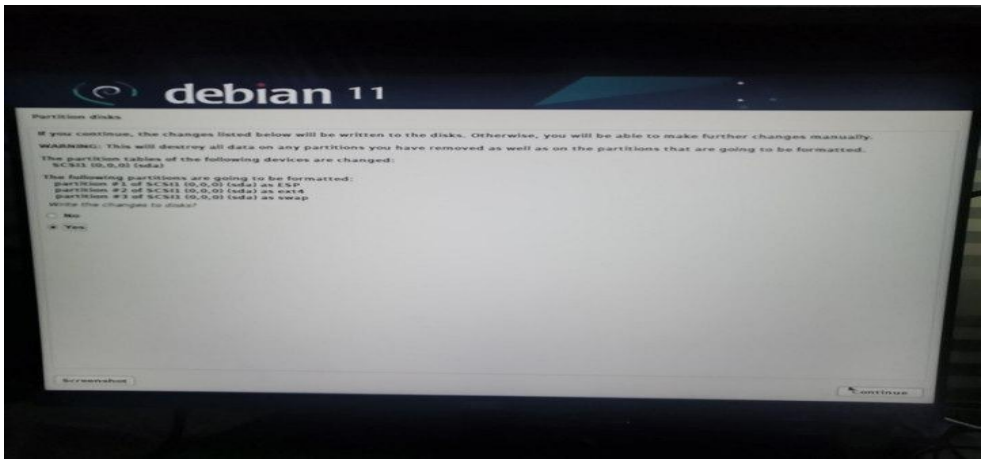


③执行分区方案，完成分区

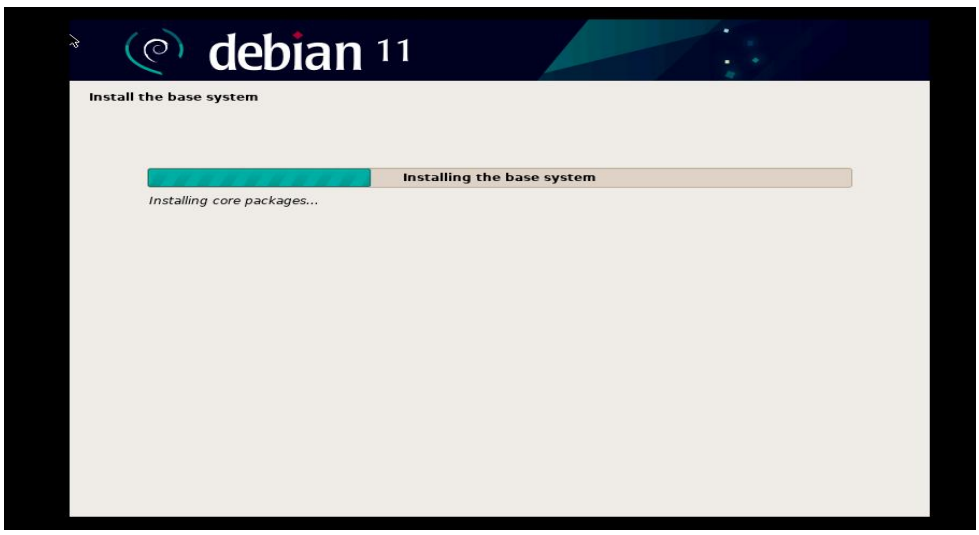
这里主要是执行前面步骤中设定的分区方案，如果想重新进行分区，在这一步点击 Continue 之前还来得及，如果没有问题，直接点击 Continue。



