ORIENT DISPLAY

Your Total LCD Solution Provider

http://www.orientdisplay.com

USER MANUAL FOR CONTROL BOARD PRODUCT NO.: AIYA002M

Linux Manual Ver2.0

Base NXP I.MX6U

ARM@ Cortex@-A7 Processor

Embedded Development Platform

Customer's Approval:

	SIGNATURE	DATE
PREPARED BY (RD ENGINEER)		
CHECKED BY		
APPROVED BY		

Orient Display (N.A.) Ltd.
220 Royal Crest Court, Unit 06,Markham,ON,Canada L3R 9Y2
Tel: 905-477-1166 Fax: 905-477-1782

Orient Display (USA) Corp.
14925 SE Allen Road, Suite 203 B, Bellevue, WA 98006
Tel: (425)698-1938 Fax: (425)698-1852

Contents

ATTENTIONS

- Hot-plug of core board and peripheral modules is strictly prohibited.
- Please follow all the warnings and instructions marked on the product.
- Please always keep the product dry. Once it is splashed or immersed by any liquid, cut off the power and dry it out immediately.
- Please store and operate the product in ventilating conditions to avoid damages brought by overhigh temperature.
- Please do not use or store the product in dusty or untidy conditions.
- Please do not use or store the product in alternate cold and hot conditions to avoid condensing which will damage components.
- Please do not treat the product rudely. Any falling-off, knocking and violate shaking may cause destruction to circuit and components.
- Please do not clean the product with organic solvents or corrodible liquids.
- Please do not dismantle or repair the product by yourself. Contact us when the product malfunctions.
- Please do not modify the product by yourself or use accessories unauthorized by us. Otherwise, the damage caused by that will be on your part and not included in guarantee terms.

Contact Orient Display Technical Support (info@orientdisplay.com) if you have any questions.

Technical Support and Updating

1. Technical Support

1.1 Information about our company’s software and hardware

1.2 Problems related to our software and hardware manuals

1.3 After-sale technical support for OEM and ODM

1.4 Requirement of source code and other info which is lost or updated

1.5 Malfunction diagnose and other after-sale services

2. Range of Technical Discussion (non-compulsory)

2.1 Modification and comprehension of source code

2.2 How to port OS

2.3 Software and hardware problems occurred in self-modifying and programming

3. Accesses to Technical Support

3.1 Tel (non-instant messenger):	1-905-477-1166
3.2 Email address (non-instant messenger):	info@orientdisplay.com

4. Timetable for Technical Support

9:00am to 5:30pm, EST, Monday to Friday

Support will not be available on public holidays. Please send your questions to the email addresses above. We’ll reply as soon as we are back.

5. Accesses to Materials

Board related technical files will be provided via a Dropbox download link

Copyright Announcement

Update Record

Date	Version	Update
Feb., 2021	V1.0	first edition
Jun., 2021	V2.0	Second edition

Chapter 1 Software Features Overview

Device driver	Driver code path in kernel	Device name
Wifi	/usr/lib/	8188gtvu.ko

Chapter 2 Hardware Introduction

2.1 PRODUCT OVERVIEW

AIY-A002M belongs to Linux intelligent mainboard, use i.MX6UL ARMCortex-A7 high energy efficiency CPU, It is a cost-effective application processor used in industrial-grade single-board computers for industrial control and communications.

AIY-A002M build-in 512MB DDR3L and 8GB EMMC, Integrated Ethernet, WiFi, NB IOT, USB, CAN, TF card and other functions. Supports RGB1366*768 resolution video output

The power supply voltage of the product is 12V DC. the working temperature of the product is -20~60°C, and the storage temperature is -30~70°C, which can work reliably in different environments.

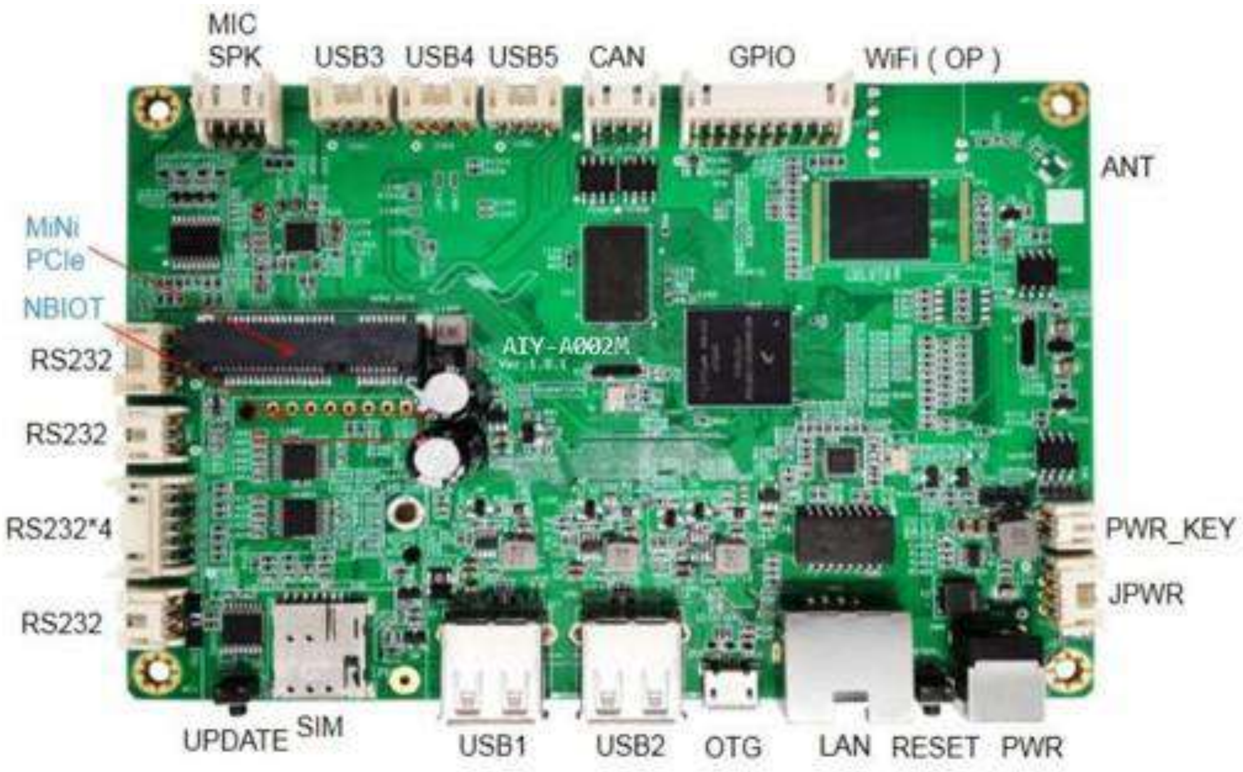
2.2 Hardware parameters

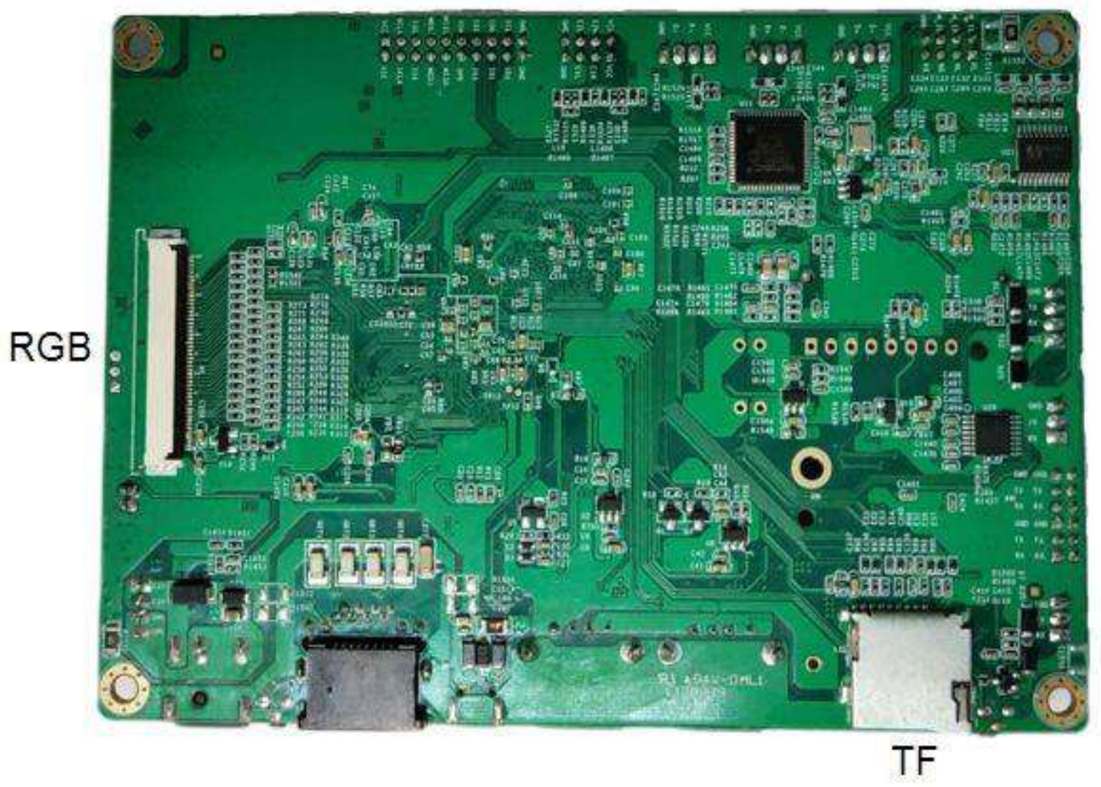
Features: **Diagram:**

Operation system	Android7.1, Linux
CPU	RK3288 ARM quad-core A17, Maximum main frequency 1.6 GHz, Integration Mali-T764 GPU
Memory	2G DDR3+ 8GB EMMC (16G/32G/64GOptional)
Display Interface	EDP*1(4K@30Hz), HDMI*1(1080P*120Hz or 4k@60Hz), MIPI*2(1080@60Hz), LVDS*1(Double 8bit LVDS)
Camera	MIPI
Video Format Support	WMV, AVI, FLV, RM, RMVB, MPRG, TS, PM4
Image Format Support	BMP, JPEG, PNG, GIF
Communication Interface	Gigabit Ethernet, WIFI, BT, RS232*2, RS232/RS485*1, OTG*1, USB*2, CAN*1, I2C*2, MiniPCIE 4G, SIM Card, TF Card slot, GPIO*6
Audio interface	Audio, MIC, SPK (8Ω, 5W)
Power Supply	9-24V DC IN
Power Mode	Standby mode, sleep mode
System upgrade	Support for Local USB Upgrade, OTA, OTG
OSD Language	Multilingualism
Working temperature	-20~60℃
Storage temperature	-30~70℃
Dimensions	154*02 mm

