



Linux Development Guide (35 Series)

Contents

Chapter 1 Product Overview	1
1.1 OS Version	1
1.2 Shipping List.....	1
1.3 Optional Accessories.....	1
1.4 Recommended Tools and Consumables	1
Chapter 2 Tools.....	3
2.1 Tools Provided by DWIN	3
2.1.1 A40i Toolchain/SDK.....	3
2.1.2 Update Package DWIN_V1-x-x.tar.....	3
2.1.3 DEMO Projects.....	3
2.2 User-Provided Tools.....	3
Chapter 3 Debugging Interfaces.....	4
3.1 Serial Port Debugging.....	4
3.1.1 Wiring Example	4
3.1.2 MobaXterm Configuration	5
3.2 Ethernet Telnet Debugging	6
3.2.1 IP Configuration	6
3.2.2 MobaXterm Procedure	6
Chapter 4 Common Command Reference	8
4.1 Network Configuration	8
4.2 Accessing Web Pages with a Browser	9
Chapter 5 Update Package Guide.....	12
5.1 Creation Procedure.....	12
5.2 Usage Instructions.....	12
5.3 Update Package Examples.....	12
5.3.1 Changing the Application Launched at Startup	12
5.3.2 Changing the Boot Logo.....	13
Chapter 6 Firmware Flashing	16
Chapter 7 Environment Setup	18
7.1 Installing VMware Workstation.....	18
7.2 Installing Ubuntu	21
7.2.1 Overview.....	21
7.2.2 Downloading the Ubuntu Image.....	21
7.2.3 Installing Ubuntu	21
7.2.4 Installing Open-vm-tools (VMware Tools)	29
7.2.5 Configuring Shared Folders.....	30
7.3 Installing the A40i Toolchain.....	31
Chapter 8 Cross-Compiling Qt Project Files	33

8.1 Installing Qt Creator in Ubuntu	33
8.2 Opening the Project.....	33
8.3 Setting the Qmake Path.....	34
8.4 Configuring the Compiler Toolchain	35
8.5 Configuring Kits	36
8.6 Configuring Environment Variables	37
8.7 Building and Downloading/Deploying the Project.....	38
8.7.1 Running qmake.....	38
8.7.2 Build	38
8.7.3 Downloading/Deploying via USB Flash Drive	40
Chapter 9 Cross-Compiling LVGL Project Files	43
9.1 Software Requirements.....	43
9.2 Hardware.....	43
9.3 Cross-Compilation	43
9.3.1 Pre-Compilation Preparation	43
9.3.2 Compilation Procedure	44
9.3.3 Running the Test	45
9.4 Notes	47
9.4.1 Makefile.....	47
9.4.2 Third-Party Libraries	47
Chapter 10 Revision History	49

Chapter 1 Product Overview

1.1 OS Version

This series uses the Linux 3.10 kernel and is built with Buildroot.

1.2 Shipping List

- 1 × display (standard configuration)

1.3 Optional Accessories

The following accessories can be purchased from DWIN sales representatives according to project requirements:

- Camera: 5 MP MIPI interface; supports both landscape and portrait display orientations.
- Speaker:
- 8 Ω / 2 W (320 mm cable length, front sound output)
- 8 Ω / 0.8 W (180 mm cable length, side sound output)
- 4G module: Regional versions available for China/India, Europe, Australia, and other regions.

1.4 Recommended Tools and Consumables

- 12 V / 2 A power supply
- USB Type-A to Type-C or Micro-USB adapter cable (select the USB debugging interface type specified in the display datasheet)
- USB flash drive
- SD card
- Ethernet cable

- Adapter board and cables for connecting the serial port and power supply

Chapter 2 Tools

2.1 Tools Provided by DWIN

2.1.1 A40i Toolchain/SDK

Qt development Buildroot version: buildroot-A40i-QT-sdk-20220623.tar.gz

LVGL development Buildroot version: buildroot-A40i-LVGL9_1_0-sdk-20241219.tar

2.1.2 Update Package DWIN_V1-x-x.tar

Mechanism: Create a tar archive whose version number matches the current system version (the version number is stored in /etc/emcversion). After startup, the system automatically checks whether the update directory on the USB flash drive contains an update package with the matching version number. If found, it automatically extracts the package and runs the install.sh script.

Applications: Suitable for batch system upgrades, version updates, boot logo replacement, application deployment, and other scenarios.

2.1.3 DEMO Projects

Qt development: The Qt DEMO is used to verify the complete development workflow (compile, build, download, and run). Browser project: qtwebviewtest.

LVGL development: LvglDemo, lvgl_demo_source

2.2 User-Provided Tools

- MobaXterm (supports serial port and Telnet debugging)

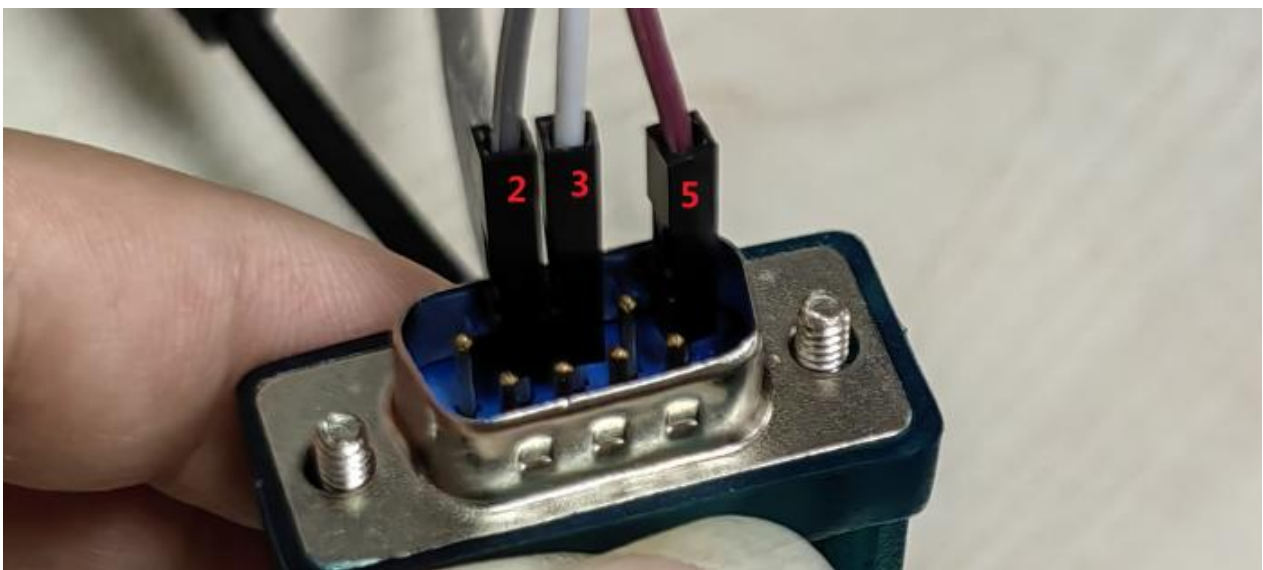
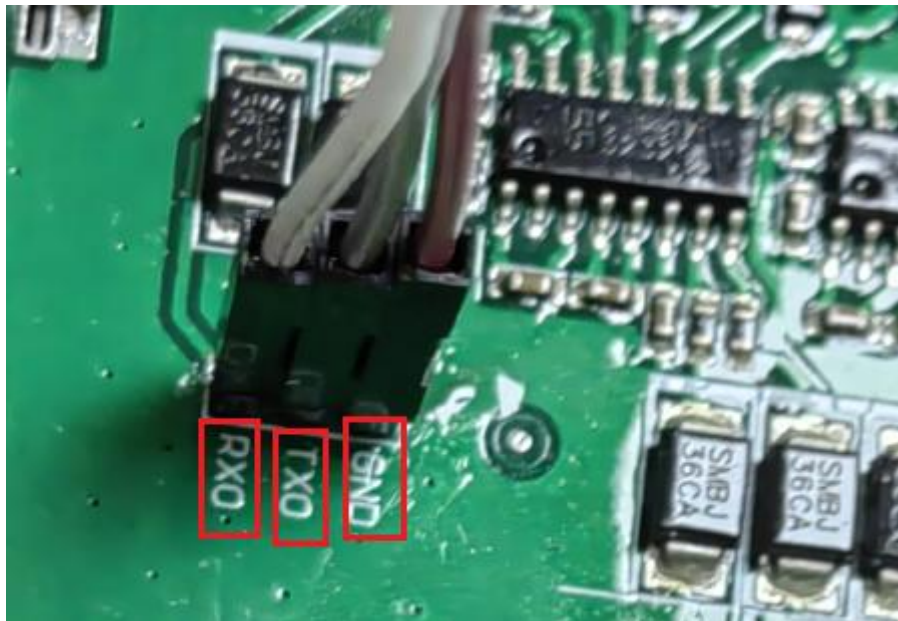
Chapter 3 Debugging Interfaces

The 35 Series Linux displays support two debugging methods: serial port and Ethernet Telnet.

3.1 Serial Port Debugging

3.1.1 Wiring Example

In the example below, Serial Port 0 is used for communication. Wiring: GND ↔ GND (5), RX0 ↔ TX (3), TX0 ↔ RX (2).