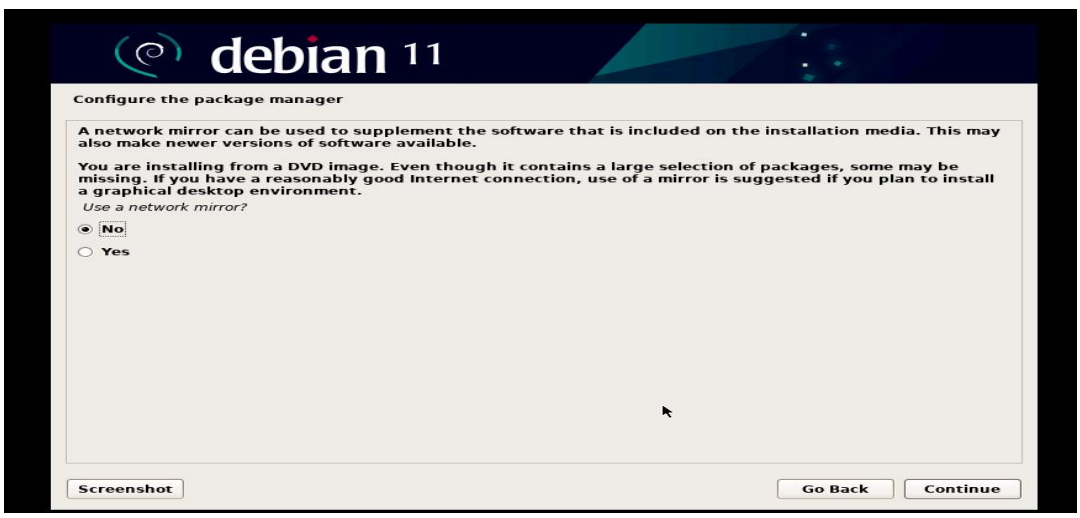
④确认执行分区方案，进行写入磁盘操作，点击 Continue



⑤如图所示正在执行分区操作并安装系统，耐心等待。。。。。。



⑥安装系统镜像：询问是否使用网络镜像，默认选择 no,然后点击 Continue

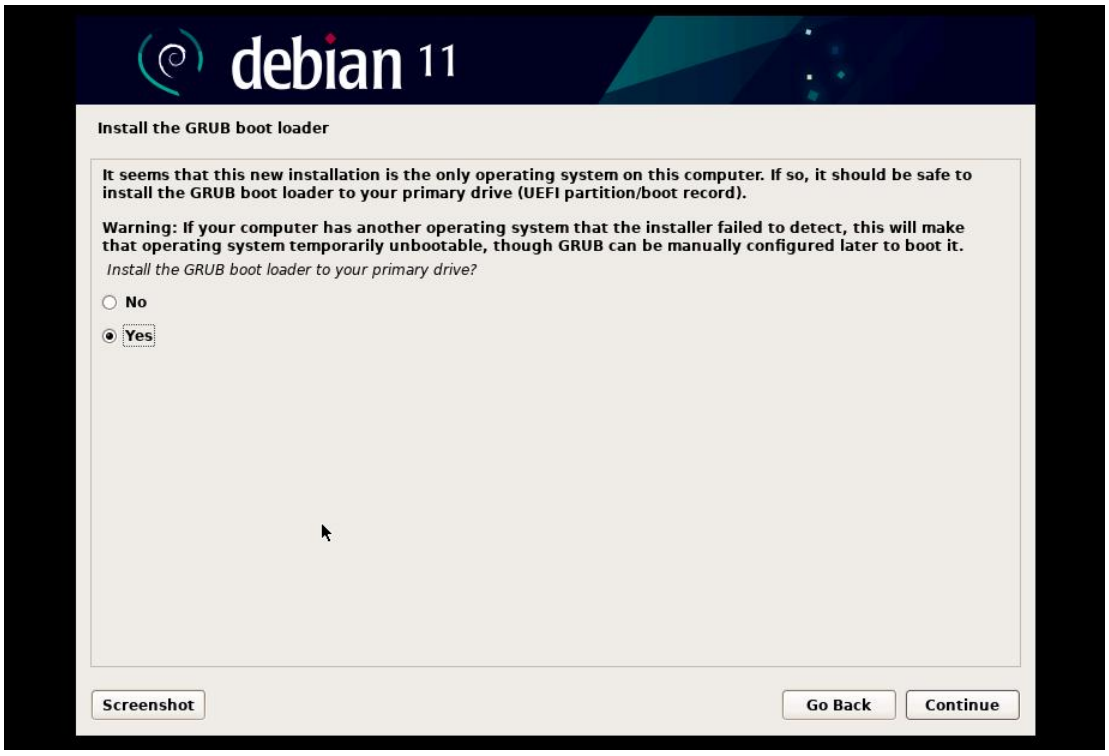


⑦软件包使用情况调查：这里默认选择 no,不用管这个，点击 Continue 继续往下走

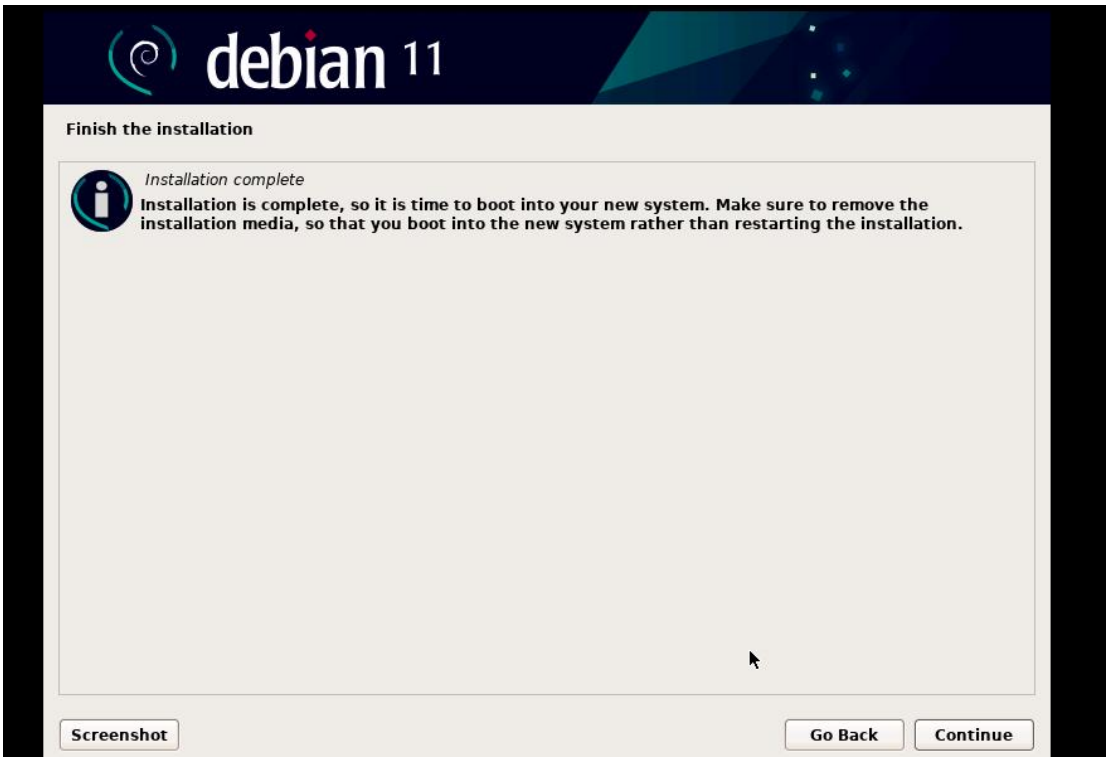
⑧安装环境

这里选择安装环境，在勾选如图四个选项，勾选上 SSH server，这个是为了后面可以进行用户 ssh 客户端软件进行连接，主要是为了方便命令操作和上传下载文件，然后点击 Continue 进入

⑨安装引导程序：这里是安装系统引导程序，默认 yes 即可，点击 Continue



⑩系统安装完成：走到这一步说明系统已经安装成功，点击 Continue 重启进入系统



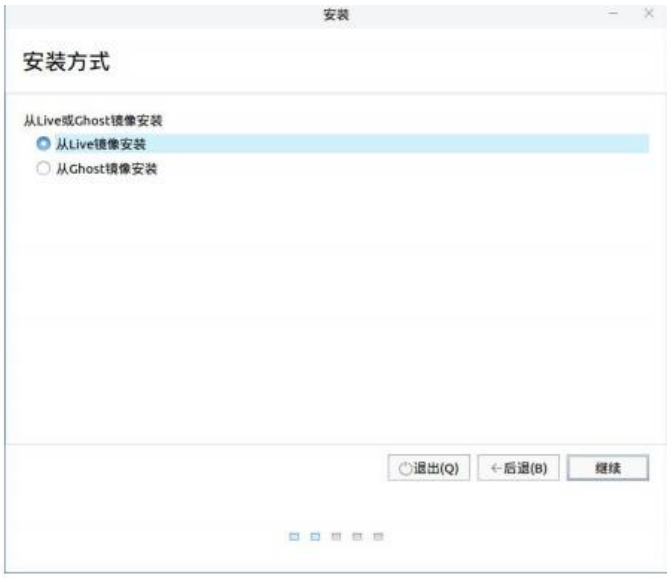
14）系统登录：这里直接点击用户名即可，如果你需要切换用户，点击 Not listed?

5.4 银河麒麟系统安装

1） 双击图标“安装 Kylin-Desktop-V10”，开始安装引导，此处可选择语言。

2） 点击“继续”，勾选同意许可协议，再点击“继续”，进入安装方式选择。

3） 选择“从 Live 镜像安装”，点击“继续”，进入安装类型界面。“从 Ghost 镜像安装”



4） 以下是对 4 个选项的详细介绍： ① “创建备份还原分区”：挂载点为“/backup”。勾选后，选择“快速安 装 Kylin”时，分区大小默认与根分区相同。只有创建了该分区，备份还原功能 才可以使用。备份还原对用户恢复数据或系统非常有帮助，建议创建。 ②“创建数据盘”：挂载点为“/data”。勾选后，选择“快速安装 Kylin”。

5） 银河麒麟桌面操作系统（飞腾版）V10 用户手册 第 88 页/共 93 页 时，分区大小为整个磁盘除掉其他分区外的所有空间。/data 类似于 Windows 系 统除 C 盘外的其他盘符，建议创建。 注意，①、②两个选项的勾选，是针对快速安装的设置。 ③ “高级安装”：用户自行根据实际需求，进行分区创建和分区大小分配。 详细说明见第 5 部分。 ④ “快速安装 Kylin”：全盘安装，该选项将会格式化整个硬盘，并进行自动分区。

6） 选择“快速安装 Kylin”选项，同时勾选“创建备份还原分区”和“创建数 据盘”，点击“现在安装”按钮，弹出格式化分区警告信息。

7） 点击“继续”（此时硬盘已经被格式化和重新分区），弹出创建用户信息窗口。

8） 信息正确填完后，“继续”按钮由灰变亮，点击“继续”按钮，此时会将系 统信息写入硬盘。

9） 安装完成后，会弹出提示窗口。

10）点击“现在重启”按钮，系统会重新启动。重启过程中系统会自动弹出光驱 或提示拔出 U 盘。取回光驱或 U 盘后，等待系统进入登录界面，输入密码后即可 进入系统。

5.5 tigervnc 局域网远程桌面

步骤一：切换 root 用户

```
kylin@Kylin:~$ sudo -i
[sudo] kylin 的密码：
```

步骤二：将 tigervnc 安装包上传到操作系统并安装

```
root@Kylin:/home/kylin/tigervnc-20180315-arm64# ls
libfltk1.3_1.3.3-4kord_arm64.deb          tigervnc-standalone-server_1.7.0-2kord1k
libfltk-images1.3_1.3.3-4kord_arm64.deb   tigervnc-viewer_1.7.0-2kord1k3_arm64.deb
tigervnc-common_1.7.0-2kord1k3_arm64.deb
root@Kylin:/home/kylin/tigervnc-20180315-arm64# dpkg -i *.deb
正在选中未选择的软件包 libfltk1.3:arm64。
(正在读取数据库 ... 系统当前共安装有 181776 个文件和目录。)
正准备解包 libfltk1.3_1.3.3-4kord_arm64.deb  ...
正在解包 libfltk1.3:arm64 (1.3.3-4kord) ...
正在选中未选择的软件包 libfltk-images1.3:arm64。
正准备解包 libfltk-images1.3_1.3.3-4kord_arm64.deb  ...
```

或者

{

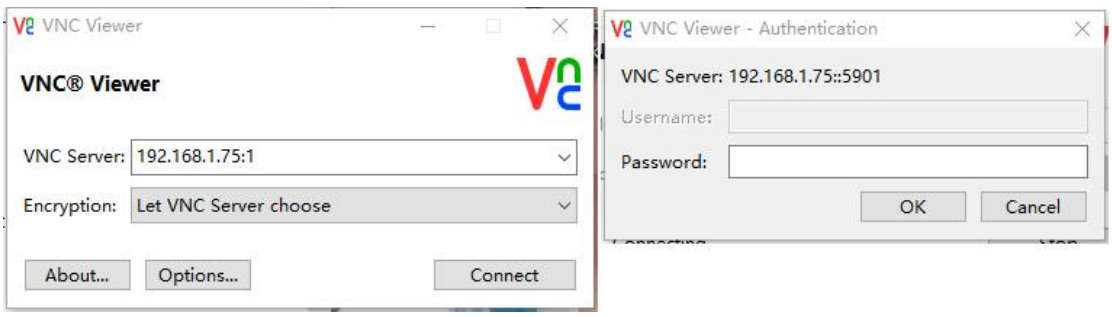
```
root@Kylin:/usr/share/lightdm/lightdm.conf.d# apt-get install tigervnc-standalone-server
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
下列软件包是自动安装的并且现在不需要了：
  cifs-utils localechooser-data user-setup
使用'apt autoremove'来卸载它(它们)。
将会同时安装下列软件：
  xauth
建议安装：
  xfonts-100dpi | xfonts-75dpi xfonts-scalable
推荐安装：
  tigervnc-common
下列【新】软件包将被安装：
  tigervnc-standalone-server xauth
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 0 个软件包未被升级。
需要下载 0 B/844 kB 的归档。
解压缩后会消耗 2,369 kB 的额外空间。
您希望继续执行吗？ [Y/n] Y
root@Kylin:/usr/share/lightdm/lightdm.conf.d# apt-get install tigervnc-common
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
下列软件包是自动安装的并且现在不需要了：
  cifs-utils localechooser-data user-setup
使用'apt autoremove'来卸载它(它们)。
下列【新】软件包将被安装：
  tigervnc-common
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 0 个软件包未被升级。
需要下载 0 B/61.5 kB 的归档。
解压缩后会消耗 188 kB 的额外空间。
【警告】：下列软件包不能通过认证！
  tigervnc-common
没有验证的情况下就安装这些软件包吗？ [y/N] y
```