2.3 Interface description

2.3.1 Interface diagram





2.3.2 Pin Definition

1) JPOWER

Define	PIN		Define
12V	1	2	12V
GND	3	4	GND

2) PWR KEY

Define	PIN		Define
GND	1	2	PWR KEY

3) RGB

Define	Pin		Define
VCC	1	26	R3
VCC	2	27	R4
VLCD	3	28	R5
VLCD	4	29	R6
GND	5	30	R7
GND	6	31	GND
B0	7	32	PWM
B1	8	33	RST
B2	9	34	DISP
B3	10	35	INT
B4	11	36	GPIO
B5	12	37	SCL
B6	13	38	SDA
B7	14	39	GND
G0	15	40	DE
G1	16	41	VSYNC
G2	17	42	HSYNC
G3	18	43	PCLK
G4	19	44	GND
G5	20	45	NC
G6	21	46	NC
G7	22	47	NC
R0	23	48	NC
R1	24	49	NC
R2	25	50	GND

4) WIFI

Define	PIN		Define
VCC	1	4	GND
USB_DN	2	5	GND
USB_DP	3	6	ANT

5) Speaker/MIC

Define	PIN		Define
OUTPL	1	2	MIC_IN1L
OUTNL	3	4	MIC_IN1R
OUTNR	5	6	MIC_IN2L
OUTPR	7	8	GND

6) LAN

Define	PIN		Define
TX+	1	8	NC
TX-	2	9	VENET
RX+	3	10	LED2/Nintse
NC	4	11	VENET
NC	5	12	LED1/REGOFF
RX-	6	13	GND
NC	7	14	GND

7) RS232（4PIN, it can power the peripherals）

Define	PI	N	Define
GND	1	3	RX
TX	2	4	VCC

8) RS232（3PIN）

Define	PI N		Define
GND	1	3	RX
TX	2		

9) RS232（COM4）

Define	PIN		Define
GND	1	2	GND
TX	3	4	TX
RX	5	6	RX
GND	7	8	GND
TX	9	10	TX
RX	11	12	RX

10) MINI PCIE（4G）

Define	PIN		Define
WAKE	1	2	VCC_PCIE
NC	3	4	GND
NC	5	6	+1.5V
CLKREQ#	7	8	USIM1_VCC
GND	9	10	USIM1_IO
NC	11	12	USIM1_CLK
NC	13	14	USIM1_RST
GND	15	16	NC
NC	17	18	GND
NC	19	20	W_DISABLE
GND	21	22	PERST
NC	23	24	VCC_PCIE
NC	25	26	GND
GND	27	28	+1.5V
GND	29	30	NC

NC	31	32	NC
NC	33	34	GND
GND	35	36	USB_HUB_6-R
GND	37	38	USB_HUB_6+R
VCC_PCIE	39	40	GND
VCC_PCIE	41	42	LED_WWAN
GND	43	44	LED_WLAN
TP5	45	46	LED_WPAN
TP7	47	48	+1.5V
TP6	49	50	GND
TP8	51	52	3V3

11) USB1/USB2

Define	PIN		Define
VCC	1	3	D+
D-	2	4	GND

12) USB3/USB4/USB5

Define	PIN		Define
VCC	1	3	D+
D-	2	4	GND

13) OTG

Define	PIN		Define
VCC	1	4	ID
DM	2	5	GND
DP	3		

14) CAN*2

Define	PIN		Define
VCC	1	2	VCC

CAN1_H	3	4	CAN2_H
CAN1_L	5	6	CAN2_L
GND	7	8	GND

15) SPI/GPIO/I2C

Define	PIN		Define
SPI_PWR	1	2	SPI_PWR
SPI1_SCLK	3	4	SPI2_SCLK
SPI1_SS0	5	6	SPI2_SS0
SPI1_MOSI	7	8	SPI2_MOSI
SPI1_MISO	9	10	SPI2_MISO
GPIO_4	11	12	GND
GPIO_2	13	14	GPIO_3
GPIO_0	15	16	GPIO_1
I2C_SCL_EXP	17	18	I2C_SDA_EXP
GND	19	20	GND

16) NB-IOT

Define	PIN		Define
VBAT	1	5	RXD
VCC_NB	2	6	PEN
GND	3	7	NC
TXD	4	8	NC

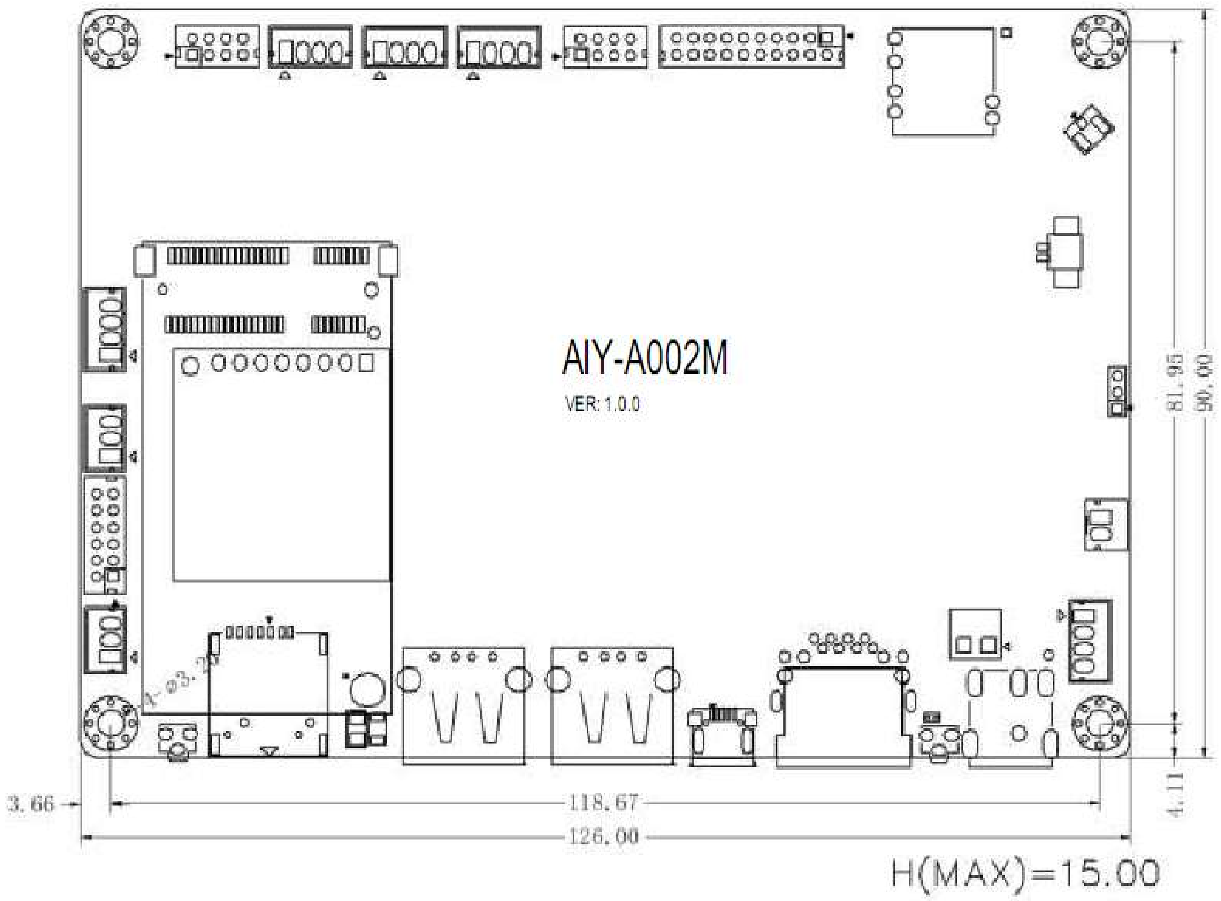
17) TF CARD

Define	PIN		Define
DATA2	1	6	VSS
CD/DATA3	2	7	DATA0
CMD	3	8	DATA1
VDD	4	9	SD
CLK	5	10	GND

18) SIM

Define	PIN		Define
NC	4	2	RST
GND	5	7	I/O
VCC	1	3	CLK
VPP	6	8	GND

2.4 Mechanical structures



2.5 Points for attention

During assembly and use, note the following (and not limited to) problem points.

1) The short circuit problem of bare board and peripheral.
2) In the process of installation and fixing, avoid the deformation problem caused by the fixing of the bare plate.
3) Pay attention to the direction of the first foot when connecting the screen.
4) When the peripheral (UART. etc) is installed, The IO level of the peripheral should match with the motherboard
5) Make sure RX/TX is properly connected when installing serial ports
6) Verify that the input voltage is correct before power is connected.

Chapter 3 Board Environment Setup

3.1 VMware Workstation Installing

3.1.1 Before starting

Before downloading and installing VMware Workstation. Ensure that your physical machine meets the system requirements for VMware Workstation. For more information, see *System Requirements*:

- **VMware Workstation 16 Pro Tech Specs**

3.1.2 Downloading VMware Workstation

To download VMware Workstation:

1. Navigate to the **VMware Workstation Download Center**.
2. Based on your requirements, click **Go to Downloads** for VMware Workstation for Windows or VMware Workstation for Linux.
3. Click **Download Now**.
4. If prompted, log in to your My VMware profile. If you do not have a profile, create one.
5. Review the End User License Agreement and click **Yes**.
6. Click **Download Now**.

If the installer fails to download during the download process:

- Delete the cache in your web browser. For more information, see:
  - Mozilla Firefox: **How to clear the Firefox cache**
  - Google Chrome: **Delete your cache and other browser data**
  - Microsoft Internet Explorer: **How to delete the contents of the Temporary Internet Files folder**
- Disable the pop-up blocker in your web browser. For more information, see:
  - Mozilla Firefox: **How do I disable a Pop-up blocker?**
  - Google Chrome: **Manage pop-ups**
  - Microsoft Internet Explorer: **How to turn Internet Explorer Pop-up Blocker on or off on a Windows XP SP2-based computer**
- Download using a different web browser application.
- Disable any local firewall software.
- Restart the virtual machine.
- Download the installer from a different computer or network.

3.1.3 Installing VMware Workstation

Notes:

You must have only one VMware Workstation installed at a time. You must uninstall previous version of VMware Workstation before installing a new version.

To install VMware Workstation on a Windows host:

1. Log in to the Windows host system as the Administrator user or as a user who is a member of the local Administrators group.
2. Open the folder where the VMware Workstation installer was downloaded. The default location is the **Downloads** folder for the user account on the Windows host.
3. Right-click the installer and click **Run as Administrator**.
4. Select a setup option:
    - **Typical**: Installs typical Workstation features. If the Integrated Virtual Debugger for Visual Studio or Eclipse is present on the host system, the associated Workstation plug-ins are installed.
    - **Custom**: Lets you select which Workstation features to install and specify where to install them. Select this option if you need to change the shared virtual machines directory, modify the VMware Workstation Server port, or install the enhanced virtual keyboard driver. The enhanced virtual keyboard driver provides better handling of international keyboards and keyboards that have extra keys.
5. Follow the on-screen instructions to finish the installation.
6. Restart the host machine.