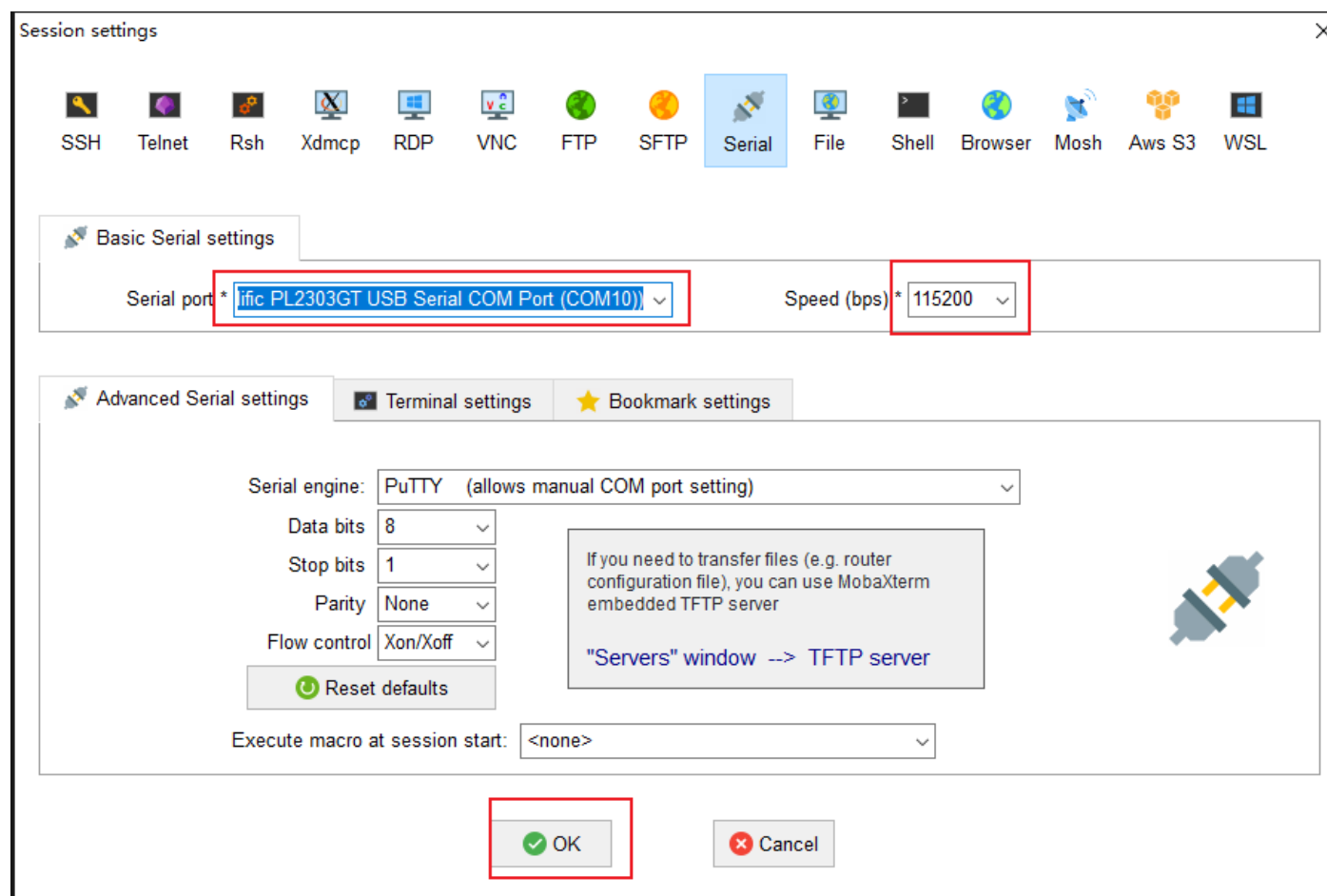


Headquarter: F18, Bldg 5, Jia 2, N. Xisanhuan Rd, Haidian Dist, Beijing 100081

Factory: No. 8, Venture Ave, Zhangjiang Innovation Park, Taoyuan, Hunan

Web: www.dwin-global.com | **Tel:** (86) 400 018 9008

3.1.2 MobaXterm Configuration



The screenshot shows the 'Session settings' dialog box in MobaXterm. The 'Serial' connection type is selected in the top toolbar. In the 'Basic Serial settings' tab, the 'Serial port' is set to 'iflic PL2303GT USB Serial COM Port (COM10)' and the 'Speed (bps)' is set to '115200'. In the 'Advanced Serial settings' tab, the 'Serial engine' is set to 'PuTTY (allows manual COM port setting)'. The 'Data bits' are set to '8', 'Stop bits' to '1', 'Parity' to 'None', and 'Flow control' to 'Xon/Xoff'. A 'Reset defaults' button is present. A text box provides information about the embedded TFTP server. The 'Execute macro at session start' is set to '<none>'. At the bottom, the 'OK' button is highlighted with a red box.

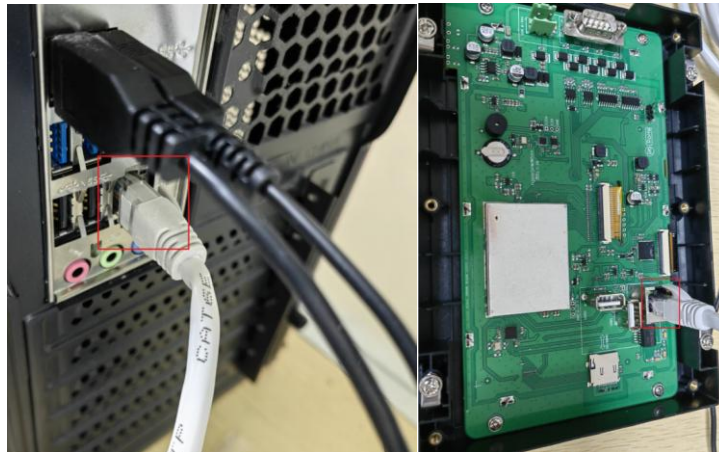
- Launch MobaXterm on the computer and click “Session” → “New session” on the top toolbar.
- Select the “Serial” connection type.
- Set the COM port number according to the COM port shown in Device Manager.
- Set the baud rate to 115200.
- Click “OK” to start the connection.
- After powering on the display, if no text interface is shown, enter root directly.

3.2 Ethernet Telnet Debugging

3.2.1 IP Configuration

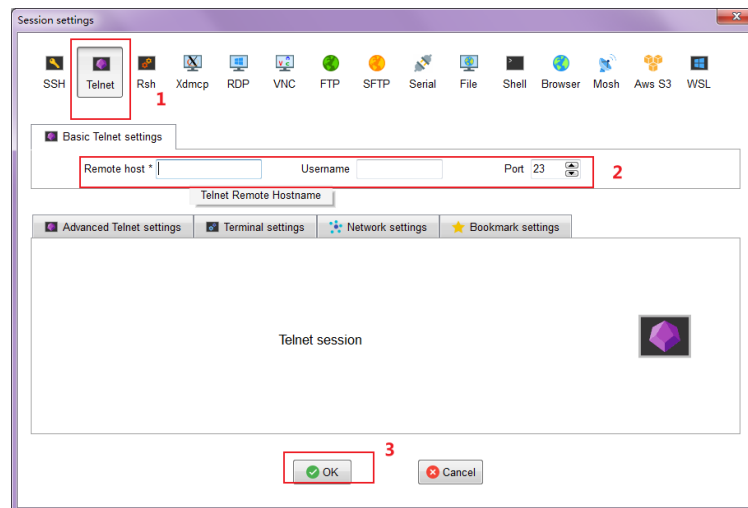
The default IP address of the display is 192.168.10.202. Make sure the computer and display are on the same subnet. If they are not:

- Connect the computer and display using an Ethernet cable.
- Set the computer to a static IP address in the form 192.168.10.xxx (xxx $\neq$ 202) and try to connect to the display.
- After the connection is successful, if the device needs to connect to a LAN, refer to Section 4.1 to modify the display IP address, then use two Ethernet cables to connect the device and computer separately.



3.2.2 MobaXterm Procedure

- Launch MobaXterm on the computer and click “Sessions” → “New session” → “Telnet” on the top toolbar.
- Enter the display IP address and click “OK”. Note: The default display IP address is 192.168.10.201 or 192.168.10.202. The computer and display must be on the same LAN to communicate.



- Click “OK”.
- Enter `root` to start operating the device.

Chapter 4 Common Command Reference

4.1 Network Configuration

- Use a serial connection to configure the display, or connect the display and computer with two Ethernet cables on the same LAN.

- After connecting the Ethernet cable, configure the gateway with: `route add default gw Gateway_IP` (192.168.10.1 is used as an example). After configuration, use `route -n` to check whether the gateway has been configured successfully.

```
# route add default gw 192.168.10.1
```

- Configure DNS with `vi /etc/resolv.conf`. Press “i” to enter insert mode, enter `nameserver 8.8.4.4`, press “Esc”, then enter `:wq` to save and exit.