}

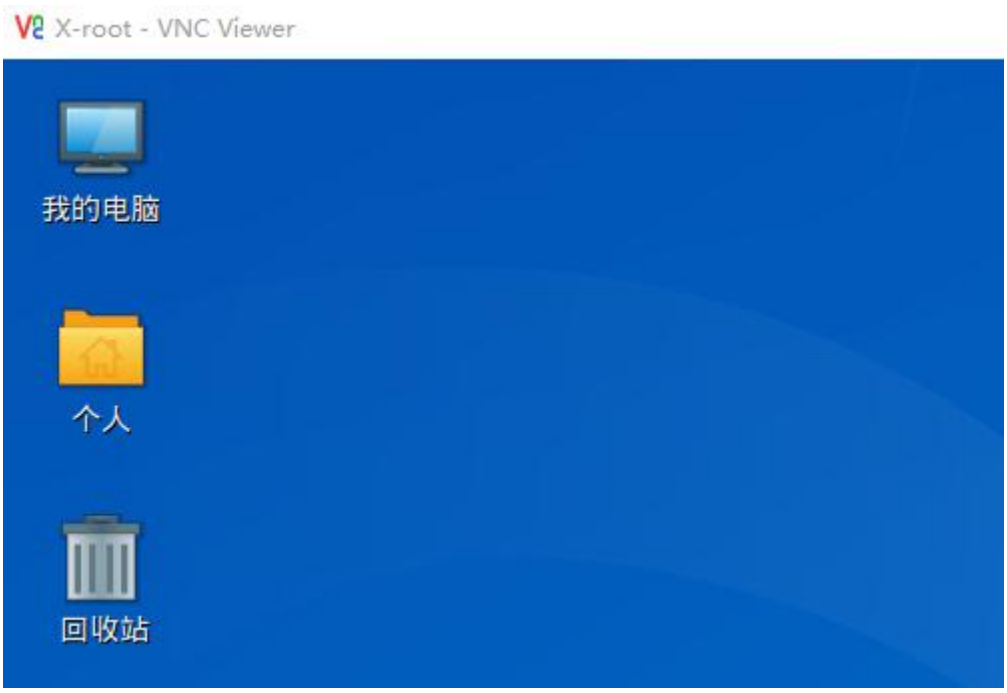
步骤三：启动远程桌面管理服务 vncserver，注意图中绿框中的端口号（：1）内容，客户端会用到

```
root@Kylin:/home/kylin/tigervnc-20180315-arm64# vncserver

You will require a password to access your desktops.

Password:
Verify:
Would you like to enter a view-only password (y/n)? y
Password:
Verify:
/usr/bin/xauth:  file /root/.Xauthority does not exist
Creating default startup script /root/.vnc/xstartup

New 'X-root' desktop at Kylin:1

Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/Kylin:1.log
```

步骤四：在客户端安装 VNCviewer 软件,在客户端填入“IP 地址：端口号”，确认后输入密码；



步骤五：连接后的状态，用户默认为 root 用户；



5. 6 Python 更新 3.7

1. wget 下载 python 安装包

wget https://www.python.org/ftp/python/3.7.3/Python-3.7.3.tgz

2. 安装
   1. 新建安装路径，然后解压安装包
      Mkdir -p /usr/local/python3
      tar -zxvf Python-3.7.2.tgz
   2. 进入解压后的路径编译，安装
      # cd Python-3.7.3
      #用./configure --prefix=/usr/local/python3 --enable-optimizations
      make && make install
      如果 make install 出现报错，需要安装 libffi-dev
      from _ctypes import Union, Structure, Array ModuleNotFoundError: No module named '_ctypes' Makef
      apt-get install libffi-dev
3. 编译安装成功后，添加 python3 软链接
   sudo ln -s /usr/local/bin/python3.7 /usr/bin/python3.7

```
root@kylin-Phytium-FT2000-4:/bin# cd /usr/local/python3/
root@kylin-Phytium-FT2000-4:/usr/local/python3# ll
总用量 8
drwxr-xr-x  2 root root 4096 4月  14 10:29 ./
drwxr-xr-x 11 root root 4096 4月  14 09:14 ../
lrwxrwxrwx  1 root root   30 4月  14 10:29 python3 -> /usr/local/python3/bin/python3
```

4. 添加 pip3 的软链接
   sudo ln -s /usr/local/bin/pip3.7 /usr/bin/pip3.7

```
root@kylin-Phytium-FT2000-4:/usr/bin# ln -s /usr/local/bin/pip3.7 /usr/bin/pip3.7
```

   sudo rm -rf /usr/bin/python3
   sudo rm -rf /usr/bin/pip3
5. 查看 python 版本

```
root@kylin-Phytium-FT2000-4:~# python3 --version
Python 3.7.3
```

5.7 GCC 升级 9.4.0

1. 新建 gcc 目录
   Mkdir gcc9.4.0
   Cd gcc9.4.0
2. 下载 gcc,gmp，mpfr，mpc，
   Wget https://ftp.gnu.org/gnu/gcc/gcc-9.4.0/gcc-9.4.0.tar.xz
   wget https://mirrors.tuna.tsinghua.edu.cn/gnu/gmp/gmp-6.2.1.tar.xz
   wget https://mirrors.tuna.tsinghua.edu.cn/gnu/mpfr/mpfr-4.1.0.tar.xz
   wget https://mirrors.tuna.tsinghua.edu.cn/gnu/mpc/mpc-1.2.1.tar.gz

```
root@kylin-Phytium-FT2000-4:~/gcc9.4.0# wget https://mirrors.tuna.tsinghua.edu.cn/gnu/gmp/gmp-6.2.1.tar.xz
--2022-04-14 11:10:02--  https://mirrors.tuna.tsinghua.edu.cn/gnu/gmp/gmp-6.2.1.tar.xz
正在解析主机 mirrors.tuna.tsinghua.edu.cn (mirrors.tuna.tsinghua.edu.cn)... 101.6.15.130, 2402:f000:1:400::2
正在连接 mirrors.tuna.tsinghua.edu.cn (mirrors.tuna.tsinghua.edu.cn)|101.6.15.130|:443... 已连接。
错误：无法验证 mirrors.tuna.tsinghua.edu.cn 的由 “CN=R3,O=Let's Encrypt,C=US” 颁发的证书：
  颁发的证书已经过期。
要以不安全的方式连接至 mirrors.tuna.tsinghua.edu.cn，使用“--no-check-certificate”。
root@kylin-Phytium-FT2000-4:~/gcc9.4.0# wget --no-check-certificate  https://mirrors.tuna.tsinghua.edu.cn/gnu/gmp/gmp-6.2.1.tar.xz
--2022-04-14 11:10:32--  https://mirrors.tuna.tsinghua.edu.cn/gnu/gmp/gmp-6.2.1.tar.xz
正在解析主机 mirrors.tuna.tsinghua.edu.cn (mirrors.tuna.tsinghua.edu.cn)... 101.6.15.130, 2402:f000:1:400::2
正在连接 mirrors.tuna.tsinghua.edu.cn (mirrors.tuna.tsinghua.edu.cn)|101.6.15.130|:443... 已连接。
警告：无法验证 mirrors.tuna.tsinghua.edu.cn 的由 “CN=R3,O=Let's Encrypt,C=US” 颁发的证书：
  颁发的证书已经过期。
已发出 HTTP 请求，正在等待回应... 200 OK
长度：2027316 (1.9M) [application/octet-stream]
正在保存至：“gmp-6.2.1.tar.xz”

gmp-6.2.1.tar.xz                    100%[=================================================================>]   1.93M  1.52MB/s    in

2022-04-14 11:10:33 (1.52 MB/s) - 已保存 “gmp-6.2.1.tar.xz” [2027316/2027316])
```

3. 解压 gmp-6.2.1.tar.xz，进行安装
   tar xvf gmp-6.2.1.tar.xz
   cd gmp-6.2.1

./configure --prefix=/usr/local/gmp-6.2.1

```
root@kylin-Phytium-FT2000-4:~/gcc9.4.0/gmp-6.2.1# ./configure --prefix=/usr/local/gmp-6.2.1/
```

Make -j4

Make install

Cd ..

4. 解压 mpfr-4.1.0.tar.xz，进行安装

tar -xvf mpfr-4.1.0.tar.xz

cd mpfr-4.1.0

./configure --prefix=/usr/local/mpfr-4.1.0 --with-gmp=/usr/local/gmp-6.2.1/

```
root@kylin-Phytium-FT2000-4:~/gcc9.4.0/mpfr-4.1.0# ./configure --prefix=/usr/local/mpc-1.2.1/ --with-gmp=/u
sr/local/gmp-6.2.1/ --with-mpfr=/usr/local/mpfr-4.1.0/
```

Make -j4

Make install

5. 解压 mpc-1.2.1.tar.gz，进行安装

tar -xvf mpc-1.2.1.tar.gz

cd mpc-1.2.1

./configure --prefix=/usr/local/mpc-1.2.1 --with-gmp=/usr/local/gmp-6.2.1/ --with-mpfr=/usr/local/mpfr-4.1.0

```
root@kylin-Phytium-FT2000-4:~/gcc9.4.0/mpc-1.2.1# ./configure --prefix=/usr/local/gcc-9.4.0/ --enable-check
ing=release --enable-languages=c,c++ --disable-multilib --with-gmp=/usr/local/gmp-6.2.1/ --with-mpfr=/usr/l
ocal/mpfr-4.1.0/ --with-mpc=/usr/local/mpc-1.2.1/
```

Make -j4

Make install

6. 解压 tar -xvf gcc-9.4.0.tar.gz ，进行安装

1) tar zxvf gcc-9.4.0.tar.gz

2) Cd gcc-9.4.0/

3) ./configure --prefix=/usr/local/gcc-9.4.0/ --enable-checking=release --enable-languages=c,c++ --disable-multilib --with-gmp=/usr/local/gmp-6.2.1/ --with-mpfr=/usr/local/mpfr-4.1.0/ --with-mpc=/usr/local/mpc-1.2.1/

```
root@kylin-Phytium-FT2000-4:~/gcc9.4.0/gcc-9.4.0# ./configure --prefix=/usr/local/gcc-9.4.0/ --enable-check
ing=release --enable-languages=c,c++ --disable-multilib --with-gmp=/usr/local/gmp-6.2.1/ --with-mpfr=/usr/l
ocal/mpfr-4.1.0/ --with-mpc=/usr/local/mpc-1.2.1/
```