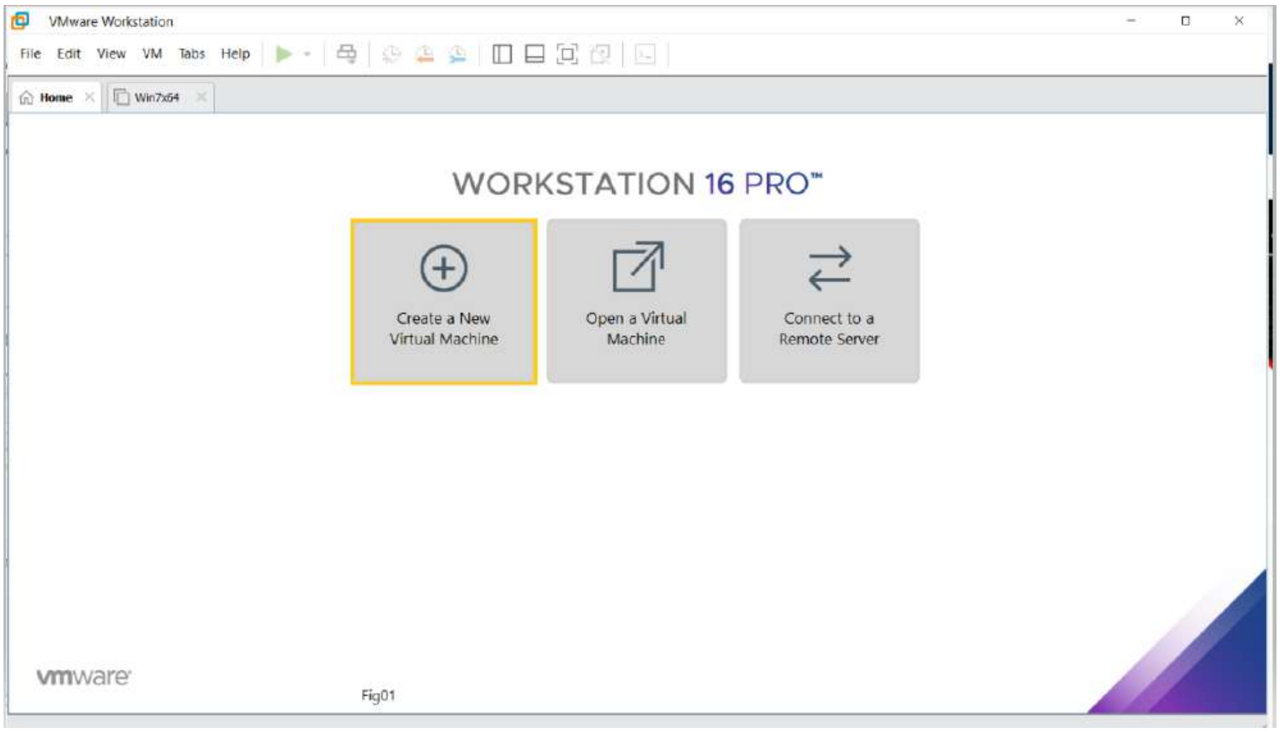
3.2 Ubuntu Installing

3.2.1 Download Ubuntu

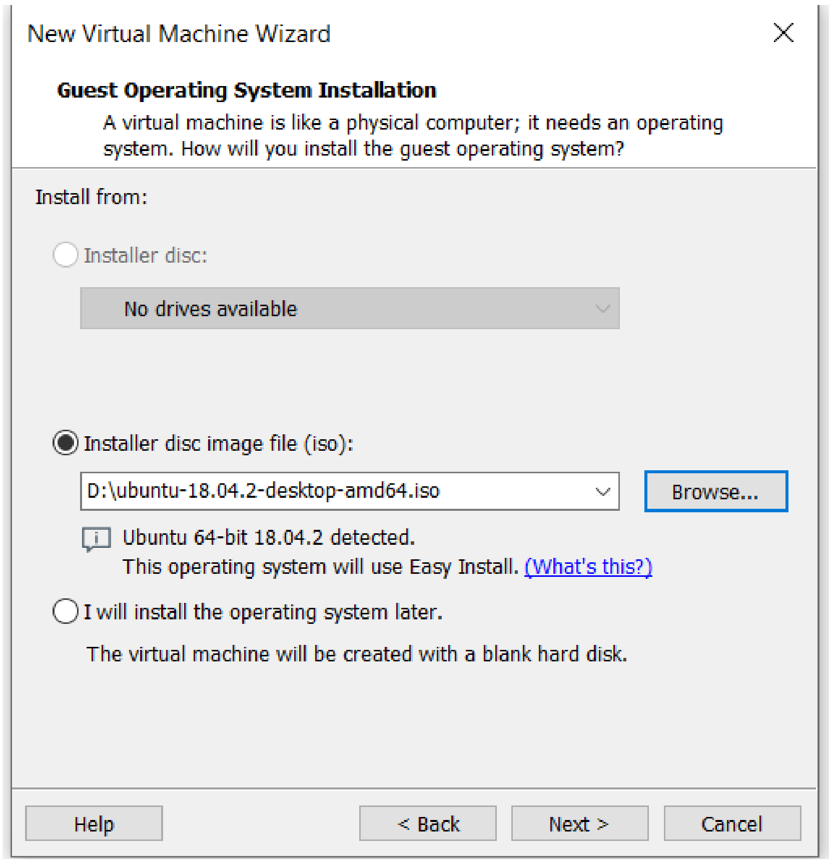
Open the **Official Page** to download the Ubuntu 18.04 LTS Desktop

3.2.2 Create the Virtual Machine

In this step, we will create the Virtual Machine used to install Ubuntu On VMware Workstation Player. Launch the VMware Workstation Player if it's not running. It will show the Welcome Screen as shown in Fig 01.


VMware Workstation
File Edit View VM Tabs Help
Home
Win7x64
WORKSTATION 16 PRO™
Create a New Virtual Machine
Open a Virtual Machine
Connect to a Remote Server
vmware
Fig01

Click the **Create a New Virtual Machine Button** as shown in Fig 2.



New Virtual Machine Wizard

Guest Operating System Installation

A virtual machine is like a physical computer; it needs an operating system. How will you install the guest operating system?

Install from:

( ) Installer disc:

No drives available

(●) Installer disc image file (iso):

D:\ubuntu-18.04.2-desktop-amd64.iso

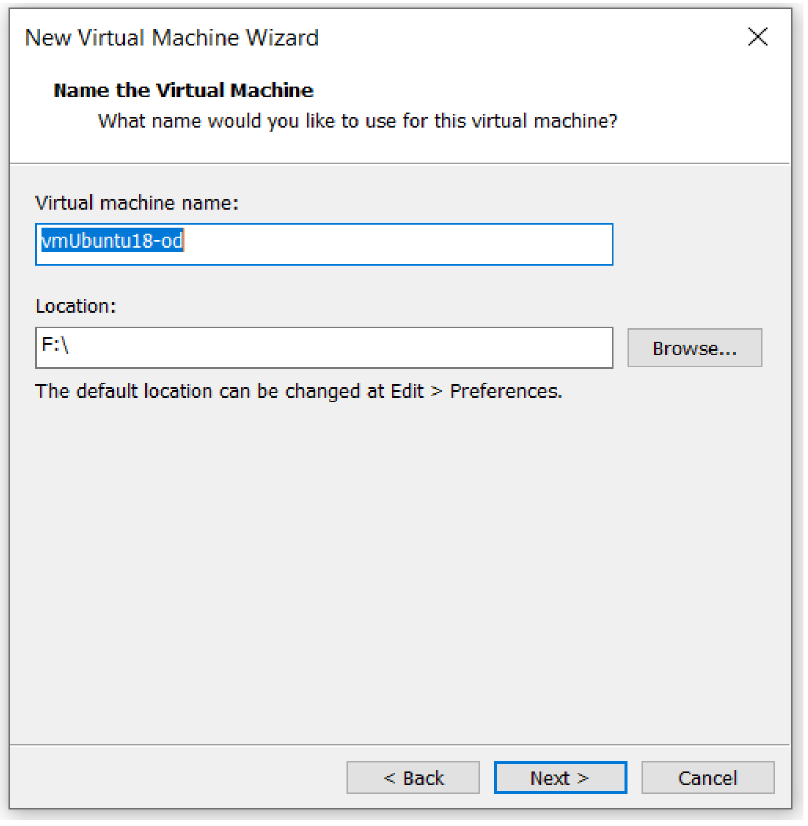
Ubuntu 64-bit 18.04.2 detected.
This operating system will use Easy Install. (What's this?)

( ) I will install the operating system later.

The virtual machine will be created with a blank hard disk.

Fig2

Click the **Next Button** to choose the Operating System as shown in Fig 3



New Virtual Machine Wizard

Name the Virtual Machine

What name would you like to use for this virtual machine?

Virtual machine name:

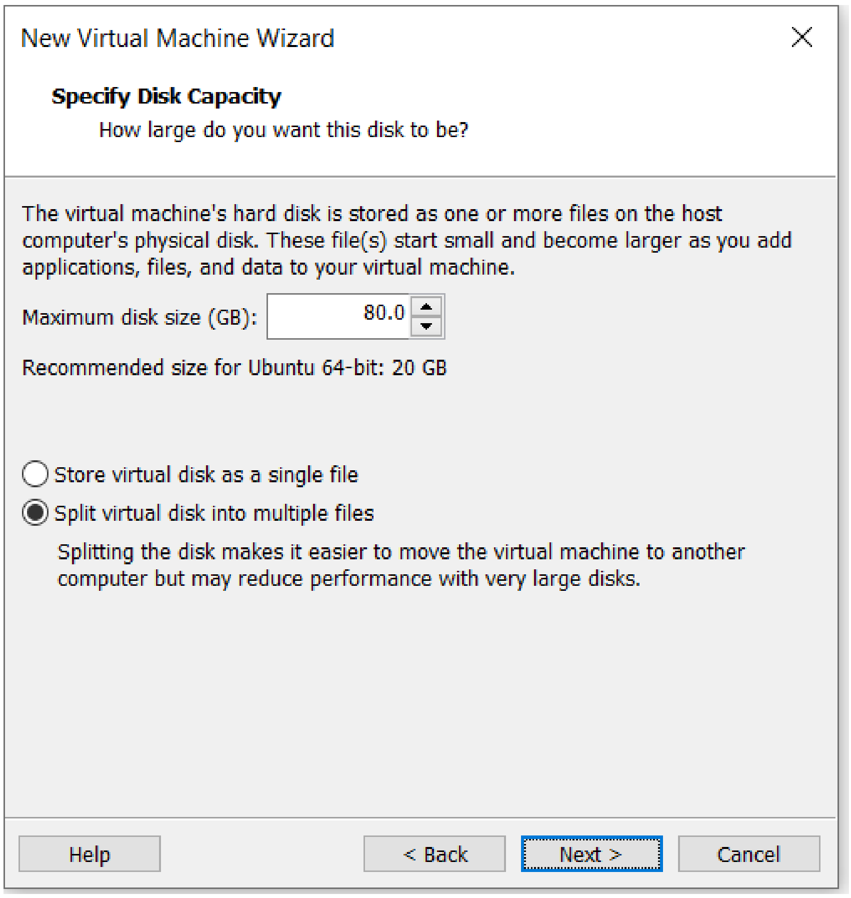
vmUbuntu18-od

Location:

F:\

The default location can be changed at Edit > Preferences.

configure the disk and allocate space according your own system


New Virtual Machine Wizard
Specify Disk Capacity
How large do you want this disk to be?