- Try pinging an external address to verify the result.

```
# ping www.baidu.com
PING www.baidu.com (112.80.248.75): 56 data bytes
64 bytes from 112.80.248.75: seq=0 ttl=55 time=88.332 ms
64 bytes from 112.80.248.75: seq=1 ttl=55 time=109.084 ms
64 bytes from 112.80.248.75: seq=2 ttl=55 time=68.276 ms
64 bytes from 112.80.248.75: seq=3 ttl=55 time=73.401 ms
64 bytes from 112.80.248.75: seq=4 ttl=55 time=103.740 ms
64 bytes from 112.80.248.75: seq=5 ttl=55 time=60.290 ms
64 bytes from 112.80.248.75: seq=6 ttl=55 time=58.539 ms
^C
--- www.baidu.com ping statistics ---
7 packets transmitted, 7 packets received, 0% packet loss
round-trip min/avg/max = 58.539/80.237/109.084 ms
```

To permanently modify the gateway and DNS, add the following commands to `/etc/init.d/rcS` or `/etc/init.d/netconfig` (if available), after the `ifconfig eth0` IP address command:

```
route add default gw Gateway_IP
```

```
echo "nameserver 8.8.4.4" >> /etc/resolv.conf
```

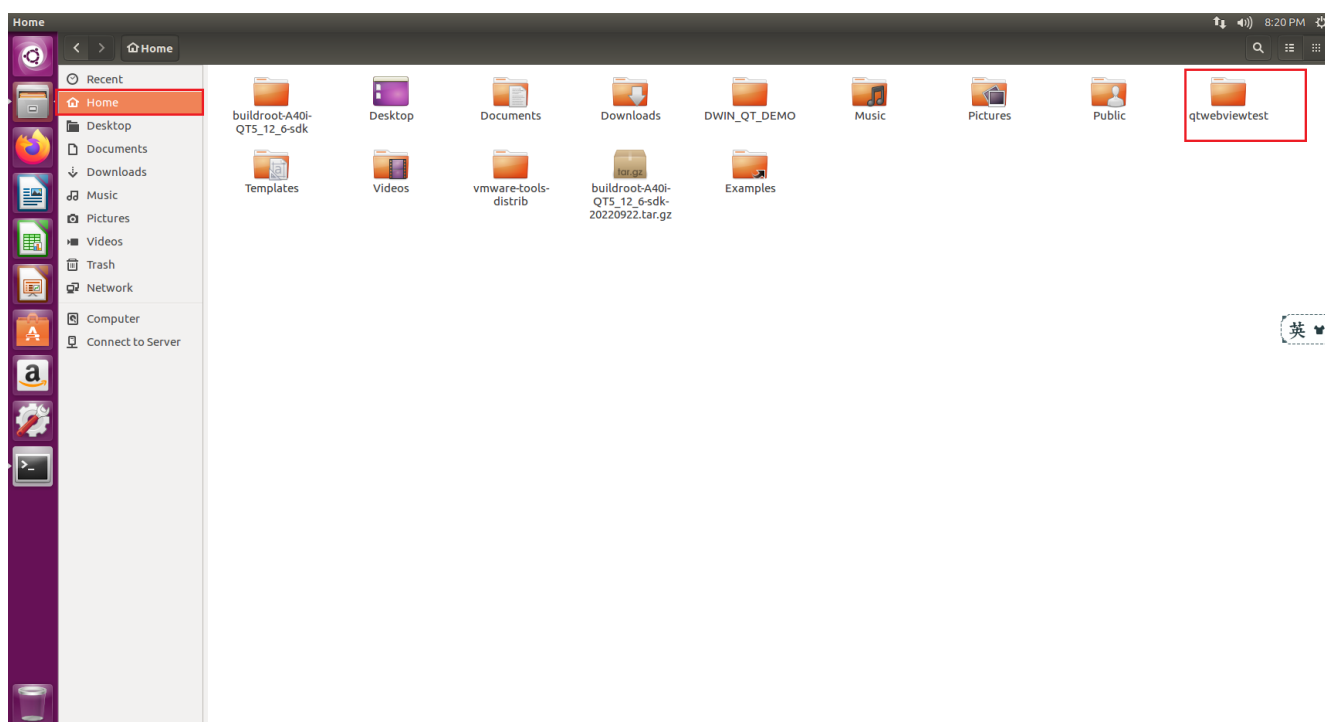
The modified configuration is shown in the original figure.

```
ifconfig eth0 192.168.10.205
route add default gw 192.168.10.1
echo "nameserver 8.8.4.4">> /etc/resolv.conf
```

4.2 Accessing Web Pages with a Browser

The 35 Series supports QtWebEngine, which can be used to implement browser functionality. The example project is qtwebviewtest.

- Place the project in the `/home/dwin/` directory.



- Configure the development environment (refer to Chapter 8), and run `qmake -v` to verify that the configuration is successful. Then enter the qtwebviewtest directory and complete the build.

```
dwin@ubuntu: ~/qtwebviewtest
dwin@ubuntu:~$ cd buildroot-A40i-QT5_12_6-sdk/
dwin@ubuntu:~/buildroot-A40i-QT5_12_6-sdk$ source env-setup.sh
dwin@ubuntu:~/buildroot-A40i-QT5_12_6-sdk$ cd ..
dwin@ubuntu:~$ cd qtwebviewtest/
dwin@ubuntu:~/qtwebviewtest$ qmake -v
QMake version 3.1
Using Qt version 5.12.6 in /home/dwin/buildroot-A40i-QT5_12_6-sdk/sysroot/usr/local/Qt-5.12.6/lib
dwin@ubuntu:~/qtwebviewtest$ ls
main.cpp      mainwindow.h  qtwebviewtest.pro  webbrowser.h
mainwindow.cpp mainwindow.ui  webbrowser.cpp
dwin@ubuntu:~/qtwebviewtest$ qmake
Info: creating stash file /home/dwin/qtwebviewtest/.qmake.stash
dwin@ubuntu:~/qtwebviewtest$ ls
main.cpp      mainwindow.h  Makefile           webbrowser.cpp
mainwindow.cpp mainwindow.ui  qtwebviewtest.pro  webbrowser.h
dwin@ubuntu:~/qtwebviewtest$ make
/home/dwin/buildroot-A40i-QT5_12_6-sdk/sysroot/usr/local/Qt-5.12.6/bin/uic mainwindow.ui -o ui_mainwindow.h
arm-linux-gnueabihf-g++ -c -pipe --sysroot=/home/dwin/buildroot-A40i-QT5_12_6-sdk/sysroot -O2 -std=gnu++11 -Wall -W -D_REENTRANT -fPIC -DQT_NO_DEBUG -DQT_WEBENGINEWIDGETS_LIB -DQT_WEBENGINECORE_LIB -DQT_QUICK_LIB -DQT_WIDGETS_LIB -DQT_GUI_LIB -DQT_WEBCHANNEL_LIB -DQT_QML_LIB -DQT_NETWORK_LIB -DQT_POSITIONING_LIB -DQT_CORE_LIB -I. -I../buildroot-A40i-QT5_12_6-sdk/sysroot/usr/local/Qt-5.12.6/include
```

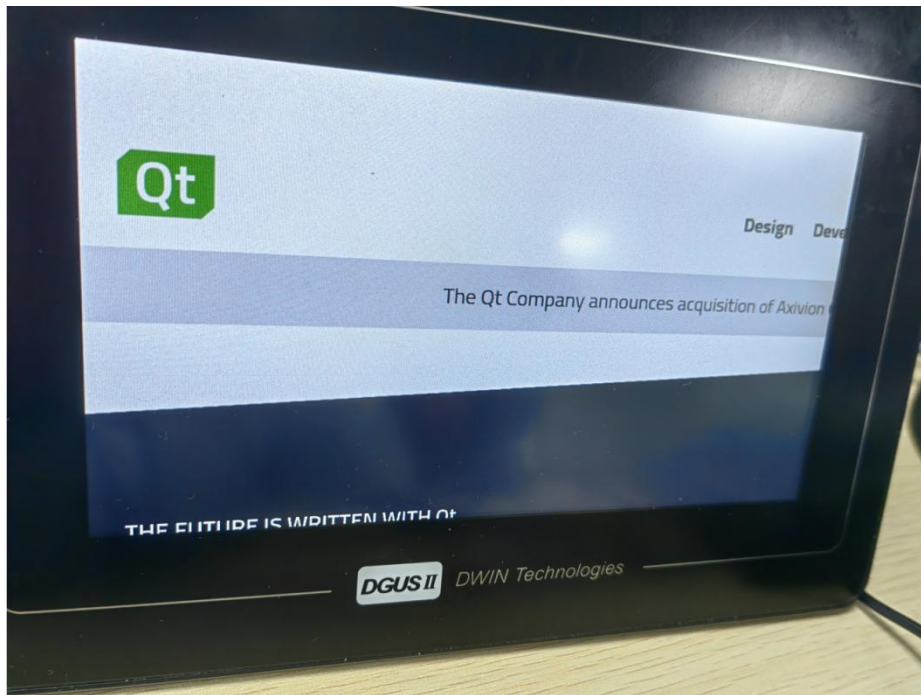
- Copy the generated executable to the display using a USB flash drive.
- Edit `vi /etc/init.d/runqt`:

```
# vi runqt
#!/bin/sh
export LD_LIBRARY_PATH=/usr/local/Qt-5.12.6/lib:$LD_LIBRARY_PATH
export QT_PLUGIN_PATH=/usr/local/Qt-5.12.6/plugins
export QT_QPA_FONTDIR=/usr/local/Qt-5.12.6/fonts
export QT_QPA_PLATFORM_PLUGIN_PATH=$QT_PLUGIN_PATH/platforms
export QT_QPA_PLATFORM="linuxfb:fb=/dev/fb0:size=1024x600:mmssize=169x179"
export PATH=/usr/local/Qt-5.12.6/bin:$PATH
export QTDIR=/usr/local/Qt-5.12.6
export QT_ROOT=/usr/local/Qt-5.12.6
export QT_QPA_GENERIC_PLUGINS=tslib,evdevmouse:/dev/input/event4
#export QT_QPA_USE_DWINTOUCH=1
#qtesttool &
/usr/bin/qtwebviewtest --no-sandbox &
```

- Move the cursor to the beginning of the `qtesttool` line, press “i” to enter insert mode, enter #, then move the cursor to the end of the line, press Enter to move to the next line, and enter the absolute path to the `qtwebviewtest` executable followed by a space, `--no-sandbox`, another space, and `&`.
- Press “Esc” to exit insert mode, enter `:wq`, and save the changes.