4) 添加下面一行，否则会报错

vi /etc/ld.so.conf

/usr/local/mpfr-4.1.0/lib

:wq

#Ldconfig

不执行的会安装 gcc 报错 libmpfr.so.6: cannot open shared object file: No such file or directory

5) Make -j4

Make install

```
If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,-rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
make[4]: Nothing to be done for 'install-data-am'.
make[4]: Leaving directory '/root/gcc9.4.0/gcc-9.4.0/aarch64-unknown-linux-gnu/libatomic'
make[3]: Leaving directory '/root/gcc9.4.0/gcc-9.4.0/aarch64-unknown-linux-gnu/libatomic'
make[2]: Leaving directory '/root/gcc9.4.0/gcc-9.4.0/aarch64-unknown-linux-gnu/libatomic'
make[1]: Leaving directory '/root/gcc9.4.0/gcc-9.4.0'
root@kylin-Phytium-FT2000-4:~/gcc9.4.0/gcc-9.4.0# ~
```

6）删除旧版本链接

rm /usr/bin/gcc

rm /usr/bin/g++

rm /usr/bin/c++

rm /usr/bin/cc
rm /usr/lib/libstdc++.so.6

7）建立软链接

```
root@kylin-Phytium-FT2000-4:~# ln -s /usr/local/gcc-9.4.0/bin/gcc /usr/bin/gcc
root@kylin-Phytium-FT2000-4:~# ln -s /usr/local/gcc-9.4.0/bin/g++ /usr/bin/g++
root@kylin-Phytium-FT2000-4:~# ln -s /usr/local/gcc-9.4.0/bin/c++ /usr/bin/c++
root@kylin-Phytium-FT2000-4:~# ln -s /usr/local/gcc-9.4.0/bin/gc /usr/bin/c++
root@kylin-Phytium-FT2000-4:~# ln -s /usr/local/gcc-9.4.0/bin/gcc  /usr/bin/cc
```

8）升级库文件软连接最新库文件

```
root@kylin-Phytium-FT2000-4:~# ln -s /usr/local/gcc-9.4.0/lib64/libstdc++.so.6.0.28 /usr/lib/libstdc++.so.6
```

9）查看 gcc 版本

```
root@kylin-Phytium-FT2000-4:~# gcc -v
使用内建 specs。
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/local/gcc-9.4.0/libexec/gcc/aarch64-unknown-linux-gnu/9.4.0/lto-wrapper
目标：aarch64-unknown-linux-gnu
配置为：./configure --prefix=/usr/local/gcc-9.4.0/ --enable-checking=release --enable-languages=c,c++ --dis
able-multilib --with-gmp=/usr/local/gmp-6.2.1/ --with-mpfr=/usr/local/mpfr-4.1.0/ --with-mpc=/usr/local/mpc
-1.2.1/
线程模型：posix
gcc 版本 9.4.0 (GCC)
root@kylin-Phytium-FT2000-4:~#
```

六、 使用开发板在线开发程序

（以下代码都是文字加贴图）

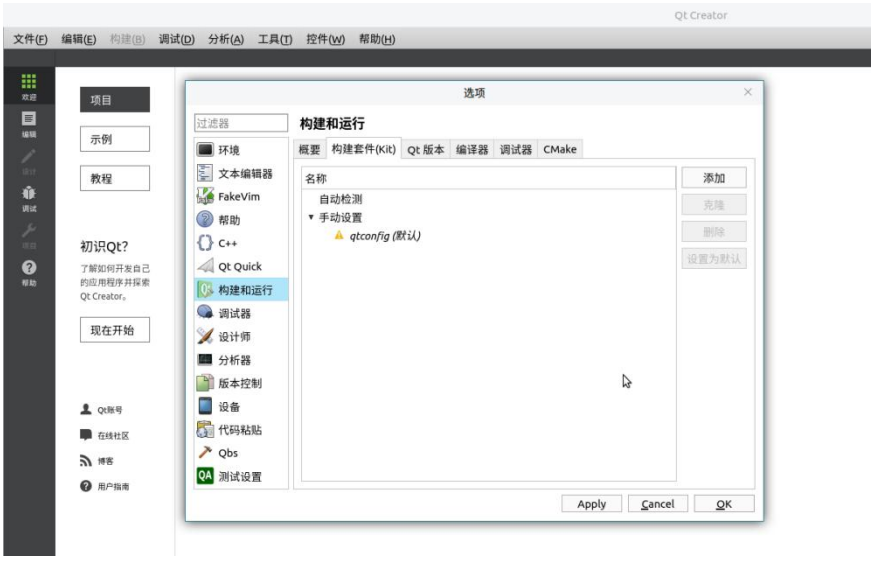
6.1 搭建QT环境

1) apt-get remove qt*

sudo apt-get install qt5*

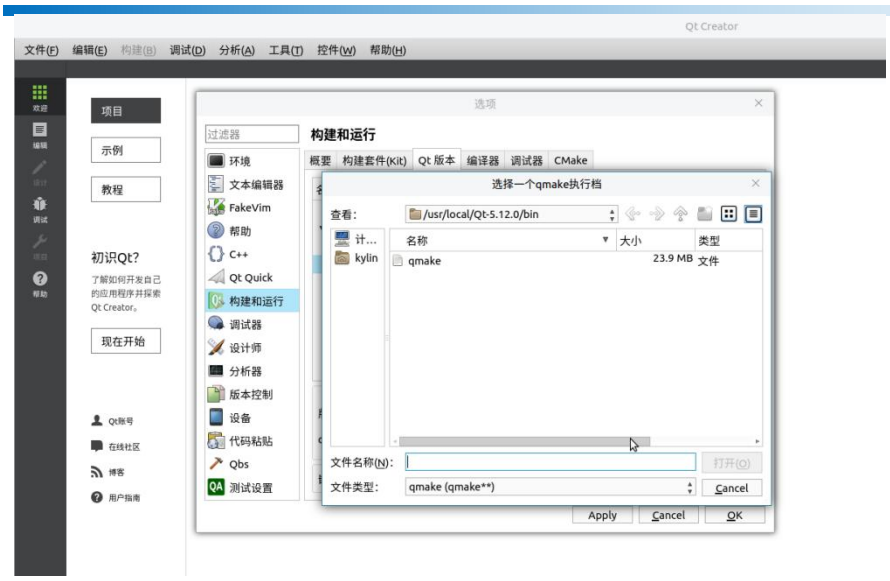
apt-get install qtcreator

2) 打开麒麟自带的 Qt creator工具，然后点击“工具”，再点击“选项”进入到选项界面

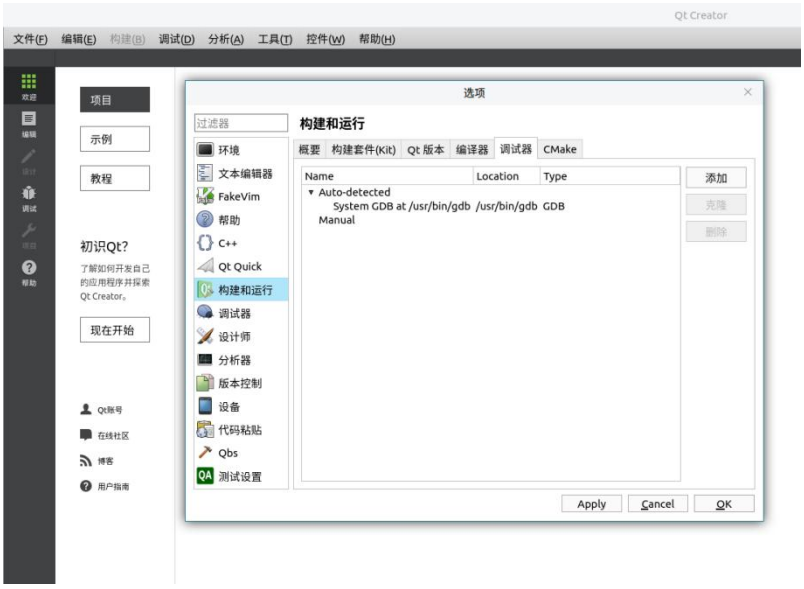


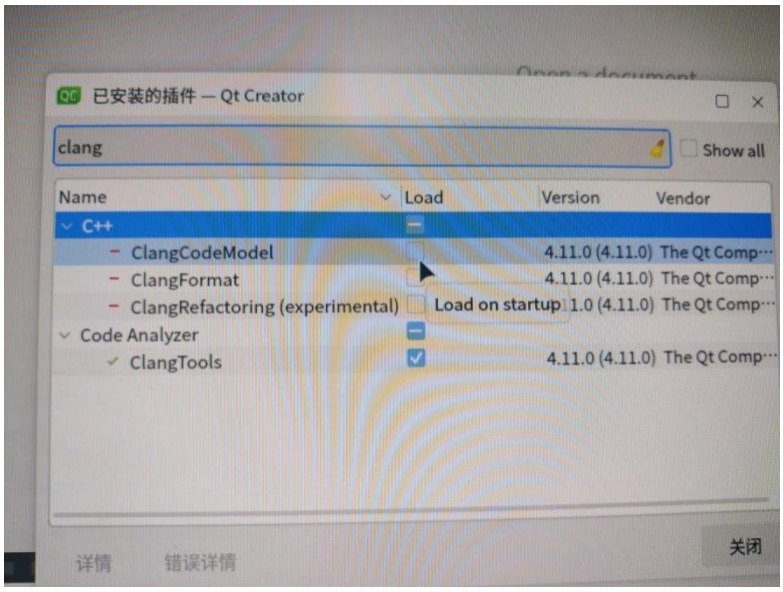
3) 选择qt版本选项，看是否能自动识别到QT-5.12.0,不能自动识别到就重启一下。