The virtual machine's hard disk is stored as one or more files on the host computer's physical disk. These file(s) start small and become larger as you add applications, files, and data to your virtual machine.

Maximum disk size (GB): 80.0

Recommended size for Ubuntu 64-bit: 20 GB

( ) Store virtual disk as a single file

(●) Split virtual disk into multiple files

Splitting the disk makes it easier to move the virtual machine to another computer but may reduce performance with very large disks.

3.2.3 Install Ubuntu

3.2.4 Install VMware Tools

The most important feature of VMware Tools is Enhanced Graphics. The issue is where Ubuntu does not occupy the full screen. The other features include sharing files and time synchronization between the guest and host operating systems. We can even copy and paste files by using drag and drop.

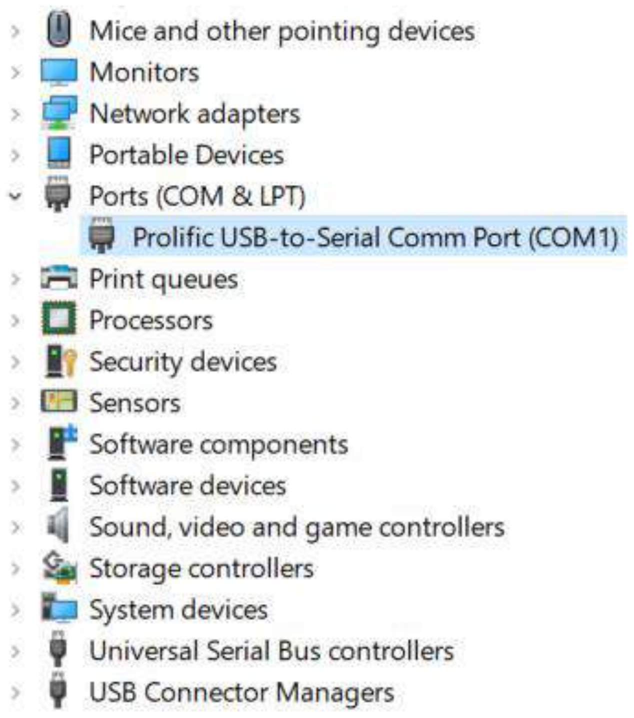
```
# Refresh the packages list
sudo apt update
# Install VMware Tools - Desktop
sudo apt install open-vm-tools-desktop
```

Chapter 4 Basic Operation

This chapter describes the basic operations of the AIY-A002M motherboard, such as hardware connection, booting, and system settings. This chapter is a description of some operations, and they are also common operations in the embedded Linux development process, which need to be mastered

4.1 Install UART(Debug) Driver

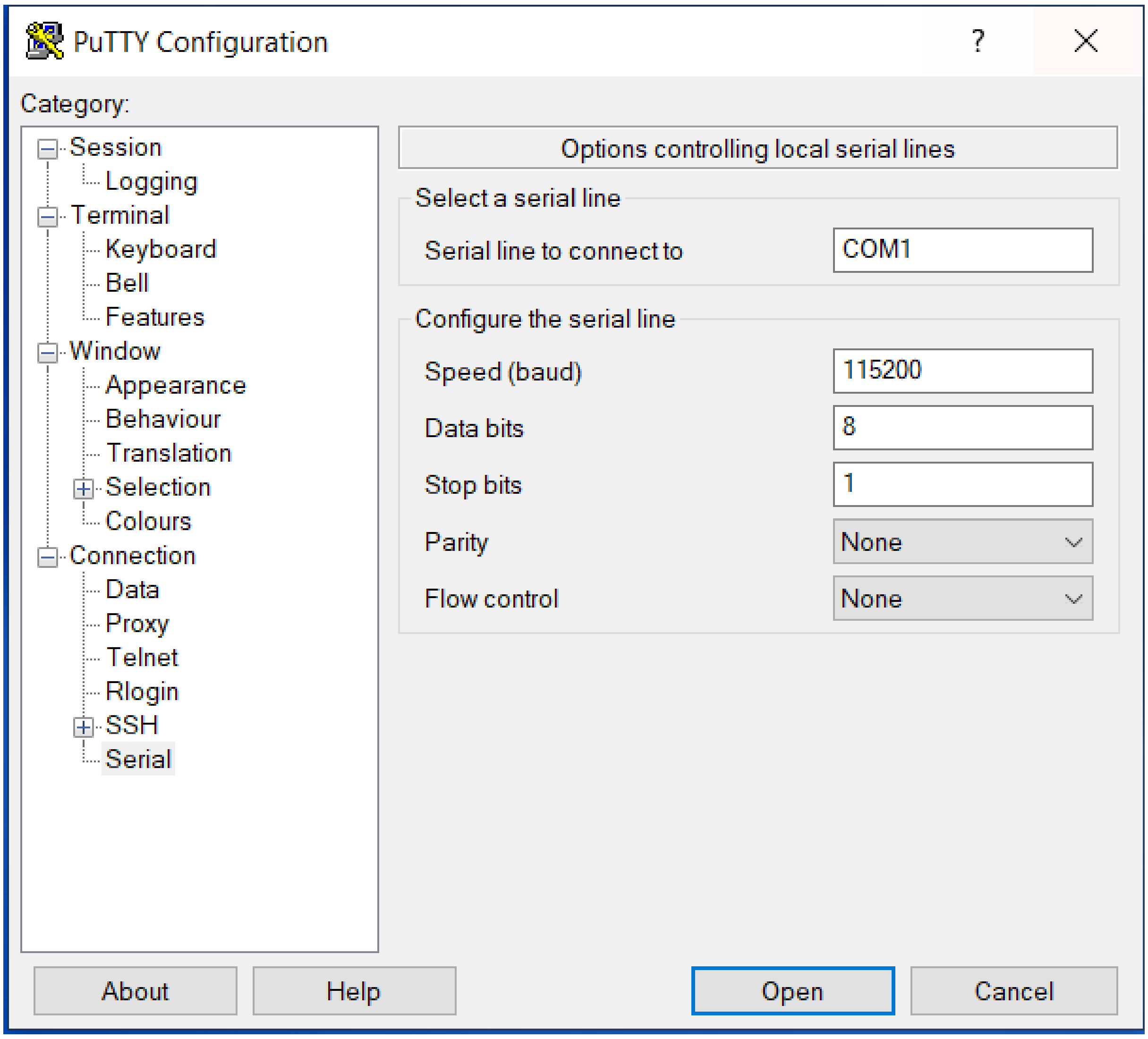
AIY-A002M has a debugging serial port (COM1), which uses RS-232 level. If the debugging computer is equipped with a standard serial port (usually RS-232 level), you can use a serial port extension cable to connect the computer with AIY-A002M. If you are using a portable computer (usually without a standard serial port) or a desktop computer that does not provide a standard serial port, you need a USB to RS-232 serial cable to connect the computer and the motherboard. When using a USB to RS-232 serial cable on a computer running Windows operating system, you need to install the corresponding driver (provided by the manufacturer of the conversion chip).



4.2 Hyper terminal Settings

4.2.1 Connect target board with PC by a micro usb cable and power on the board

4.2.2 Set hyper terminal as below:


PuTTY Configuration

Category:

Options controlling local serial lines

Select a serial line

Serial line to connect to	COM1

Configure the serial line

Speed (baud)	115200
Data bits	8
Stop bits	1
Parity	None
Flow control	None

notice: system log in user name: root, no pass code

4.3 Power on and log

After power supply is connected, the AIY-A002M motherboard runs automatically. After the PC is connected to the motherboard through COM1 (the baud rate is 115200), you can log in to the motherboard through the serial port. The login password and account are both root.

4.4 Shut down and restart

When you need to shut down or restart, if there is a data storage operation, in order to ensure that the data is completely written, you can enter the sync command. After completing the data synchronization, turn off the power and press the reset button to restart. You can also enter the reboot command to restart:

```
[root@AIY-002M ~] # reboot
```

This command will automatically complete the data synchronization and restart the system.

4.5 Install basic software

```
$ sudo apt update
$ sudo apt install net-tools
```

Install SSH server:

```
$ sudo apt install openssh-server
$ sudo systemctl status ssh
```

If not running enable the ssh server and start it as follows:

```
$ sudo systemctl enable ssh
$ sudo systemctl start ssh
```

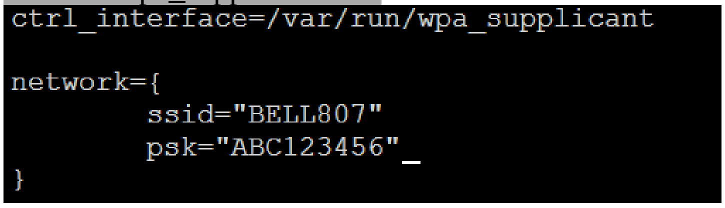
4.6 Connect WIFI

* Load the wifi driver

```
insmod /usr/lib/8188gtvu.ko
```

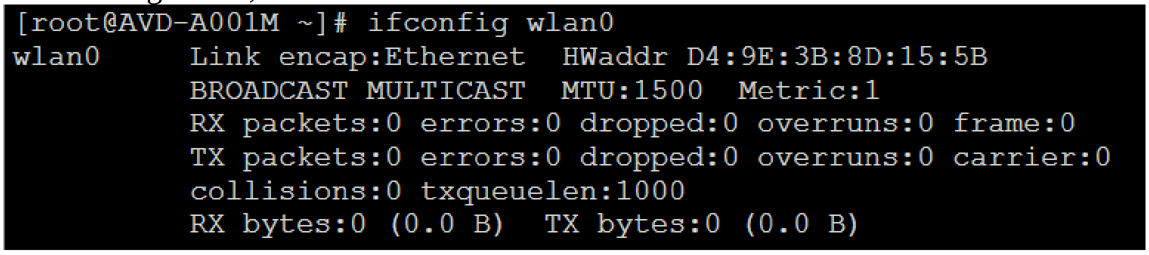
* Configure the ap name and password to /etc/wpa_supplicant.conf

```
# vi /etc/wpa_supplicant.conf
```