- Before running the program, configure the network (the gateway setting described above becomes invalid after a reboot), and then run the program in the `/etc/init.d/` directory.

```
# ./runqt
# xkbcommon: ERROR: failed to add default include path /usr/share/X11/xkb
Failed to create xkb context
This plugin does not support createPlatformOpenGLContext!
Cannot open mouse input device /dev/input/event4 (No such file or directory)
evdevmouse: Failed to open mouse device /dev/input/event4
WebBrowser construct!
WebEngineContext used before QtWebEngine::initialize() or OpenGL context creatio
n failed.
shared memfd open() failed: Function not implemented
url changed, url: "https://www.qt.io/"
js: UF: Pollyfill not needed, skipping.
```



- The device will display the default web page <http://www.qt.io>.

Chapter 5 Update Package Guide

This chapter explains how to create and use update packages to automatically update system applications, logos, and other content. The update package is provided by DWIN.

5.1 Creation Procedure

- Create a new directory in an Ubuntu environment and place all files that need to be updated in it.
- Add an **install.sh** script to the directory to copy files and modify file attributes.
- Create a tar archive using: `tar -cvf DWIN_V1-X-X.tar <INSTALL>`
- Copy a file named in the form `DWIN_V1.X.X.tar` to the **update** directory on a USB flash drive.

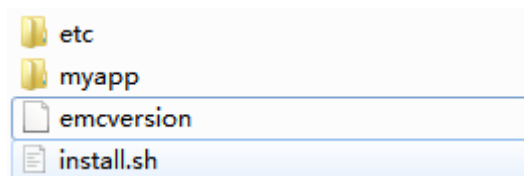
5.2 Usage Instructions

- Power off the display and insert the USB flash drive containing the update package.
- After powering on, the display automatically detects and starts the update. When the update is complete, the display automatically powers off.

5.3 Update Package Examples

5.3.1 Changing the Application Launched at Startup

Folder structure before compression:



- The `etc` folder stores script files in `/etc/init.d/` that may need to be modified (it may be omitted if not required).
- The `myapp` folder stores the files to be upgraded or modified.
- The `emcversion` file stores the updated version number used to update the version number on the device.

Example install.sh file (modify it as required):

```
#!/bin/sh

copy_dir()
{
    if [ -d $1 ]; then
        for libfile in $1/*; do
            if [ -f $libfile ]; then
                cp $libfile $2/
                chmod $3 $2/${libfile##*/}
                #echo $2/${libfile##*/}
            fi
        done
    fi
}

instmdir=$(cd `dirname $0`; pwd)

# update the emcversion
cp $instmdir/emcversion /etc/

# copy application file
cp $instmdir/myapp/myapp /usr/local/bin/myapp

# modify permission
chmod 755 /usr/local/bin/myapp

# modify runqt script file if needed
cp -a $instmdir/etc/init.d/* /etc/init.d/
```

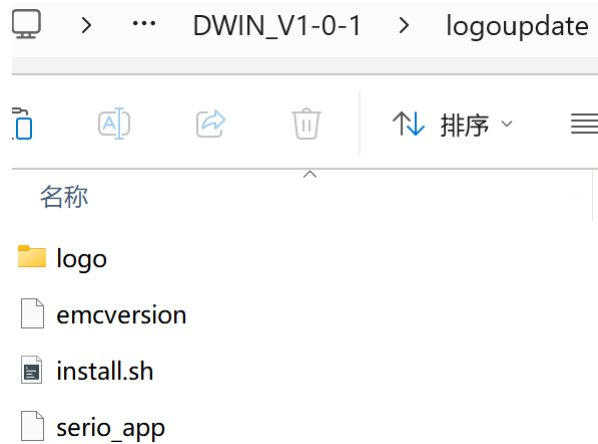
5.3.2 Changing the Boot Logo

Folder structure of DWIN_V1-x-x.tar before compression:

Headquarter: F18, Bldg 5, Jia 2, N. Xisanhuan Rd, Haidian Dist, Beijing 100081

Factory: No. 8, Venture Ave, Zhangjiang Innovation Park, Taoyuan, Hunan

Web: www.dwin-global.com | **Tel:** (86) 400 018 9008



- The emcversion file stores the updated version number used to update the version number on the device.
- The logo folder stores the replacement logo image. The image must be named bootlogo.bmp and must be a 24-bit

BMP image. The logo image must be stored at: logoupdate/logo/bootlogo.bmp

Example install.sh file:

```
#!/bin/sh

copy_dir()
{
    if [ -d $1 ]; then
        for libfile in $1/*; do
            if [ -f $libfile ]; then
                cp $libfile $2/
                chmod 755 $2/${libfile##*/}
                #echo $2/${libfile##*/}
            fi
        done
    fi
}

instmdir=$(cd `dirname $0`; pwd)
```

Headquarter: F18, Bldg 5, Jia 2, N. Xisanhuan Rd, Haidian Dist, Beijing 100081

Factory: No. 8, Venture Ave, Zhangjiang Innovation Park, Taoyuan, Hunan

Web: www.dwin-global.com | **Tel:** (86) 400 018 9008

```
# update the emcversion

cp $instdir/emcversion /etc/

# copy logo file

if [ -f $instdir/logo/bootlogo.bmp ]; then

    mkdir -p /extp/temp0p2

    mount /dev/mmcblk0p2 /extp/temp0p2

    cp -a $instdir/logo/bootlogo.bmp /extp/temp0p2/

    umount /extp/temp0p2

    rm -r /extp/temp0p2

    sync

fi

sync

$instdir/serio_app
```

- Compress the logoupdate directory into a tar file and name it DWIN_V1.0.1.tar. Copy it to the update directory on a USB flash drive→power off→insert the USB flash drive→power on again. After the upgrade succeeds, a beep will sound, followed by the display turning off. Remove the USB flash drive, power off and restart the display, and check whether the boot logo is correct.

- Notes:

Testing shows that under Windows, the bootlogo.bmp file can be replaced after extraction and the package can then be recompressed successfully. If the upgrade fails, the install.sh script may not have execute permission. Check its execute permission in a Linux environment.

Command for creating a tar file in Linux: **tar -cvf DWIN_V1.X.X.tar <INSTALL>**

Do not place too many files on the USB flash drive. A dedicated USB flash drive containing only the update files is recommended; otherwise, the update may fail.

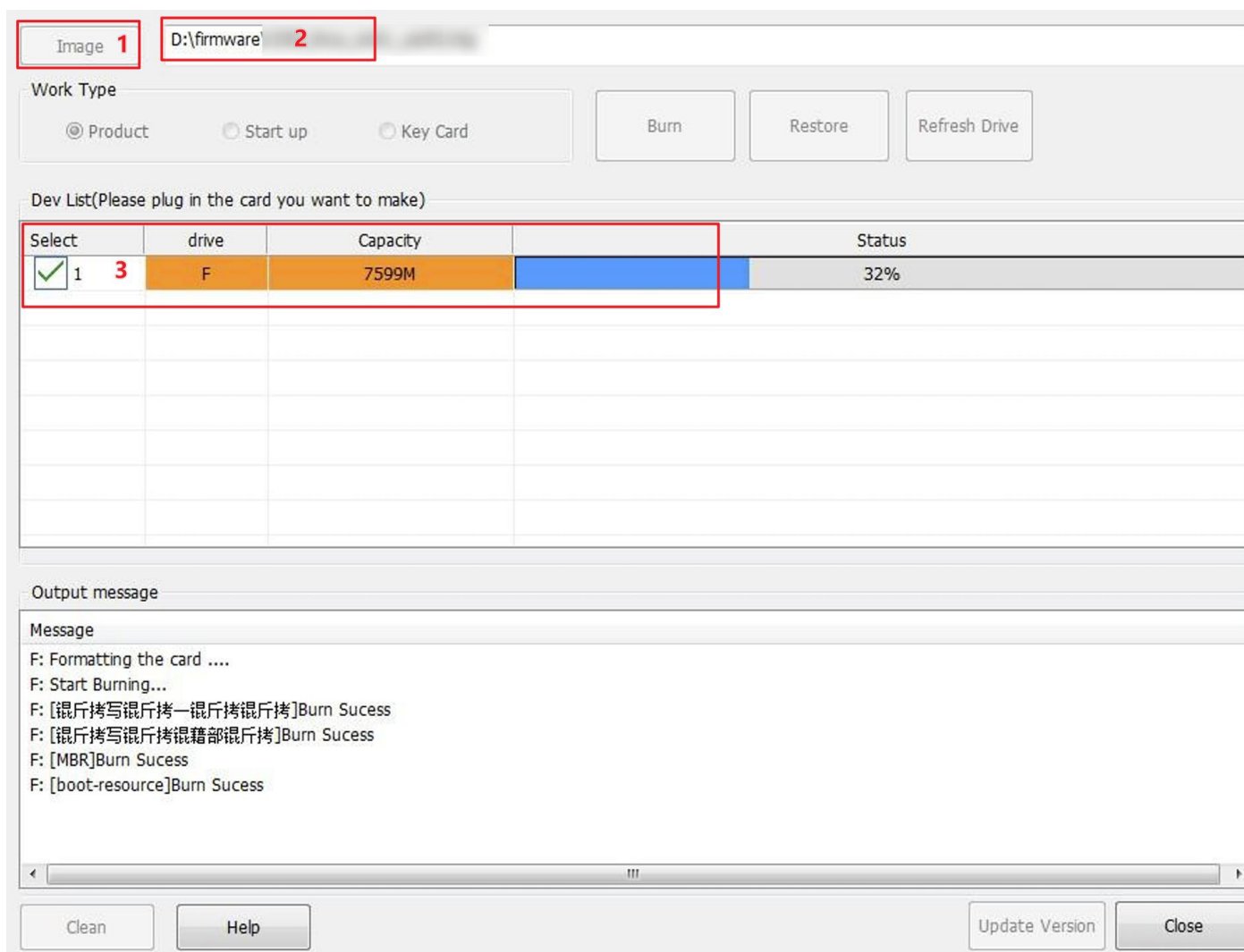
Chapter 6 Firmware Flashing

SD card: 8 GB or larger

Software tool: PhoenixCard (provided by DWIN)

Firmware: Provided by DWIN

- Insert the SD card reader containing the SD card into a USB port on the computer, open PhoenixCard, and load the firmware.
- Select the drive letter to be used for flashing and click “Burn”.