或者手动添加一个，找到/usr/local/QT-5.12.0/bin路径下的qmake，点击打开。


Qt Creator
文件(F) 编辑(E) 构建(B) 调试(D) 分析(A) 工具(T) 控件(W) 帮助(H)
项目
示例
教程
初识Qt?
现在开始
选项
构建和运行
过滤器
环境
文本编辑器
FakeVim
帮助
C++
Qt Quick
构建和运行
调试器
设计师
分析器
版本控制
设备
代码粘贴
Qbs
测试设置
概要 构建套件(Kit) Qt 版本 编译器 调试器 CMake
选择一个qmake执行档
查看: /usr/local/Qt-5.12.0/bin
kylin
名称 大小 类型
qmake 23.9 MB 文件
文件名称(N):
文件类型: qmake (qmake**)
打开(O)
Cancel
Apply
Cancel
OK

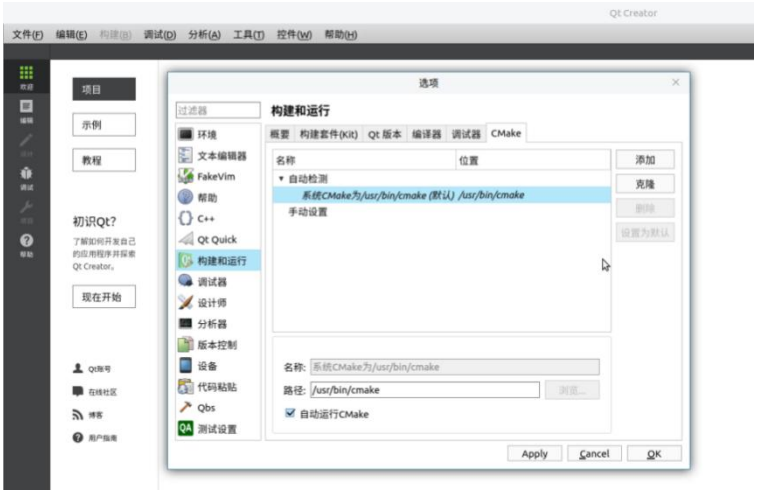
4）确认默认的调试器和CMAKE是否已经配置了





5）还要把这个clangCodeModel的勾去掉。

6）若如上已配置好，则可以构建套件kit了。



6.2 搭建opencv环境

1）先搭建好opencv环境，安装编译工具：sudo apt-get install build-essential

2）安装 cmake、git、pkg-config等辅助工具：sudo apt-get install cmake git pkg-config libgtk2.0-dev libavcodec-dev libavformat-dev libswscale-dev

3）安装关联库：sudo apt-get install python-dev python-opencv python-numpy libtbb2 libtbb-dev libjpeg-dev libpng-dev libtiff-dev libjasper-dev libdc1394-22-dev

4）上传opencv-3.4.14.zip，终端输入：unzip opencv-3.4.14.zip

5）解压完成后将在改目录下生成opencv-3.4.14目录，进入该目录，新建一个目录进行OpenCV的编译目录（否则会报以下错误），mkdir build

cd bulid

cmake ..

执行完毕后终端输入：make -j4

make install

6）等make install执行完毕之后，需要对环境变量进行配置

终端输入：vim /etc/ld.so.conf.d/opencv.conf

在文件末尾添加：/usr/local/lib（可能是空文件）

7）配置环境变量

终端输入：vim /etc/bash.bashrc

在文件末尾添加：

PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/lib/pkgconfig

export PKG_CONFIG_PATH

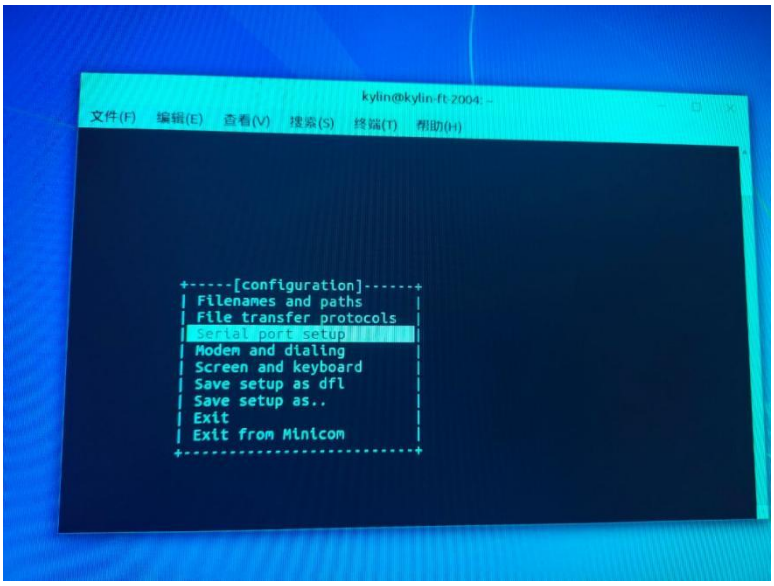
6.3 安装minicom

1) 登录kylin PC，键盘输入Ctrl+Alt+T，弹出命令行终端，在命令行终端输入如下命令即可安装

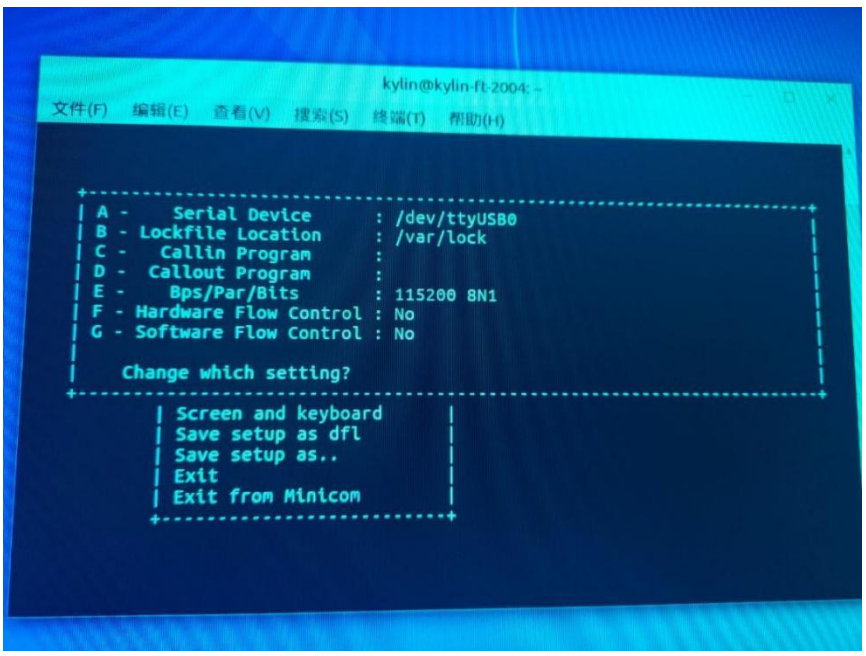
minicom程序
```
linux@kylin:~$ sudo apt-get install minicom
```
2) 配置minicom

```
linux@kylin:~$ sudo minicom -s
```

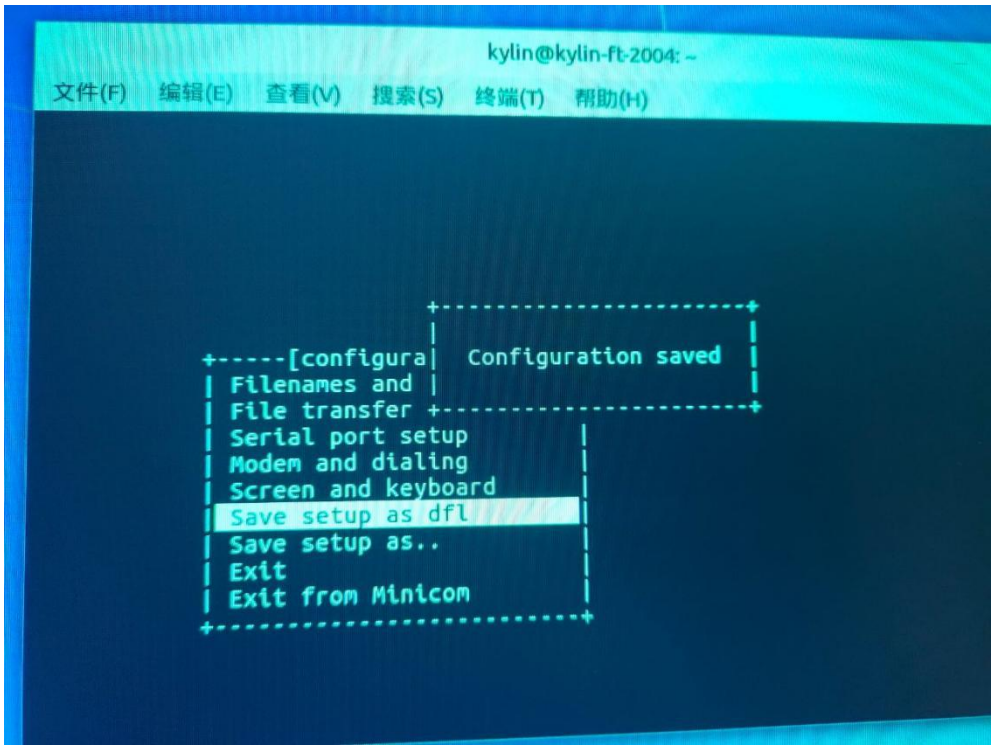
弹出如下界面：



- 使用上下键选择“Serial port setup”，回车进入串口设置。
- 键盘输入字符a，修改“Serial Device”为/dev/ttyUSB0，然后回车保存。
- 键盘输入字符f，修改“Hardware Flow Control”为No，然后回车保存。
- 配置成功后，如下图所示：