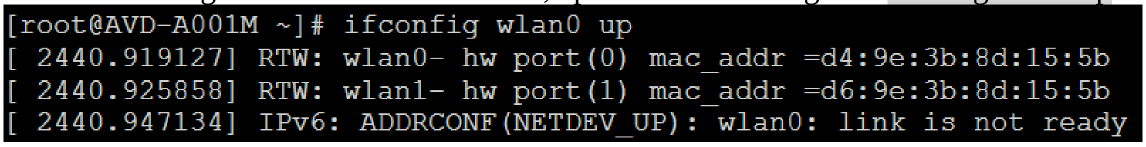
```
ctrl_interface=/var/run/wpa_supplicant

network={
        ssid="BELL807"
        psk="ABC123456"_
}
```

* Use ifconfig wlan0, check if wifi device is exist?



```
[root@AVD-A001M ~]# ifconfig wlan0
wlan0     Link encap:Ethernet  HWaddr D4:9E:3B:8D:15:5B
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

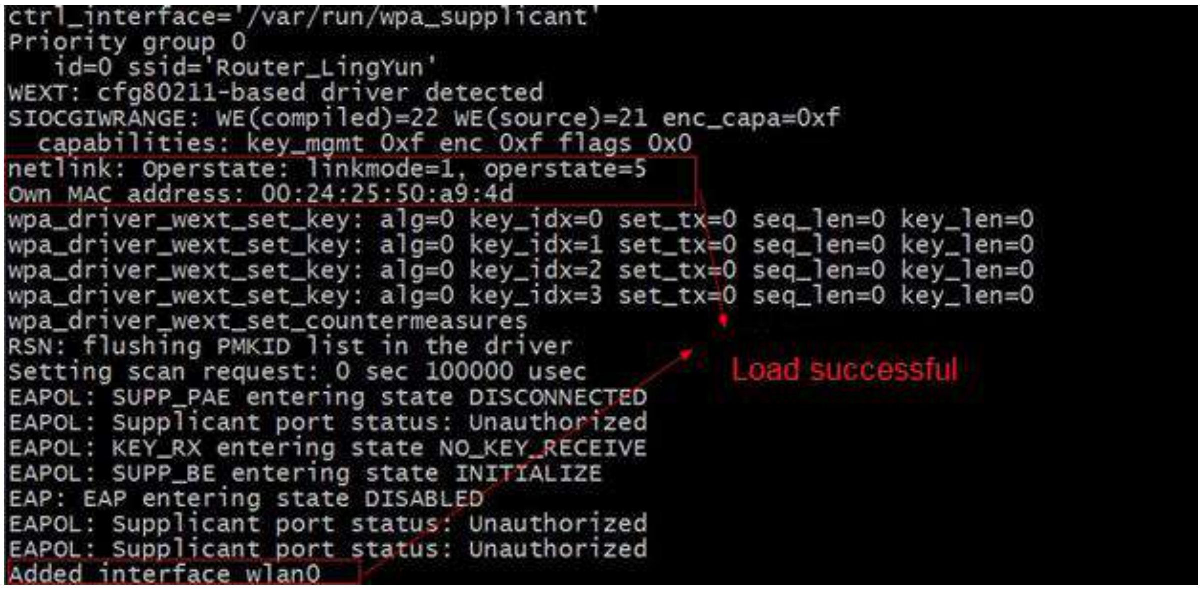
* After confirming that the wifi device exists, open the wifi through the ifconfig wlan0 up command



```
[root@AVD-A001M ~]# ifconfig wlan0 up
[ 2440.919127] RTW: wlan0- hw port(0) mac_addr =d4:9e:3b:8d:15:5b
[ 2440.925858] RTW: wlan1- hw port(1) mac_addr =d6:9e:3b:8d:15:5b
[ 2440.947134] IPv6: ADDRCONF(NETDEV_UP): wlan0: link is not ready
```

* Run: `wpa_supplicant -d -Dwext -iwlan0 -c/etc/wpa_supplicant.conf &`

* Check the print information, if the following information is displayed, it indicates that the connection to the ap is successful

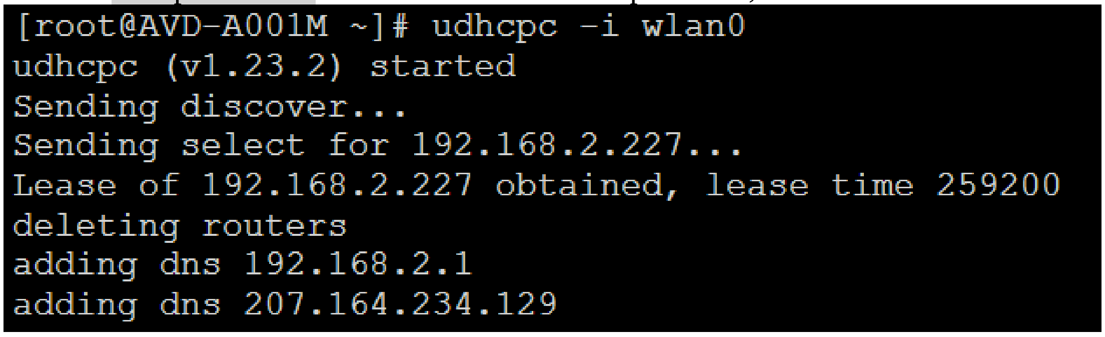


```
ctrl_interface='/var/run/wpa_supplicant'
Priority group 0
   id=0 ssid='Router_LingYun'
WEXT: cfg80211-based driver detected
SIOCGIWRANGE: WE(compiled)=22 WE(source)=21 enc_capa=0xf
  capabilities: key_mgmt 0xf enc 0xf flags 0x0
netlink: Operstate: linkmode=1, operstate=5
Own MAC address: 00:24:25:50:a9:4d
wpa_driver_wext_set_key: alg=0 key_idx=0 set_tx=0 seq_len=0 key_len=0
wpa_driver_wext_set_key: alg=0 key_idx=1 set_tx=0 seq_len=0 key_len=0
wpa_driver_wext_set_key: alg=0 key_idx=2 set_tx=0 seq_len=0 key_len=0
wpa_driver_wext_set_key: alg=0 key_idx=3 set_tx=0 seq_len=0 key_len=0
wpa_driver_wext_set_countermeasures
RSN: flushing PMKID list in the driver
Setting scan request: 0 sec 100000 usec
EAPOL: SUPP_PAE entering state DISCONNECTED
EAPOL: Supplicant port status: Unauthorized
EAPOL: KEY_RX entering state NO_KEY_RECEIVE
EAPOL: SUPP_BE entering state INITIALIZE
EAP: EAP entering state DISABLED
EAPOL: Supplicant port status: Unauthorized
EAPOL: Supplicant port status: Unauthorized
Added interface wlan0
```

* Configure the wireless network card IP and subnet mask:

```
# ifconfig wlan0 192.168.1.244 broadcast 192.168.1.1 netmask 255.255.255.0 up
```

* Use the udhcpc -i wlan0 command to obtain the ip address, and the network can be used normally



```
[root@AVD-A001M ~]# udhcpc -i wlan0
udhcpc (v1.23.2) started
Sending discover...
Sending select for 192.168.2.227...
Lease of 192.168.2.227 obtained, lease time 259200
deleting routers
adding dns 192.168.2.1
adding dns 207.164.234.129
```

* Test

```
# ifconfig eth0 donw
# ifconfig
# ping 192.168.2.1
```

4.7 Set up automatic start up

4.7.1 Boot script

The system file: /etc/init.d/S90start_userapp.sh is a script that is automatically executed when booting. The content of the script is shown in code listing 4.1. Commands or applications that need to be automatically executed at boot can be added to this file.

Code list 4.1 start_userapp file content

```
#!/bin/sh
Ifconfig eth0 192.168.1.2
# you can add your app start_command here
```

4.7.2 Add auto-execute command on boot

If you want to automatically execute a program when you turn on the machine, such as the mydemo program in the /root directory, add a command to execute the /root/mydemo program in the S90start_userapp.sh file:

```
#!/bin/sh
ifconfig eth0 192.168.1.2
# you can add your app start command here
/root/mydemo
```

4.6 Use TF card

After inserting the TF card into the TF slot of the motherboard, the Linux system will automatically detect the TF card and print the relevant kernel information:

```
[root@AIY-A002M    ~] #
mmc0: new high speed SD card at address 21ed
mmcblk0: mmc0:21ed APPSD 121 MiB
mmcblk0: p1
```

The system will generate a directory under /media for each partition of the TF card. The name of the directory is mmcblk0pn (n represents a different partition, n=0, 1, 2, 3...). Each partition of the TF card is automatically mounted in these directories. The files saved by the user in these directories will be stored in the corresponding partition of the TF card.

Enter the df command to view the mounting status and usage of each partition:

```
[root@AIY-A002M ~]# df
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/root 6846632 58888 6436624 1% /
devtmpfs 866940 0 866940 0% /dev
tmpfs 1031292 0 1031292 0% /dev/shm
tmpfs 1031292 112 1031180 0% /tmp
 tmpfs 1031292 20 1031272 0% /run
/dev/sda1 30267776 83600 30184176
```

Before the TF card is used up and ejected, you need to uninstall all the partitions first

```
[root@AIY-A002M ~]#umount /dev/mmcblk0p1
```

Note that before unmounting a partition, you must first move the current directory out of the directory where the partition is mounted, that is to say, you cannot unmount /media/mmcblk0p1 under the /media/mmcblk0p1 directory.

4.7 Use USB Disk

AIY-A002M integrates two universal USB ports, which can directly support U disk connection. After inserting the USB flash drive into the USB interface, Linux will automatically detect the connection of the USB flash drive and print out the information:

```
root@AIY-002M ~]# usb 1-1.2: new high-speed USB device number 4 using ci_hdrc
usb-storage 1-1.2:1.0: USB Mass Storage device detected
scsi host1: usb-storage 1-1.2:1.0
scsi 1:0:0:0: Direct-Access SanDisk Cruzer Blade 1.26 PQ: 0 ANSI: 6
sd 1:0:0:0: [sda] 15633408 512-byte logical blocks: (8.00 GB/7.45 GiB)
sd 1:0:0:0: [sda] Write Protect is off
sd 1:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
sda: sda4
sd 1:0:0:0: [sda] Attached SCSI removable disk
```

The system will generate a directory for each partition of the U disk under the /media directory, the name of the directory is sdxn (x is used to distinguish different U disks, n is used to distinguish different partitions, x=a, b, c... … N=0, 1, 2, 3…), each partition of the U disk is mounted under these directories. The files saved by the user in these directories will be stored in the corresponding partition of the U disk.

Enter the df command to view the mounting status and usage of each partition of the U disk:

```
[root@AIY-002M ~]# df
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/root 6617864 94472 6180560 2% /
devtmpfs 89312 0 89312 0%
/dev tmpfs 253376 0 253376 0% /dev/shm
tmpfs 253376 104 253272 0% /tmp
tmpfs 253376 172 253204 0% /run
/dev/mmcblk1p1 511720 6592 505128 1% /media/mmcblk1p1
/dev/sda4 7801088 4076864 3724224 52% /media/sda4
```

Similar to the TF card, the U disk should be unloaded before unplugging the relevant partition before using it.

Chapter 5
Setup Embedded Linux Development Environment

This chapter first describes the basic methods of embedded Linux development in the Linux environment, and then introduces the software used in embedded development, including how to install and test. This chapter is indispensable for embedded Linux development and is the basis for embedded Linux development. Please understand it carefully and make correct settings.

5.1 Install cross compiler

The cross compiler is usually released under the name arm-none-linux-gnueabi.tar.bz2 (the tool chain names of different manufacturers and different platforms are mostly different and generally not universal), For AIY-A002M motherboard, the name of the cross-compilation tool chain is gcc-linaro-arm-linux-gnueabihf-4.9-2014.09_linux.tar.bz2.

5.1.1 Unzip the cross compiler

The developer copies the cross-compiler to the development host (Ubuntu is recommended as the development host), and unzip it by referring to the following command:

```
od@Linux-host:~$tar -jxvf gcc-linaro-arm-linux-gnueabihf-4.9-2014.09_linux.tar.bz2 -C /opt
```

5.1.2 Set environment variables

There are many ways to set system environment variables, the two commonly used are described below:

5.1.2.1 Modify the global configuration

/etc/profile is the global configuration file of the system. Set the path of the cross-compiler in this file, so that all users who log in to the machine can use this compiler

Open the terminal, enter the "sudo vi /etc/profile" command to open the /etc/profile file, and add at the end of the file:

```
export PATH=/opt/gcc-linaro-arm-linux-gnueabihf-4.9-2014.09_linux/bin:$PATH
```

Save the file and exit, and then enter ". /etc/profile" (dot + space + file name) in the terminal, and execute the profile file to make the changes just made effective. If there are no input errors, reopen the terminal at this time, enter arm-linux-gnueabihf-, and press the TAB key on the keyboard, you can see many commands starting with arm-linux-gnueabihf-.

5.1.2.2 Modify user profile (recommended)

"/etc/profile" is a global configuration file that will affect all users who log in to this machine. If you don’t want your personal settings to affect other users of the system, you can modify the configuration file that only belongs to the current user, usually "~/.bashrc" or “~/.bash_profile”

The modification method is similar to modifying the "/etc/profile" file. After the file is opened, add at the end:

```
export PATH=/opt/gcc-linaro-arm-linux-gnueabihf-4.9-2014.09_linux/bin:$PATH
```

In the same way as executing "/etc/profile", enter ". .bashrc" or ". .bash_profile" to execute the modified file to make the modification effective. If it is correct, reopen the terminal, enter arm-linux-gnueabihf-, and then press the keyboard TAB key, you can also see many commands beginning with arm-linux-gnueabihf-

5.1.2.3 System dependency package

The following dependency packages are required in the development environment, which can be installed directly through the apt tool

```
$ sudo apt-get install lib32z1 lib32z1-dev
$ sudo apt-get install lib32stdc++6
$ sudo apt-get install g++
```

5.1.2.4 Test toolchain

Open the terminal and run the cross-compiler tool. If you can get an output similar to the following, the cross-compiler has been able to work normally

```
od@linux-host:~$arm-linux-gnueabihf-gcc
arm-linux-gnueabihf-gcc:fatal error: no imput files
compilation terminated
```

Further, you can also write a simple c file, and then check whether the cross tool chain can successfully compile it.

Chapter 6 Hardware interface programming

This chapter mainly talks about hardware interface programming. Since users need secondary development in AIY-A002M, and often operate on the usual serial ports and GPIOs, this chapter focuses on the description of the programming and usage methods of this part of the hardware contact.

6.1 GPIO Hardware programming

This motherboard system provides a very convenient operation interface for GPIO. The GPIO sub-directory of AIY-A002M is /sys/class/gpio. First introduce how to operate GPIO in the onboard Linux system (please refer to the motherboard specification for the specific location and connection of GPIO)

6.1.1 GPIO interface

/sys/class/gpio has following file nodes:

```
[root@AIY-A002M     ~]# ls /sys/class/gpio/
export      gpiochip0@   gpiochip32@     gpiochip96@
gpio132@     gpiochip128@   gpiochip64@    unexport
```

Export and unexport are the attribute files of the GPIO subsystem. Export is used to export gpio. After gpio is exported, related nodes will appear in the /sys/class/gpio/ directory. Operating related nodes can realize the direction and level of gpio. Read and write status, etc.

Write the sequence number N of the GPIO to the export file to export its device catalog. The calculation formula of the sequence number is as follows:

GPIO sequence number = (BANK − 1) × 32 + N

In the formula, BANK is the BANK where the GPIO pin is located, and N is the serial number of the pin in that BANK. Take IO4 on the motherboard (the actual GPIO pin is GPIO1_IO04) as an example, its BANK value is 1, N value is 4, so the sequence number is (1-1)*32+4=4.

The operation command to write the sequence number is as follows (note that there is a space on each side of the ">"):

```
[root@AIY-A002M ~]# echo 4 > /sys/class/gpio/export
```

After the above command is executed, the gpio4 directory will be generated under the /sys/class/gpio. The purpose of operating this GPIO can be achieved by reading and writing the device files in this directory. Similary, you can also export other GPIO device. After the device directory of GPIO4 is generated, you can see that it contains the following property files:

```
[root@AIY-A002M ~]# ls /sys/class/gpio/gpio4/
active_low   direction   power   uevent
device   edge   subsystem   value
```

Among them, the commonly used ones are the direction and value, the direction is used to configure gpio as input or output, and the value is used for output replacement (when used as output) or read replacement (when used as input)

6.1.2 Use GPIO from the command line

After GPIO is exported, it defaults to input function. You can view the current operating direction of the GPIO by reading the direction file

```
[root@AIY-A002M     ~]# cat /sys/class/gpio/gpio4/direction
in
```

Write the "out" string to the direction file to set GPIO as output:

```
[root@AIY-A002M ~]# echo out>  /sys/class/gpio/gpio4/direction
```

In the same way, you can write an "in" string to set GPIO back to input.

When GPIO is set as input, the value file records the input level status of the GPIO pin: 1 means input is a high level; 0 means that the input is a low level. You can read the input level of GPIO by viewing the value file

```
[root@AIY-A002M ~]# echo in >
/sys/class/gpio/gpio4/direction
[root@AIY-A003L ~]# cat /sys/class/gpio/gpio4/value
0
```

When the GPIO is set to output, the state of the output level can be controlled by writing 0 or 1 to the value file (0 means output low level, 1 means output high level):

```
[root@AIY-A002M ~]# echo out >/sys/class/gpio/gpio4/direction
[root@AIY-A002M ~]# echo 1 > /sys/class/gpio/gpio4/value
[root@AIY-A002M ~]# echo 0 > /sys/class/gpio/gpio4/value
```

6.1.3 Use GPIO by writing C program

The overall operation is the same as the command line principle, and related operations are also performed by reading and writing file nodes. When using system calls to implement GPIO input and output operations, you also need to export GPIO through the export property file first.

```
#define EXPORT _PATH “sys/class/gpio/export”
#define GPIO “4”                                          // GPIO serial number
Int fd_export = open(EXPORT_PATH, O_RDWR);      // open export file
…
Write(fd_export,GPIO,strlen(GPIO));
```

Then call write function to write an in/out string to the direction device file, and set GPIO as input Or output:

```
#define DIRECT_PATH  “sys/class/gpio/gpio4/direction”
Int fd_dir, ret;
fd_dir = open(DIRECT_PATH, O_RDWR);
…
ret = write(fd_dir,direction_IN, SIZERF(DIRECTION));
```

Finally read or write the value to complete the final operation

```
#define DEV_PATH  “sys/class/gpio/gpio4/value”
Int fd_dev, ret;
fd_dev = open(DEV_PATH, O_RDWR);
…
ret = read(fd_dev,buf, sizeof(buf));
```

After coding, a binary file that can be run on the motherboard is generated through cross-compilation:

```
od@Od-System-Builder:~$ arm-linux-gnueabihf-gcc gpio_test.c -o gpio_test
```

After compiling, copy the binary file to the USB flash drive (assuming to copy to the root directory of the USB flash drive), insert the USB flash drive, and run the binary file.

```
[root@AIY-A002M ~]# /media/sda4/gpio_test
```

If you test the output, you can directly measure the gpio level to confirm whether the program is valid. If you test the input, you can directly connect the GPIO to GND or VCC (3.3v) on the motherboard with a DuPont cable, and then read and print the level through the program.

6.2 Serial programming

Like most other devices, the serial port in Linux appears as a device file. AIY-A002M motherboard serial device file is /dev/ttymxcn (n=0~2, 4~7), there are 7 serial ports in total.

6.2.1 open serial port

Before using a serial port, you must use the open function to open its corresponding device node file. For example, the code to open "/dev/ttymxc0" is shown in code listing 6.1.

Code Listing 6.1 Open the serial port device

```
#define TTY_MXC0_PATH “/dev/ttymxc0”
int fd:
fd = open(TTY_MXC0_PATH, O_RDWR | O_NOCTTY);
If(fd<0){
      printf(“open uart device ttymxc0 error\n”);
}
```

When the open call succeeds, it will return the file descriptor as a parameter of other operation functions; if it fails, it will return a negative number. When opening a serial port, in addition to the O_RDWR option flag, it is usually necessary to use O_NOCTTY, which means that the opened is a terminal device and the program will not become the controlling terminal of the port. If this flag is not used, an input of the task (such as keyboard termination signal, etc.) may affect the process.

6.2.2 Set the serial port baud rate

After opening the serial port, the serial port uses the default baud rate of 9600. In practical applications, since the baud rate of other communication terminals is not 9600, it is often necessary to set the baud rate.

Before setting the serial port, first read the serial port parameters, and then modify the baud rate: get and set terminal properties:

```
tcgetattr( fd,&options)；     // options is termios structure
```

The baud rate of the serial port is divided into input baud rate and output baud rate, respectively, through cfsetispeed() and cfsetospeed() function setting (options is the serial port attribute structure obtained through tcgetattr above).