The screenshot shows the PhoenixCard software interface. At the top, there are two input fields: "Image" (labeled 1) and "D:\firmware" (labeled 2). Below these are three radio buttons for "Work Type": "Product" (selected), "Start up", and "Key Card". To the right are three buttons: "Burn", "Restore", and "Refresh Drive". Below the work type section is a table titled "Dev List(Please plug in the card you want to make)". The table has four columns: "Select", "drive", "Capacity", and "Status". The first row is highlighted with a red box and contains the following data: "1" (with a checkmark), "F", "7599M", and "32%". Below the table is an "Output message" section with a scrollable text area showing the following messages: "F: Formatting the card", "F: Start Burning...", "F: [银斤拷写银斤拷—银斤拷银斤拷]Burn Sucess", "F: [银斤拷写银斤拷银籍部银斤拷]Burn Sucess", "F: [MBR]Burn Sucess", and "F: [boot-resource]Burn Sucess". At the bottom of the interface are four buttons: "Clean", "Help", "Update Version", and "Close".

Select	drive	Capacity	Status
✓ 1	F	7599M	32%

Output message

Message

F: Formatting the card

F: Start Burning...

F: [银斤拷写银斤拷—银斤拷银斤拷]Burn Sucess

F: [银斤拷写银斤拷银籍部银斤拷]Burn Sucess

F: [MBR]Burn Sucess

F: [boot-resource]Burn Sucess

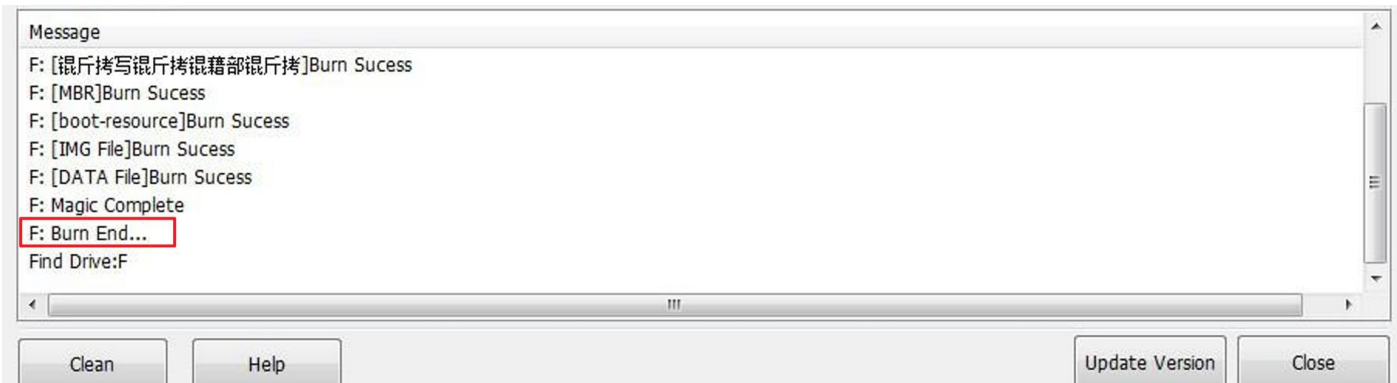
- Wait for the flashing process to finish. “Burning finished” indicates completion. (If the tool prompts you to format

Headquarter: F18, Bldg 5, Jia 2, N. Xisanhuan Rd, Haidian Dist, Beijing 100081

Factory: No. 8, Venture Ave, Zhangjiang Innovation Park, Taoyuan, Hunan

Web: www.dwin-global.com | **Tel:** (86) 400 018 9008

the SD card when you click to view it, this is normal. Do not format the card.)



- Remove the SD card, insert it into the SD card slot of the display, and power on the display. The display will automatically start the upgrade.
- Wait approximately 5 minutes. After the upgrade is complete, remove the SD card and power on the display again.

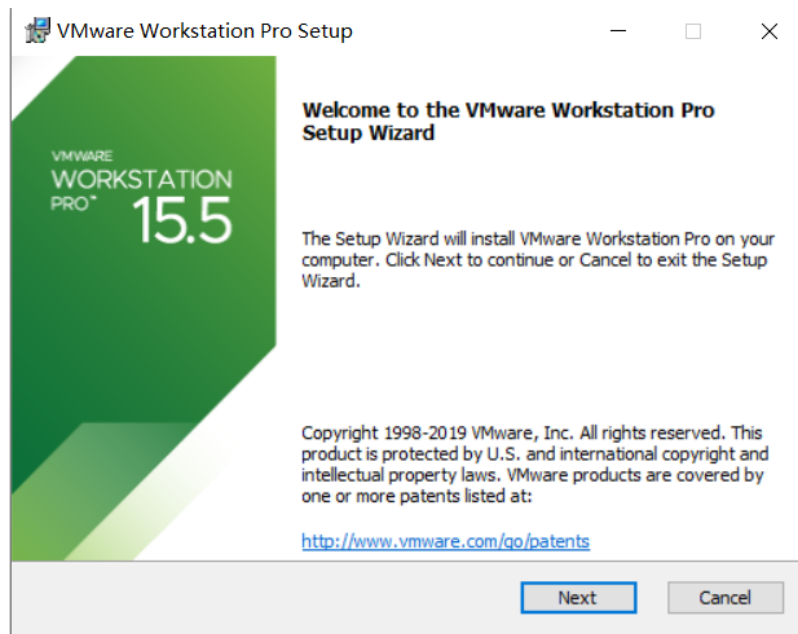
Note: For resistive touch screens, touch calibration is required on the first startup after the upgrade.

Chapter 7 Environment Setup

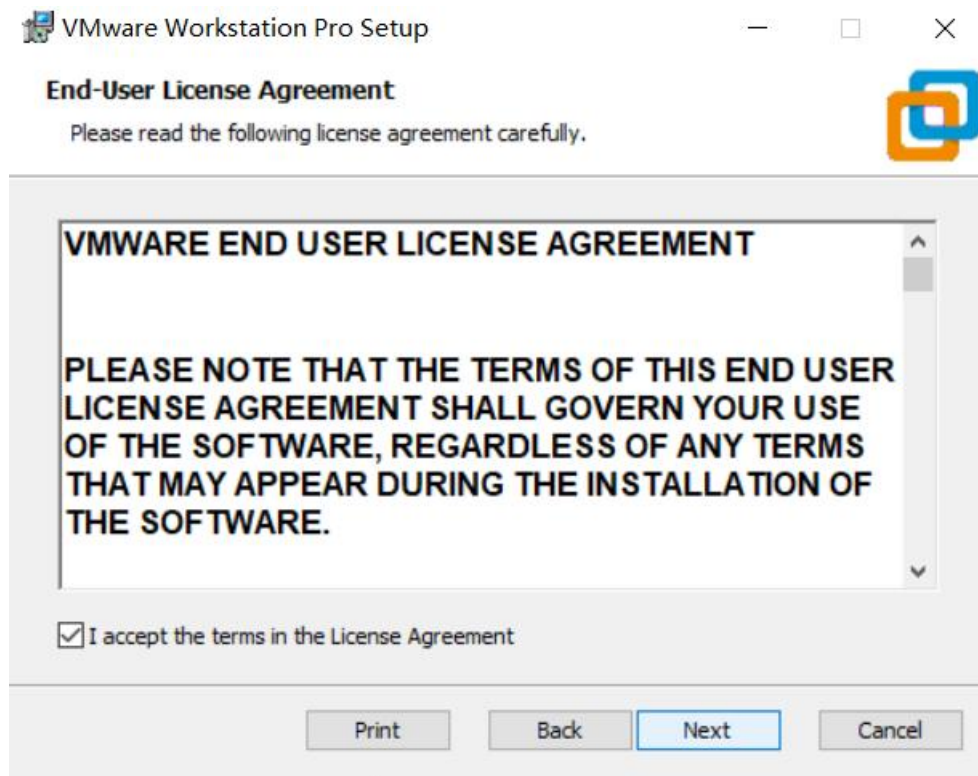
This chapter covers VMware installation, Ubuntu installation, shared folder configuration, toolchain configuration, and related setup. Follow the sections step by step.