- 返回主菜单，选择“Save setup as dfl”将其保存成默认配置



- 最后选择“Exit From Minicom”，退出minicom

6.4 在线开发TCP程序

1) 打开QT，创建一个新项目，点击文件>新建文件项目>Application>Qt Widgets Application>choose

填写项目名称TCP，点击下一步，在头文件新建server.h，源文件新建server.cpp

2) 添加服务端代码(server.h跟server.cpp)：

```
#ifndef SERVER_H
#define SERVER_H

#include <QWidget>

#include <QTcpServer>//监听套接字
#include <QTcpSocket>//通信套接字

namespace Ui {
class server;
}

class server : public QWidget
{
    Q_OBJECT

public:
    explicit server(QWidget *parent = 0);
    ~server();

private slots:
    void on_sendButton_clicked();

    void on_choseButton_clicked();

private:
    Ui::server *ui;
    QTcpServer  *tcpServer;//监听套接字
    QTcpSocket  *tcpSocket;//通信套接字
};

#endif // SERVER_H
```

```
#include "server.h"
#include "ui_server.h"
```

```
server::server(QWidget *parent) :
    QWidget(parent),
    ui(new Ui::server)
{
    ui->setupUi(this);
    tcpServer = NULL;
    tcpSocket = NULL;

    //监听套接字    指定父对象的目的，让其自动回收空间
    tcpServer = new QTcpServer(this);

    tcpServer->listen(QHostAddress::Any,8888);

    setWindowTitle("server");

    connect(tcpServer,&QTcpServer::newConnection,[=](){
        //取出建立好连接的的套接字
        tcpSocket = tcpServer->nextPendingConnection();

        //获取对方的IP和端口
        QString ip = tcpSocket->peerAddress().toString();
        qint16  port = tcpSocket->peerPort();
        QString temp = QString("[%1:%2]:成功连接").arg(ip).arg(port);

        ui->textEditRead->setText(temp);

            /*注意这里的connect的位置，一定在建立好连接后，
             * 不然会出现野指针的错误，前面定义的QTcpSocket  *tcpSocket;
             * 而程序会先从构造函数执行的，还没有执行到定义的QTcpSocket  *tcpSocket
指针。
            */
            connect(tcpSocket,&QTcpSocket::readyRead,[=](){

                //从通信套接字中取出内容
                QByteArray array = tcpSocket->readAll();

                ui->textEditRead->append(array);
            });

        });

}

server::~server()
{
    delete ui;
}
```

```
void server::on_sendButton_clicked()
{
    if(NULL == tcpSocket)
    {
        return ;
    }
    //获取编辑区的内容
    QString str = ui->textEditWrite->toPlainText();

    //给对方发送数据，使用套接字是tcpSocket
    tcpSocket->write(str.toUtf8().data());
}

void server::on_choseButton_clicked()
{
    if(NULL == tcpSocket)
    {
        return ;
    }
    //主动和客户端端口断开连接
    tcpSocket->disconnectFromHost();
    tcpSocket->close();

    tcpSocket = NULL;
}
```

server.cpp - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
#include "server.h"
#include "ui_server.h"

server::server(QWidget *parent) :
    QWidget(parent),
    ui(new Ui::server)
{
    ui->setupUi(this);
    tcpServer = NULL;
    tcpSocket = NULL;

    //监听套接字  指定父对象的目的，让其自动回收空间
    tcpServer = new QTcpServer(this);

    tcpServer->listen(QHostAddress::Any,8888);

    setWindowTitle("server");

    connect(tcpServer,&QTcpServer::newConnection,[=](){
        //取出建立好连接的的套接字
        tcpSocket = tcpServer->nextPendingConnection();

        //获取对方的IP和端口
        QString ip = tcpSocket->peerAddress().toString();
        qint16  port = tcpSocket->peerPort();
        QString temp = QString("[%1:%2]:成功连接").arg(ip).arg(port);

        ui->textEditRead->setText(temp);

            /*注意这里的connect的位置，一定在建立好连接后，
             * 不然会出现野指针的错误，前面定义的QTcpSocket  *tcpSocket;
             * 而程序会先从构造函数执行的，还没有执行到定义的QTcpSocket  *tcpSocket指针。
            */
            connect(tcpSocket,&QTcpSocket::readyRead,[=](){

                //从通信套接字中取出内容
                QByteArray array = tcpSocket->readAll();

                ui->textEditRead->append(array);
            });

        });
```

server.h - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
#ifndef SERVER_H
#define SERVER_H

#include <QWidget>

#include <QTcpServer>//监听套接字
#include <QTcpSocket>//通信套接字

namespace Ui {
class server;
}

class server : public QWidget
{
    Q_OBJECT

public:
    explicit server(QWidget *parent = 0);
    ~server();

private slots:
    void on_sendButton_clicked();

    void on_choseButton_clicked();

private:
    Ui::server *ui;
    QTcpServer  *tcpServer;//监听套接字
    QTcpSocket  *tcpSocket;//通信套接字
};

#endif // SERVER_H
```

3） 添加客户端代码(client.h跟client.cpp)：

```
#ifndef CLIENT_H
#define CLIENT_H

#include <QWidget>
#include <QTcpSocket>//仅需通信套接字

namespace Ui {
class client;
```

```
}

class client : public QWidget
{
    Q_OBJECT

public:
    explicit client(QWidget *parent = 0);
    ~client();

private slots:
    void on_pushButtonsend_clicked();

    void on_pushButtonconnect_clicked();

    void on_pushButtonclose_clicked();

private:
    Ui::client *ui;
    QTcpSocket  *tcpSocket;//通信套接字
};

#endif // CLIENT_H
```

```
#include "client.h"
#include "ui_client.h"
#include <QHostAddress>

client::client(QWidget *parent) :
    QWidget(parent),
    ui(new Ui::client)
{
    ui->setupUi(this);

    tcpSocket = NULL;

    //分配空间，指定父对象
    tcpSocket = new QTcpSocket(this);


    connect(tcpSocket,&QTcpSocket::connected,[=](){
```

```
        ui->textEditRead->setText("成功和服务器建立好连接");

    });

    connect(tcpSocket,&QTcpSocket::readyRead,[=](){
        //获取对方发送的内容
        QByteArray array = tcpSocket->readAll();

        //追加到编辑区中
        ui->textEditRead->append(array);

    });
}

client::~client()
{
    delete ui;


}


void client::on_pushButtonsend_clicked()
{
    //获取编辑区的内容
    QString str = ui->textEditWrite->toPlainText();

    //发送数据，使用套接字是tcpSocket
    tcpSocket->write(str.toUtf8().data());
}

void client::on_pushButtonconnect_clicked()
{
    //获取服务器的ip和端口
    QString ip = ui->lineEditIP->text();
    qint16 port = ui->lineEditPort->text().toInt();

    //主动和服务器建立连接
    tcpSocket->connectToHost(QHostAddress(ip),port);
}

void client::on_pushButtonclose_clicked()
{
    if(NULL == tcpSocket)
    {
        return ;
    }
```

```
    //主动和客户端端口断开连接
    tcpSocket->disconnectFromHost();
    tcpSocket->close();

    tcpSocket = NULL;
}
```

client.h - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
#ifndef CLIENT_H
#define CLIENT_H

#include <QWidget>
#include <QTcpSocket>//仅需通信套接字

namespace Ui {
class client;
}

class client : public QWidget
{
    Q_OBJECT

public:
    explicit client(QWidget *parent = 0);
    ~client();

private slots:
    void on_pushButtonsend_clicked();

    void on_pushButtonconnect_clicked();

    void on_pushButtonclose_clicked();

private:
    Ui::client *ui;
    QTcpSocket  *tcpSocket;//通信套接字
};