```
Cfsetispeed(&options, speed);
Cfsetospeed(&options, speed);
```

6.2.3 Read and write data

Use the read/write function to read and write data on the serial port, that is, read the data received by the serial port, or send data from the serial port:

```
#define DEV_NAME "/dev/ttymxc1"
```

```
fd = open(DEV_NAME, O_RDWR | O_NOCTTY); ...
len = write(fd, buf, sizeof(buf));            /* Write a string to the serial port */
...
len = read(fd, buf, sizeof(buf));             /* Read string from mouth */
```

6.2.4 Close serial port

After using the serial port, use the close() function to close the serial port (parameter fd is the file descriptor obtained when the serial port is opened):

```
close(fd);
```

Chapter 7 Embedded GUI (QT) programming

QT is a common Linux graphical interface, Qt/Embedded is the embedded version of QT. This chapter introduces the basic programming of embedded Qt, starting from the environment construction, and introduces the qmake tool and Qt Creator.

7.1 Qt/Embedded Introduction

Qt is a cross-platform application and UI development framework. With Qt, you only need to develop applications once, and you can deploy these applications across different desktops and embedded operating systems without rewriting the source code. Qt on the embedded Linux distribution belongs to the Embedded Linux branch platform of Qt (referred to as Qt/E in this article). Based on the original Qt, Qt/E has made many excellent adjustments to suit the embedded environment. AIY-A002M uses the more mature qt4.8.6, and has integrated the relevant operating environment in the motherboard file system. The developer can run the compiled qt program directly on the motherboard.

7.2 Setup Qt/Embedded cross-compilation environment

7.2.1 Introduction to Compilation Environment

Host system：	Ubuntu 18.04
Cross compilation tool：	arm-linux-gnueabihf
Target board：	AIY-A002M

7.2.2 Install tslib

tslib is an open-source program that can provide functions such as filtering, de-jittering, calibration and other functions for the samples obtained by the touch screen driver. It is usually used as the adaptation layer of the touch screen driver and provides a unified interface for the upper application.

7.2.2.1 Before Install

Install autoconf、automake、autoreconf and libtool：

```
od@linux-host:~$sudo apt-get install autoconf automake autoreconf libtool
```

7.2.2.2 Compile and install tslib

Copy tslib-master.zip to ubuntu, unzip the tslib source package:

```
od@Linux-host:~$unzip tslib-master.zip
```

Enter the tslib source directory and configure tslib:

```
od@linux-host:~$cd tslib-master
od@linux-host:~/tslib-master$ ./autogen.sh
od@linux-host:~/tslib-master$ ./configure  --prefix=/opt/tslib  --host=arm-linux-gnueabihf
ac_cv_func_malloc_0_nonnull=yes
```

Among them, --prefix specifies the installation path of tslib, and users can also specify other directories by themselves; and --host specifies the cross-compiler.

Compile tslib source code:

```
od@Linux-host:~/tslib-master$make
```

Install tslib：

```
od@linux-host:~/tslib-master $ sudo chmod 777 /opt
od@linux-host:~/tslib-master $ make install
```

7.2.3 Cross compile Qt 4.8.6

7.2.3.1 Unzip the QT source code package

Copy the source code package of Qt 4.8.6 (qt-everywhere-opensource-src-4.8.6.tar.gz) the user directory of Ubuntu, then execute the decompression command:

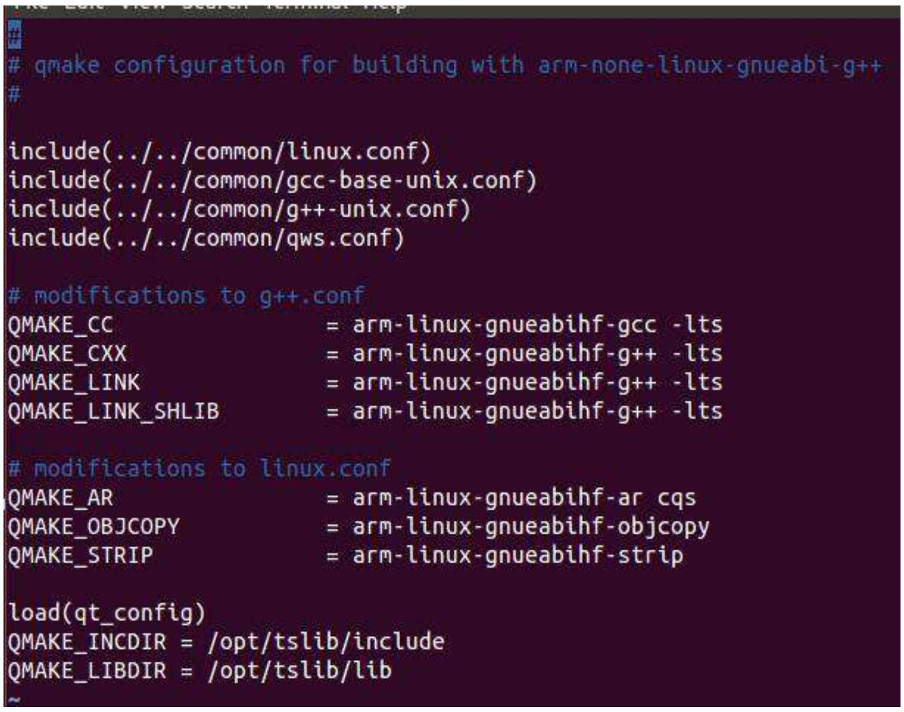
```
od@Linux-host:~/$tar zxvf qt-everywhere-opensource-src-4.8.6.tar.gz
```

After decompression, you will get the qt-everywhere-opensource-src-4.8.6 directory. Enter the directory and you will see a build-qt script. You can adjust the installation directory by editing the parameters in the script (the directory after the -prefix parameter is the installation directory. For example, it can be modified to /opt/qt4.8.6) and so on.

After adjusting build-qt, you also need to edit mkspec/qws/linux-arm-gnueabi-g++/qmake.conf file, modify the tool chain to arm-linux-gnueabihf, and in
Add the -lts parameter to g++.conf, and finally add the following two lines of parameters at the end of the file:

```
QMAKE_INCDIR = /opt/tslib/include
QMAKE_LIBDIR = /opt/tslib/lib
```

The contents of the modified file are as follows:



```
#
# qmake configuration for building with arm-none-linux-gnueabi-g++
#

include(../../common/linux.conf)
include(../../common/gcc-base-unix.conf)
include(../../common/g++-unix.conf)
include(../../common/qws.conf)

# modifications to g++.conf
QMAKE_CC                = arm-linux-gnueabihf-gcc -lts
QMAKE_CXX               = arm-linux-gnueabihf-g++ -lts
QMAKE_LINK              = arm-linux-gnueabihf-g++ -lts
QMAKE_LINK_SHLIB        = arm-linux-gnueabihf-g++ -lts

# modifications to linux.conf
QMAKE_AR                = arm-linux-gnueabihf-ar cqs
QMAKE_OBJCOPY           = arm-linux-gnueabihf-objcopy
QMAKE_STRIP             = arm-linux-gnueabihf-strip

load(qt_config)
QMAKE_INCDIR = /opt/tslib/include
QMAKE_LIBDIR = /opt/tslib/lib
~
```

After saving the file, execute the build-qt script for configuration.

```
od@Linux-host:~/qt-everywhere-opensource-src-4.8.6$ ./build-qt
```

After the configuration is complete, start to execute the following commands:

```
od@Linux-host:~/qt-everywhere-opensource-src-4.8.6$ make
```

After compiling, execute the installation command:

```
od@Linux-host:~/qt-everywhere-opensource-src-4.8.6$make install
```

7.3 Build QT SDK

7.3.1 QT SDK

Since Qt is a cross-platform graphics framework, users can first develop and debug Qt applications on the PC host. After the applications are basically formed, they can be transplanted to the target board. Therefore, users need to build QT SDK on the PC to improving development efficiency. The QT SDK includes:

QT library suitable for PC environment operation

QT Integrated Development Environment (Qt Creator)

7.3.2 Install Qt SDK

Under Ubuntu environment, the Linux version of the Qt SDK can be obtained through apt-get. When the Ubuntu host can access the Internet normally, you can use the following commands to obtain and install the Qt SDK:

```
od@Linux-host:~$ sudo apt-get install qt-sdk
```

During the installation of the Qt SDK, a Warning: Phonon is not functional warning may appear. Just press Enter and go to the next step.

After the installation is successful, you can see that there are two more executable files qmake and qmake-qt4 in the /usr/bin/ directory, we use qmake-qt4, in order to distinguish between qmake for local compilation and qmake for cross compilation, It is best to set an alias for one of qmake, for example, qmake-arm can be used to indicate that you want to use qmake for cross-compilation. This can be achieved by adding the following command at the end of the ~/.bashrc file.

```
alias qmake-arm=/opt/qt4.8.6/bin/qmake
```

7.4. qmake

Qt provides the qmake tool, which is a tool used to generate Makefiles for different platforms and compilers

Handwriting Makefile is more difficult and error-prone, especially when porting to the target board after PC development, you also need to write multiple Makefiles. When using qmake, developers only need to create a .pro file and run qmake corresponding to the platform to generate the corresponding Makefile.

For some simple projects (such as projects with a small amount of source code), you can directly execute the qmake -project command in the top-level directory of the project to automatically generate Pro files (with the suffix .pro); but for some complex Qt programs, The Pro file which was automatically generated often fails to meet the requirements, so the programmer needs to manually rewrite the Pro file. There are many rules for writing Pro files, and developers can find them online.

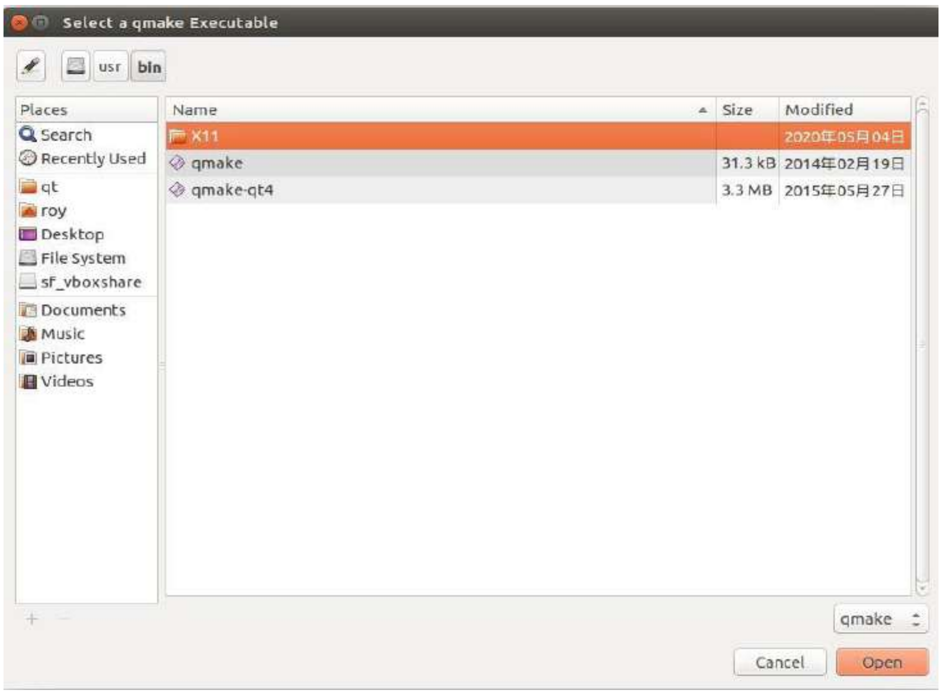
7.5 Qt Creator

7.5.1 Qt Creator Configuration

Qt Creator is a powerful cross-platform IDE that integrates editing, compiling, running, and debugging functions. Qt programming with Qt Creator can greatly improve efficiency and reduce development time. Developers can start directly throuth commands QtCreator:

```
od@linux-host: ~qtcreator
```

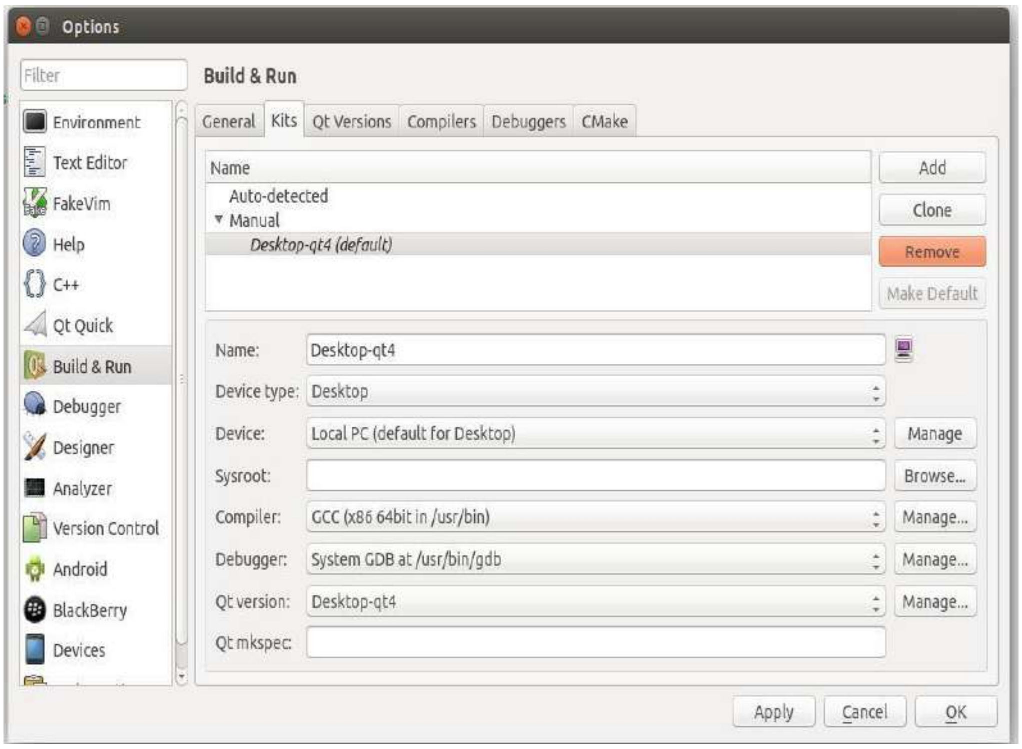
If you have installed the desktop version of Qt and the embedded version of Qt, you need to set the qmake version used by Qt Creator. Click Tools→Options in the menu, and then click Build & Run on the left. In the Build & Run that pops up on the right, select Qt Version and manually add the Qt version. Take the desktop version of Qt as an example: click Add, and select qmake. Executable file window, and then select the executable file in the path /usr/bin/qmake-qt4, as shown in the figure, click to open


Select a qmake Executable

usr / bin

Name	Size	Modified
X11		2020年05月04日
qmake	31.3 kB	2014年02月19日
qmake-qt4	3.3 MB	2015年05月27日

Enter the version name in the Version Name column, such as Destop-qt4, click Apply, then select Kits next to Qt Version, click Add, enter Name, select compiler, Qt-version, etc., click Apply, then click OK.


Name: Desktop-qt4

Device type: Desktop

Device: Local PC (default for Desktop)

Sysroot:

Compiler: GCC (x86 64bit in /usr/bin)

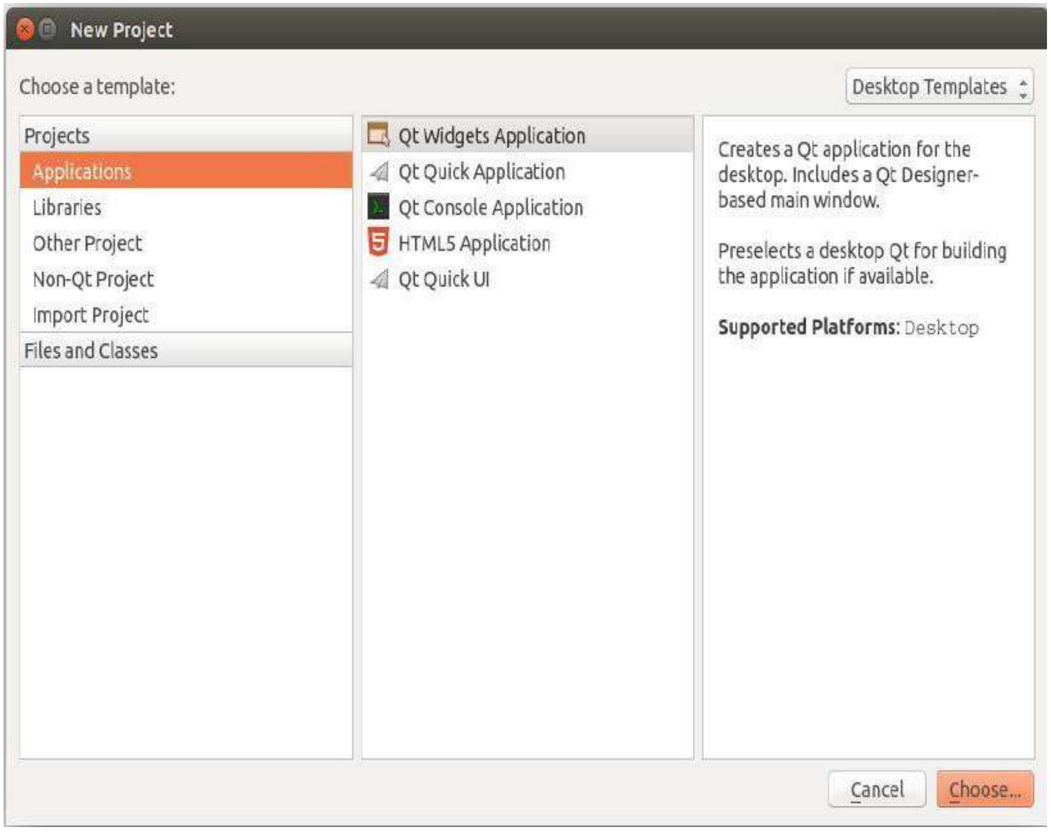
Debugger: System GDB at /usr/bin/gdb

Qt version: Desktop-qt4

Qt mkspec:

7.5.2 Use Qt Creator

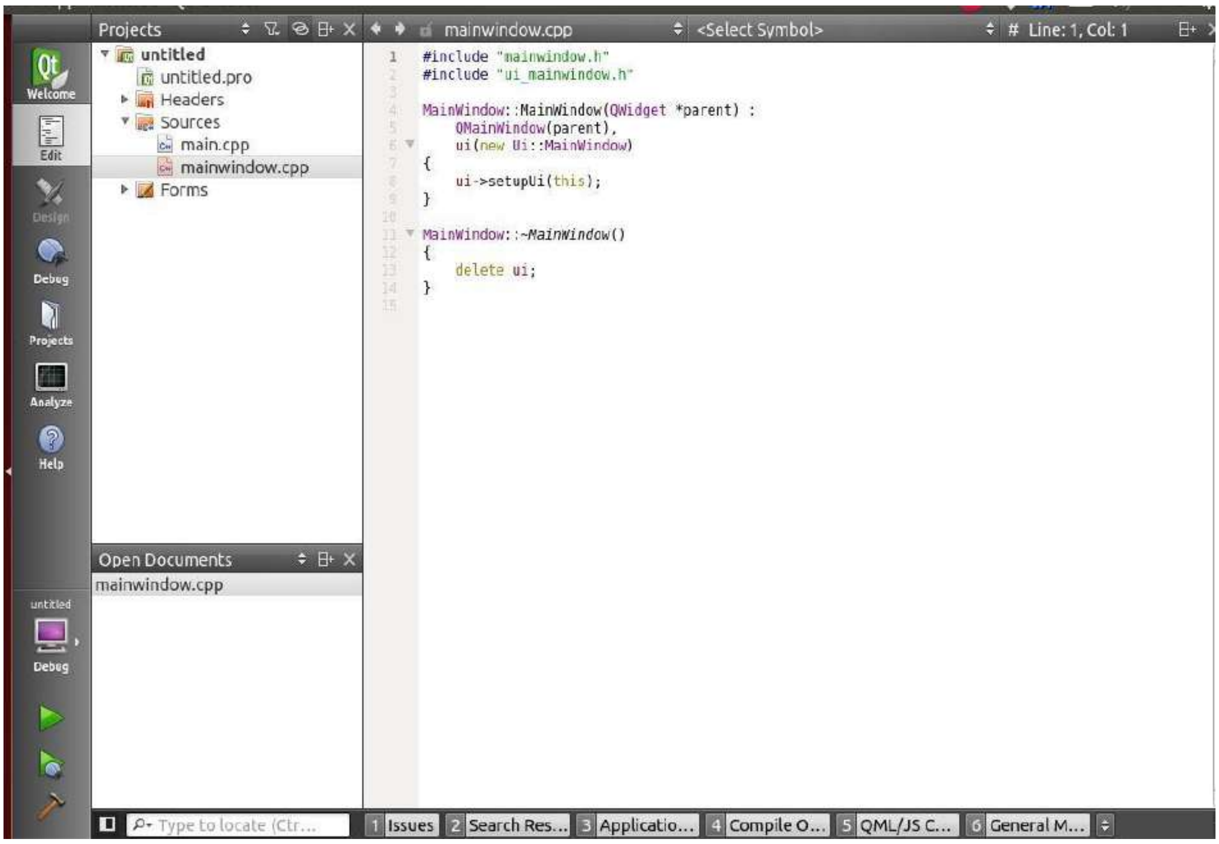
The following explains how to use Qt Creator to develop programs. Click New Project on the main interface of Qt Creator, select Qt Widgets Application, and click Choose:


Creates a Qt application for the desktop. Includes a Qt Designer-based main window.

Preselects a desktop Qt for building the application if available.

Supported Platforms: Desktop

Set the project name and path in the pop-up window. Then continue to the next step, choose the default method, and click Finish on the last page to complete the project creation and generate default code. Developers can develop their own functional design on basis of the default code.



```
#include "mainwindow.h"
#include "ui_mainwindow.h"

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

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