7.1 Installing VMware Workstation

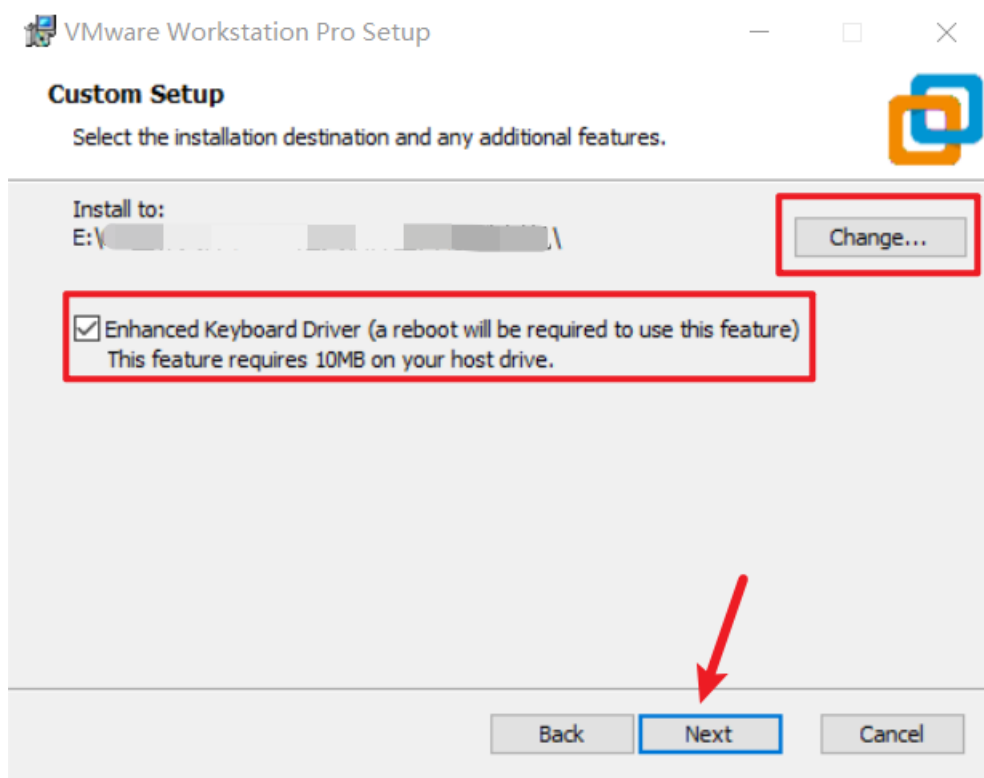
- Download and run the VMware Workstation installer.



- Read and accept the End User License Agreement, then click “Next”.



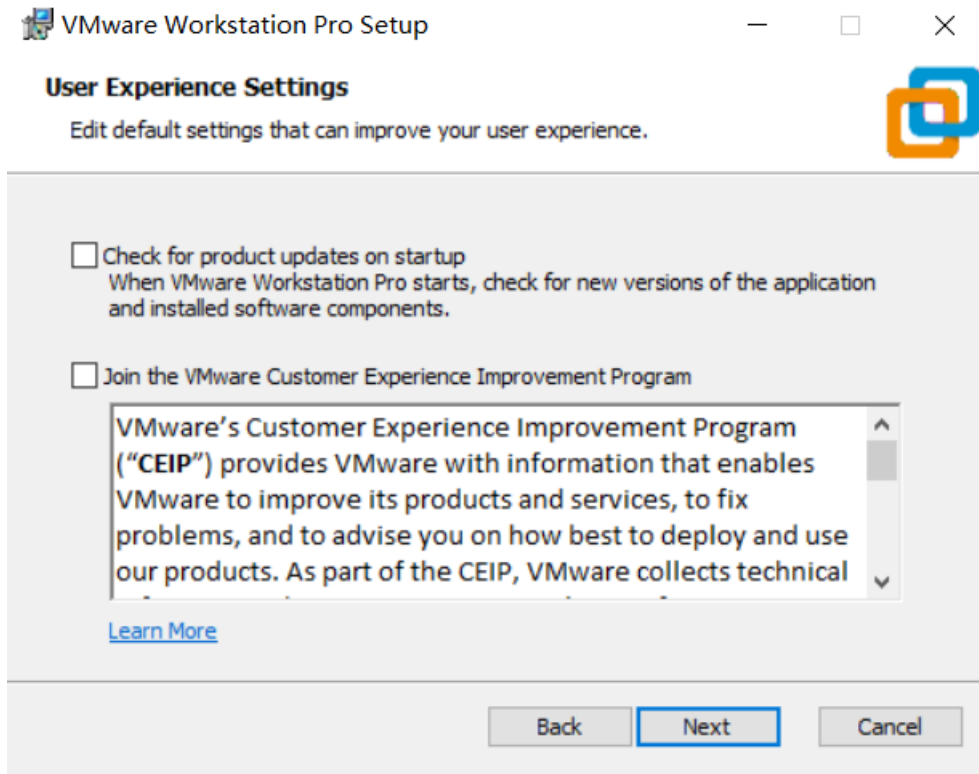
- Set the installation path (the default location may be used), select “Enhanced Keyboard Driver”, and click “Next”.



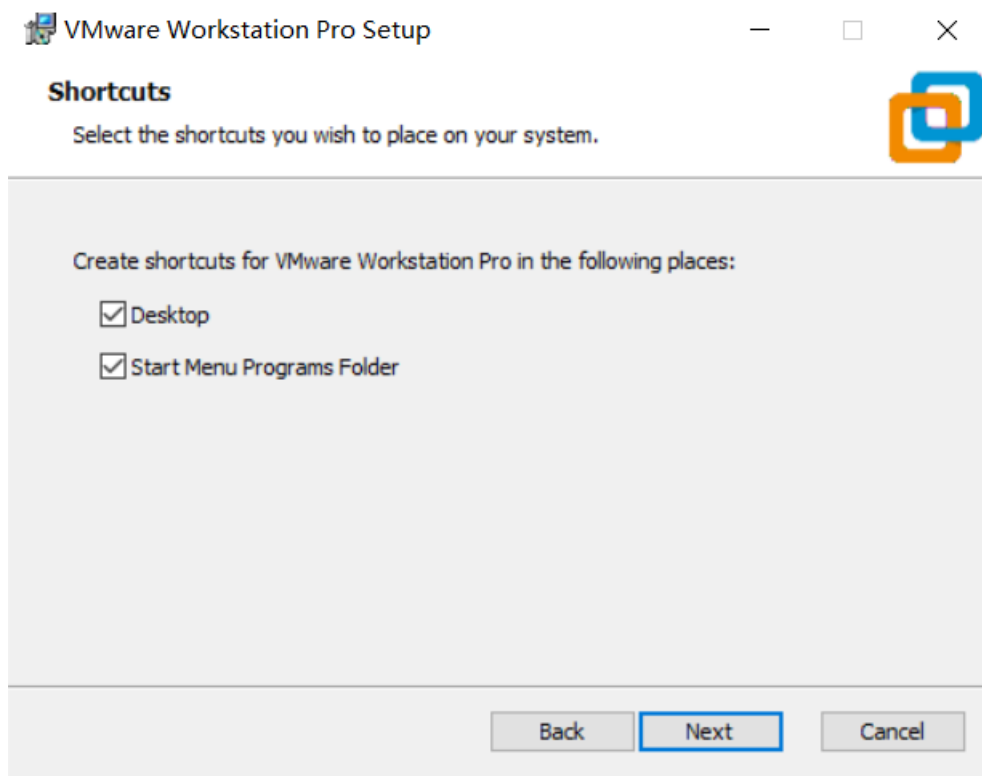
- As required, select whether to “Check for product updates on startup” and “Help improve VMware Workstation

Pro”, then click “Next”.

- Select “Create a desktop shortcut” and “Create a Start Menu shortcut”, then click “Next”.



- Click “Install”. After installation is complete, click “Finish” to exit the installation wizard.



7.2 Installing Ubuntu

7.2.1 Overview

If users choose and install an Ubuntu version themselves, this section uses the latest version, Ubuntu 24.04, as an example. The interface may vary slightly between versions, but the basic procedure is similar.

7.2.2 Downloading the Ubuntu Image

Visit the official website: <https://releases.ubuntu.com/> and download Ubuntu 24.04.

Desktop image

The desktop image allows you to try Ubuntu without changing your computer at all, and at your option to install it permanently later. This type of image is what most people will want to use. You will need at least 1024MiB of RAM to install from this image.

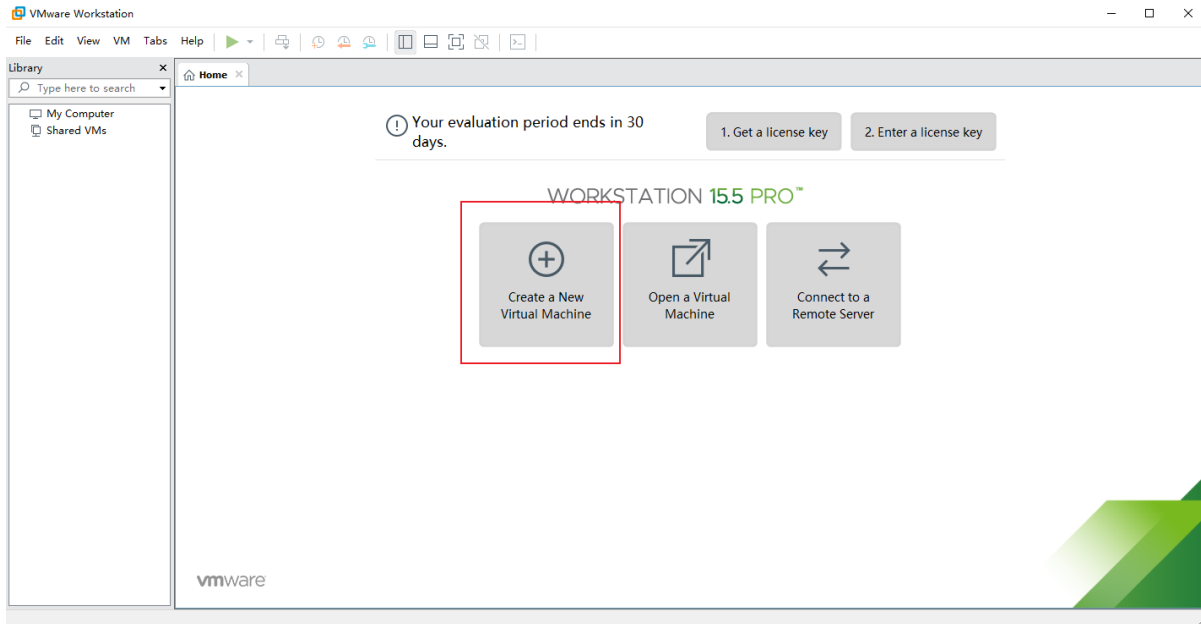
64-bit PC (AMD64) desktop image

Choose this if you have a computer based on the AMD64 or EM64T architecture (e.g., Athlon64, Opteron, EM64T Xeon, Core 2). Choose this if you are at all unsure.