#endif // CLIENT_H
```

client.cpp - 记事本

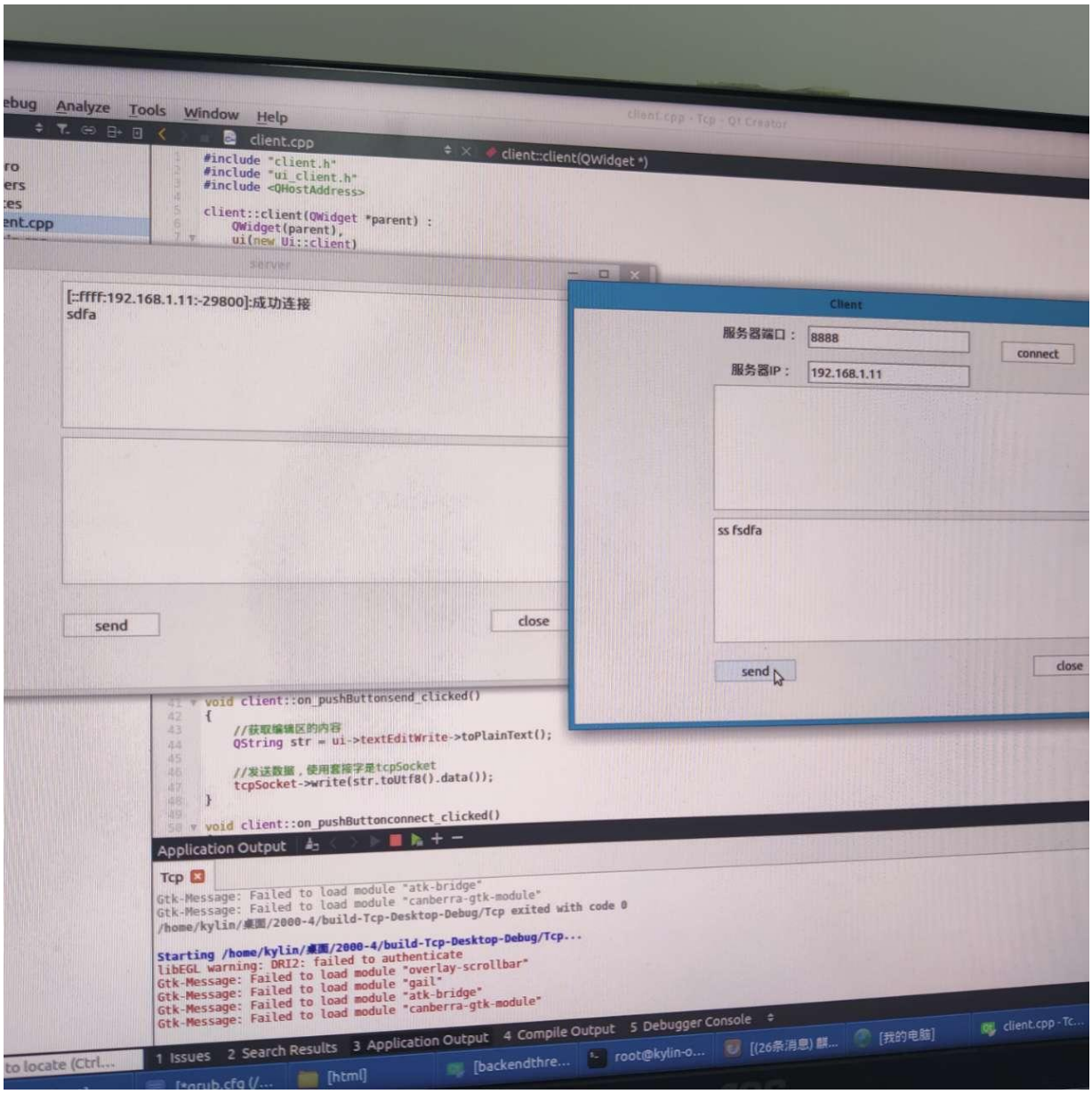
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
#include "client.h"
#include "ui_client.h"
#include <QHostAddress>

client::client(QWidget *parent) :
    QWidget(parent),
    ui(new Ui::client)
{
    ui->setupUi(this);

    tcpSocket = NULL;

    //分配空间，指定父对象
    tcpSocket = new QTcpSocket(this);


    connect(tcpSocket,&QTcpSocket::connected,[=](){

        ui->textEditRead->setText("成功和服务器建立好连接");

    });

    connect(tcpSocket,&QTcpSocket::readyRead,[=](){
        //获取对方发送的内容
        QByteArray array = tcpSocket->readAll();

        //追加到编辑区中
        ui->textEditRead->append(array);

    });
}

client::~client()
{
    delete ui;


}
```

4） 到ui界面拖动图标画ui

5） 点击右下角绿色箭头运行TCP程序

6） 运行成功后会看到如下图：



6.5 在线开发Canny实例

1） 创建Canny实例

```
root@kylin:~# touch canny.cpp
```

2） 添加如下代码：

```
#include<opencv2/opencv.hpp>
#include<opencv2/highgui/highgui.hpp>

using namespace std;
using namespace cv;

//边缘检测1
int main()
{
    Mat img = imread("jiao.jpg");
    imshow("原始图", img);
    Mat DstPic, edge, grayImage;

    //创建与src同类型和同大小的矩阵
    DstPic.create(img.size(), img.type());

    //将原始图转化为灰度图
    cvtColor(img, grayImage, COLOR_BGR2GRAY);

    //先使用3*3内核来降噪
    blur(grayImage, edge, Size(3, 3));

    //运行canny算子
    Canny(edge, edge, 3, 9, 3);

    imshow("边缘提取效果", edge);

    waitKey(0);
}
```

```
#include<opencv2/highgui/highgui.hpp>

using namespace std;
using namespace cv;

//边缘检测1
int main()
{
        Mat img = imread("jiao.jpg");
        imshow("原始图", img);
        Mat DstPic, edge, grayImage;

        //创建与src同类型和同大小的矩阵
        DstPic.create(img.size(), img.type());

        //将原始图转化为灰度图
        cvtColor(img, grayImage, COLOR_BGR2GRAY);

        //先使用3*3内核来降噪
        blur(grayImage, edge, Size(3, 3));

        //运行canny算子
        Canny(edge, edge, 3, 9, 3);

        imshow("边缘提取效果", edge);

        waitKey(0);

}
```

3) root@kylin:~# g++ canny.cpp `pkg-config --cflags --libs opencv` -o canny

4) 结果演示

6.6 在线开发Laplace实例

1） 创建laplace.cpp

root@kylin:~# touch laplace.cpp

2） 添加如下代码：

```
#include<opencv2/opencv.hpp>
#include<opencv2/highgui/highgui.hpp>

using namespace std;
using namespace cv;

//边缘检测
int main()
{
    Mat img = imread("jiao.jpg");
    imshow("原始图", img);
    Mat gray, dst,abs_dst;
    //高斯滤波消除噪声
    GaussianBlur(img, img, Size(3, 3), 0, 0, BORDER_DEFAULT);
    //转换为灰度图
    cvtColor(img, gray, COLOR_RGB2GRAY);
    //使用Laplace函数
    //第三个参数：目标图像深度；第四个参数：滤波器孔径尺寸；第五个参数：比例因子；第六个参数：表示结果存入目标图
    Laplacian(gray, dst, CV_16S, 3, 1, 0, BORDER_DEFAULT);
    //计算绝对值，并将结果转为8位
    convertScaleAbs(dst, abs_dst);

    imshow("laplace效果图", abs_dst);

    waitKey(0);

}
```

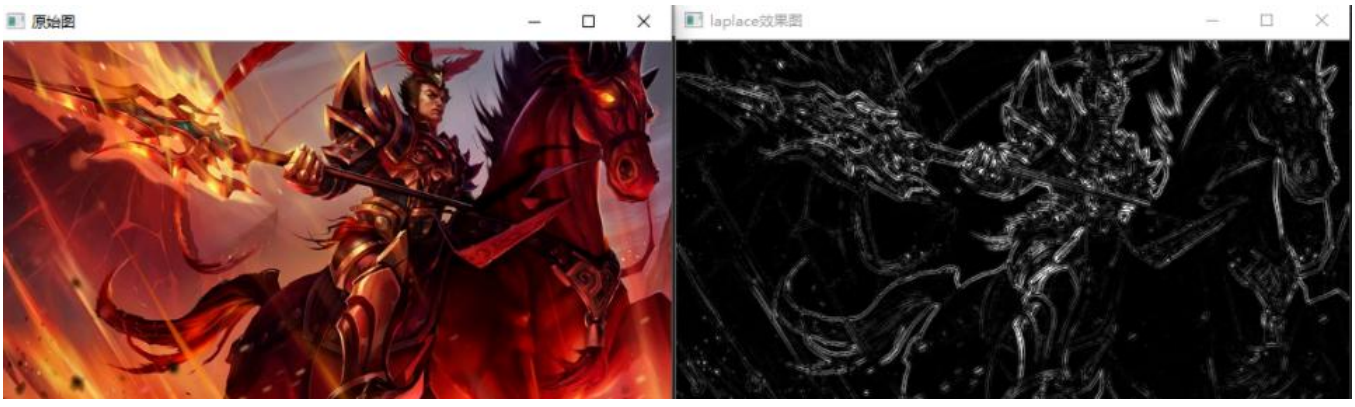
laplace.cpp - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
#include<opencv2/opencv.hpp>
#include<opencv2/highgui/highgui.hpp>

using namespace std;
using namespace cv;

//边缘检测
int main()
{
    Mat img = imread("jiao.jpg");
    imshow("原始图", img);
    Mat gray, dst,abs_dst;
    //高斯滤波消除噪声
    GaussianBlur(img, img, Size(3, 3), 0, 0, BORDER_DEFAULT);
    //转换为灰度图
    cvtColor(img, gray, COLOR_RGB2GRAY);
    //使用Laplace函数
    //第三个参数：目标图像深度；第四个参数：滤波器孔径尺寸；第五个参数：比例因子；第六个参数：表示结果存入目标图
    Laplacian(gray, dst, CV_16S, 3, 1, 0, BORDER_DEFAULT);
    //计算绝对值，并将结果转为8位
    convertScaleAbs(dst, abs_dst);

    imshow("laplace效果图", abs_dst);

    waitKey(0);

}
```

3) root@kylin:~# g++ laplace.cpp `pkg-config --cflags --libs opencv` -o laplace

4) 结果演示



6.7 在线开发Sobel实例

1） 创建sobel.cpp

root@kylin:~# touch sobel.cpp

2） 添加如下代码：

```
#include<opencv2/opencv.hpp>
#include<opencv2/highgui/highgui.hpp>

using namespace std;
using namespace cv;

//边缘检测
int main()
{
    Mat img = imread("jiao.jpg");

    imshow("原始图", img);