7.2.3 Installing Ubuntu

- Open VMware and click “Create a New Virtual Machine”. The installation procedure for VMware Workstation

15.5 is used as an example. The procedure may vary depending on the version.

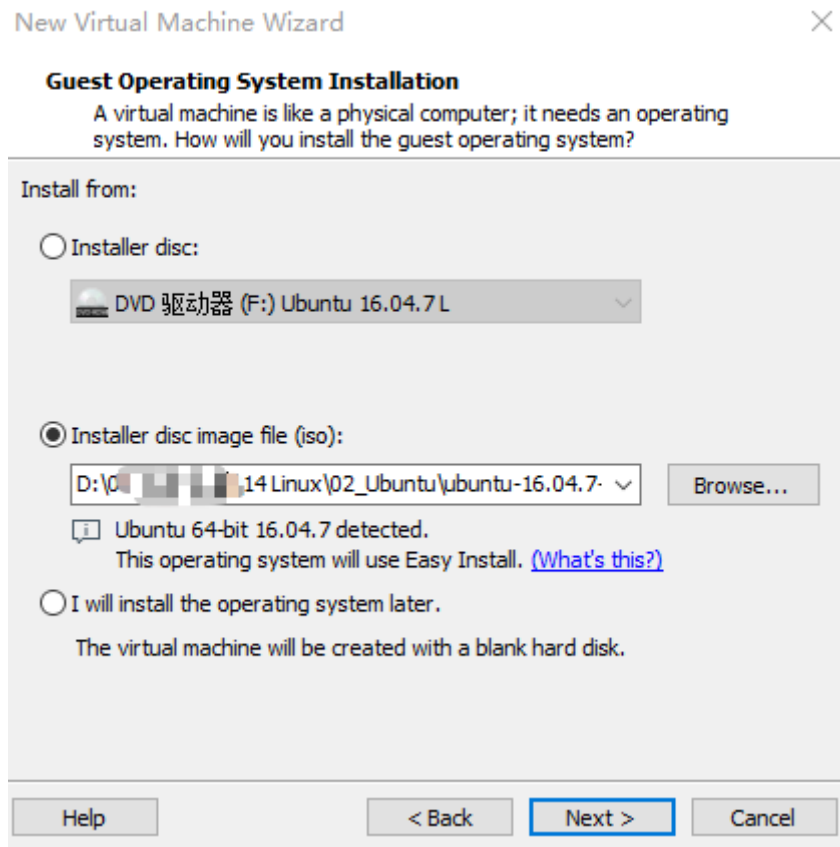


- Select “Custom” and click “Next”.

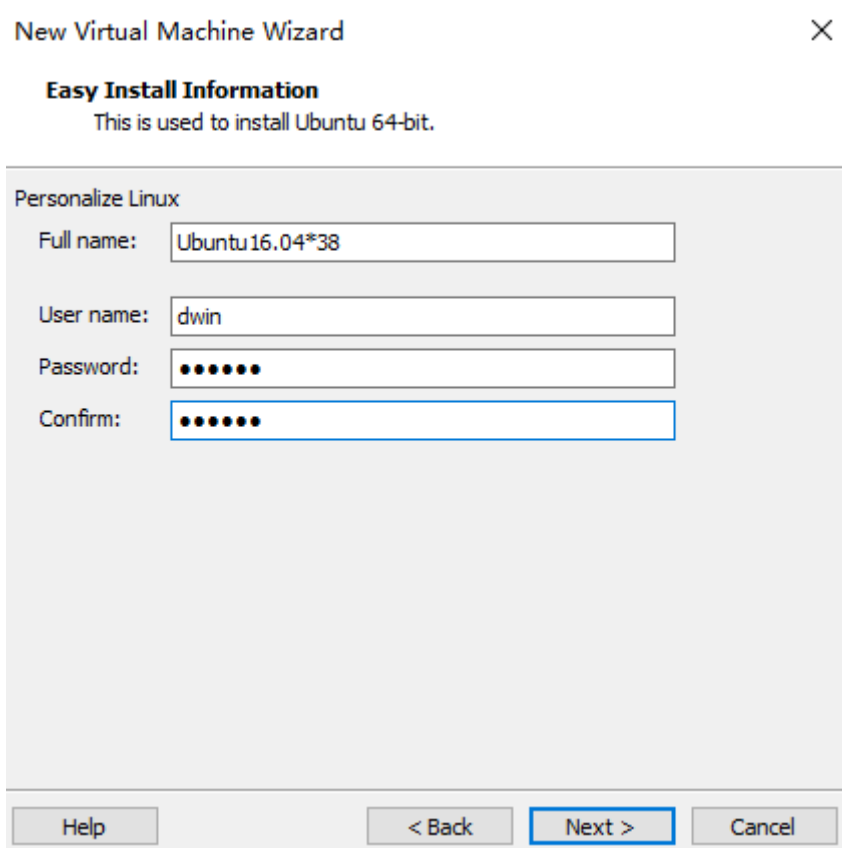


- Select “Installer disc image file”, click “Browse”, select the downloaded Ubuntu ***.iso file, and click “Next”.

The file will be automatically detected.



- Enter a custom name and password. The password is used for Ubuntu login and sudo privileges. Click “Next”.



New Virtual Machine Wizard

Easy Install Information
This is used to install Ubuntu 64-bit.

Personalize Linux

Full name:

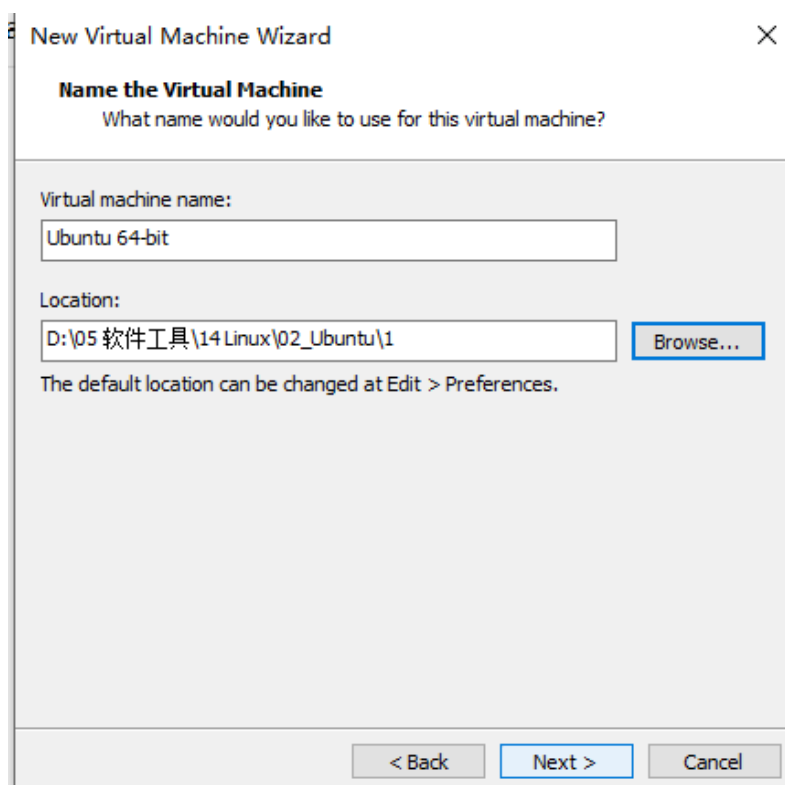
User name:

Password:

Confirm:

Help < Back Next > Cancel

- Set the Ubuntu name and installation location, then click “Next”.



New Virtual Machine Wizard

Name the Virtual Machine
What name would you like to use for this virtual machine?

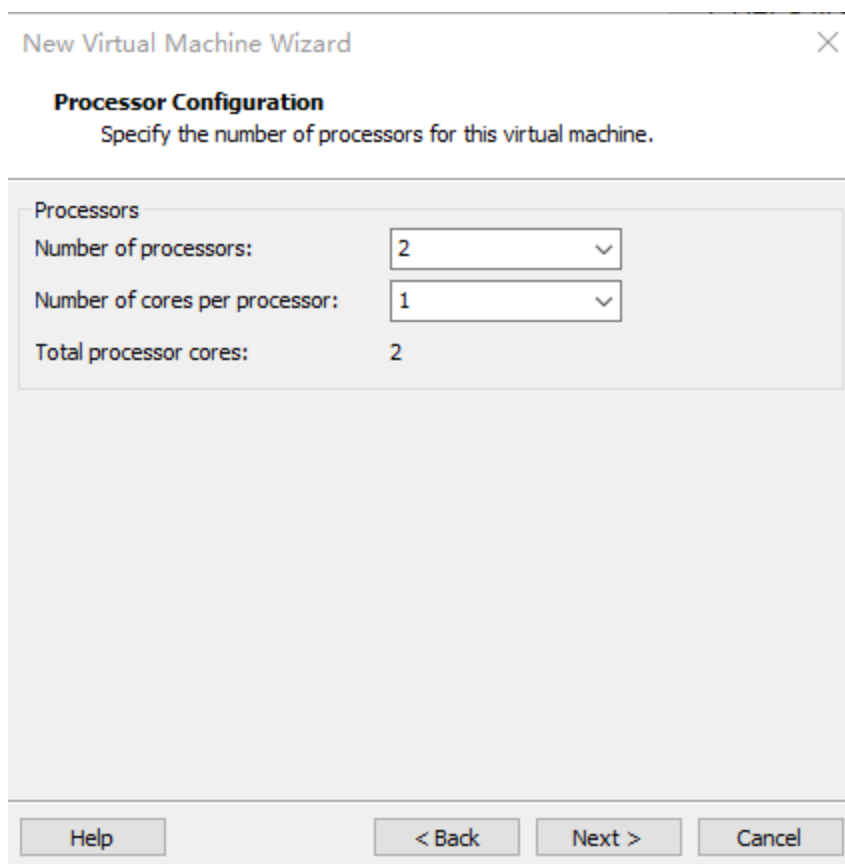
Virtual machine name:

Location:
 Browse...