    Mat grad_x, grad_y;
    Mat abs_grad_x, abs_grad_y, dst;

    //求x方向梯度
    Sobel(img, grad_x, CV_16S, 1, 0, 3, 1, 1,BORDER_DEFAULT);
    convertScaleAbs(grad_x, abs_grad_x);
    imshow("x方向soble", abs_grad_x);

    //求y方向梯度
    Sobel(img, grad_y,CV_16S,0, 1,3, 1, 1, BORDER_DEFAULT);
    convertScaleAbs(grad_y,abs_grad_y);
    imshow("y向soble", abs_grad_y);

    //合并梯度
    addWeighted(abs_grad_x, 0.5, abs_grad_y, 0.5, 0, dst);
    imshow("整体方向soble", dst);


    waitKey(0);

}
```

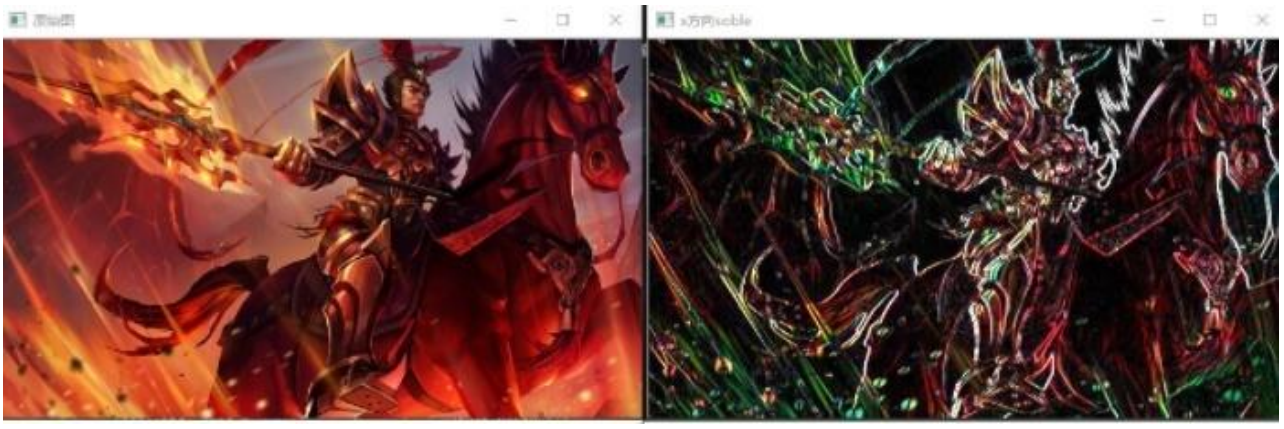
sobel.cpp - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
//边缘检测
int main()
{
    Mat img = imread("jiao.jpg");

    imshow("原始图", img);

    Mat grad_x, grad_y;
    Mat abs_grad_x, abs_grad_y, dst;

    //求x方向梯度
    Sobel(img, grad_x, CV_16S, 1, 0, 3, 1, 1,BORDER_DEFAULT);
    convertScaleAbs(grad_x, abs_grad_x);
    imshow("x方向soble", abs_grad_x);

    //求y方向梯度
    Sobel(img, grad_y,CV_16S,0, 1,3, 1, 1, BORDER_DEFAULT);
    convertScaleAbs(grad_y,abs_grad_y);
    imshow("y向soble", abs_grad_y);

    //合并梯度
    addWeighted(abs_grad_x, 0.5, abs_grad_y, 0.5, 0, dst);
    imshow("整体方向soble", dst);


    waitKey(0);

}
```

3) root@kylin:~# g++ sobel.cpp `pkg-config --cflags --libs opencv` -o sobel

4) 结果演示



6.8 FT2000/4平台实时时钟实例

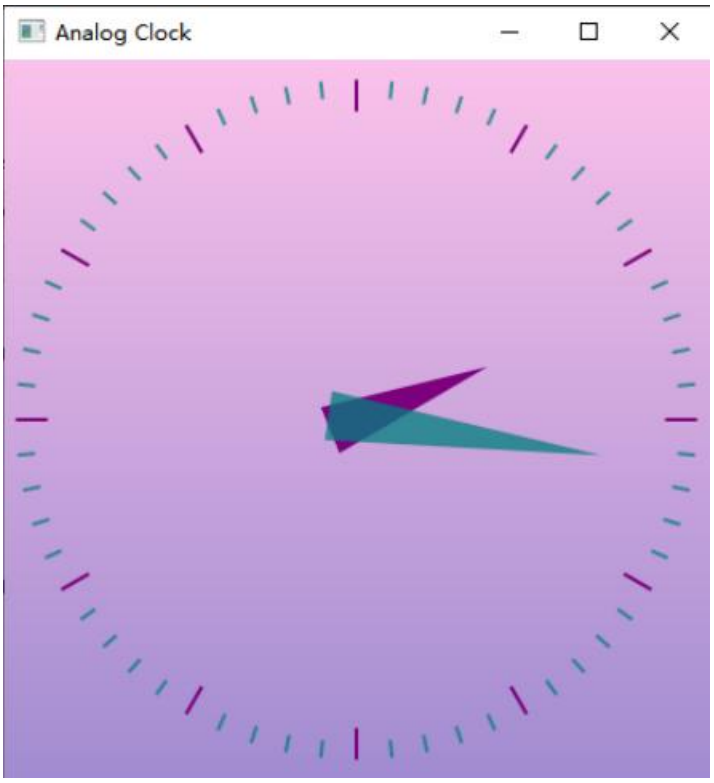
1) 在解压目录下解压开发板参考实例

   root@kylin:~# unzip analogclock.zip

2) 打开QT，点击文件>打开文件项目>选择刚解压的文件夹>analogclock.pro.user

   点击右下角绿色箭头运行程序

3) 结果演示



六、 MNN推理引擎

7.1 MNN介绍

MNN 是一个轻量级的深度神经网络推理引擎，在端侧加载网络模型进行推理预测。

7.2 MNN运行实例

1) 在家目录下解压开发板参考实例

root@kylin:~# tar zxvf F275_examples_1.0.tar.gz

2) 安装 MNN 依赖组件

root@kylin:~# apt-get install git libprotobuf-dev protobuf-compiler

3) 下载 MNN 推理引擎，并且切换到 1.1.6 版本

root@kylin:~# git clone https://gitee.com/mirrors/mnn.git

root@kylin:~# git checkout 1.1.6

4) 编译 MNN 推理引擎，编译时间大概为 10 分钟

编译成功后，mnn/build 目录下会生成多个可执行文件，其中：

MNNConvert 为模型转换工具

multiPose.out 为姿态检测实例

segment.out 为图像分割实例

pictureRecognition.out 为图像分类实例

cd mnn

./schema/generate.sh

mkdir build && cd build

cmake -DMNN_BUILD_DEMO=ON -DMNN_BUILD_CONVERTER=true ..

make -j4

5) 模型转换，将 Tensorflow Lite 模型转换为 mnn 模型

./MNNConvert -f TFLITE --modelFile ~/examples/AI/models/deeplabv3_257_mv_gpu.tflite --

MNNModel deeplabv3_257_mv_gpu_armv8.mnn --bizCode 0000

注意，板卡已经自带多个网络模型，具体路径为

examples/AI/models/deeplabv3_257_mv_gpu.tflite

examples/AI/models/mobilenet_v2_1.0_224.tflite

examples/AI/models/model-mobilenet_v1_075.pb

6) 执行图像分割实例

./segment.out deeplabv3_257_mv_gpu_armv8.mnn ~/examples/AI/test_images/test1.jpg

result.png

执行结果比较，左边为原图，右边为分割后的图像

七、 常见问题解决

7.1 调试串口没有打印信息！

答：需要排除以下故障：

1) 接入19V电源后是否按下6电源开关？
2) 检查开发板上的丝印信息，确保调试串口线连接正确，即调试串口线的绿线连接开发板RX管脚、白线连接TX管脚、黑线连接GND管脚
3) 检查调试串口线是否松动？

7.2 自动获取IP地址报错！

答：使用dhclient -i enaphyt4i0命令报错，报错信息为RTNETLINK answers: File exists。

解决方案：

1) 首先刷掉enaphyt4i0的ip地址

```
root@kylin:~# ip addr flush dev enaphyt4i0
```

2) 重新获取地址

```
root@kylin:~# dhclient -i enaphyt4i0
```

7.3 MXM显卡无法显示图像？

答：

1) 使用如下命令测试开发板是否识别AMD显卡?

```
root@kylin:~# lspci | grep AMD
```

2) 检查MXM显卡型号是否为AMD HD8570或者AMD R5 230
3) 检查散热风扇电源是否连接到开发板FAN1(14)接口
4) 请直接使用HDMI线连接到HDMI显示器，严禁使用HDMI转VGA或者HDMI转DVI
5) 拔掉开发板CPU Debug串口(10)接口上的调试串口线
6) 尝试换一台HDMI显示器进行测试

7.4 SSH无法远程登录问题？

答：关闭防火墙iptables -F

八、 支持与服务

如遇到任何问题，请与我们联系或者登录官方网站。

天固信安官网：www.skysolidiss.com.cn

销售邮箱：yanlg@skysolidiss.com.cn

技术支持：support@skysolidiss.com.cn

九、 免责声明

本文中的信息，包括参考的URL地址，如有变更，恕不另行通知。使用本文档内容产生的任何侵权，本公司不负任何责任。文中提到的所有商标名称、商标和注册商标均属其各自所有者的财产，特此声明。

十、 版权公告