The default location can be changed at Edit > Preferences.

< Back Next > Cancel

- Allocate the number of processors and processor cores according to your requirements and computer configuration (the example uses a total of 2 processor cores), then click “Next”.



New Virtual Machine Wizard

Processor Configuration
Specify the number of processors for this virtual machine.

Processors

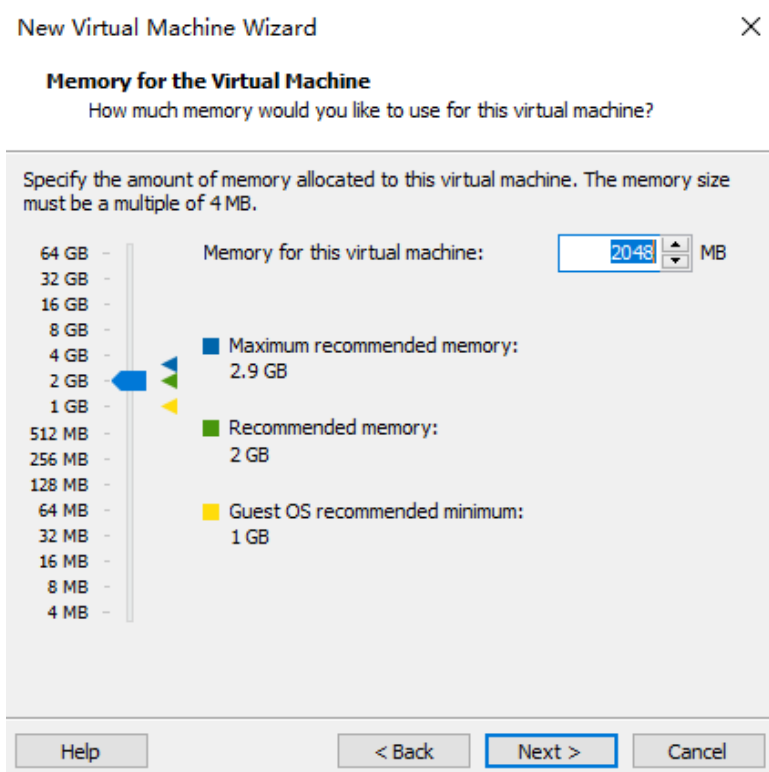
Number of processors: 2

Number of cores per processor: 1

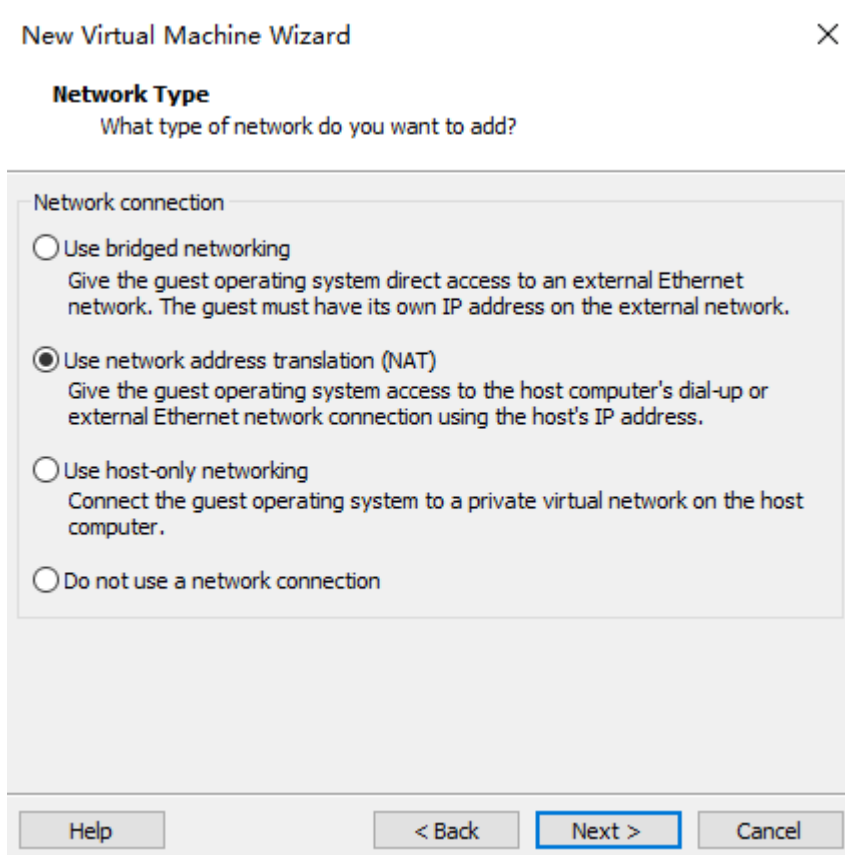
Total processor cores: 2

Help < Back Next > Cancel

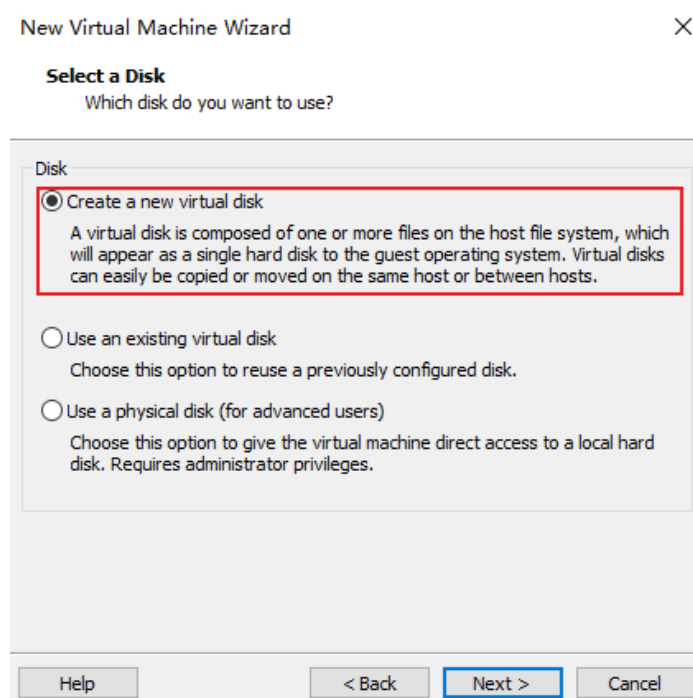
- The default memory is 2 GB and can be adjusted later. Click “Next”.



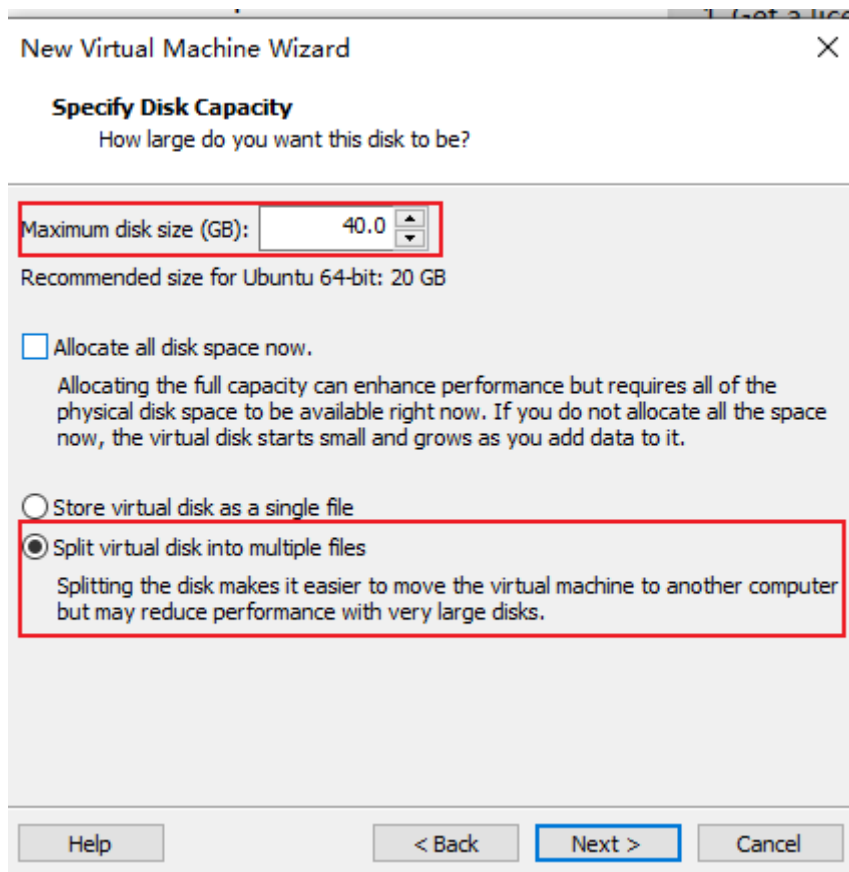
- Use the default configuration as required. For the network type, you may select bridged networking, which can be used for TFTP transfer. Click “Next” until “Select a Disk”.



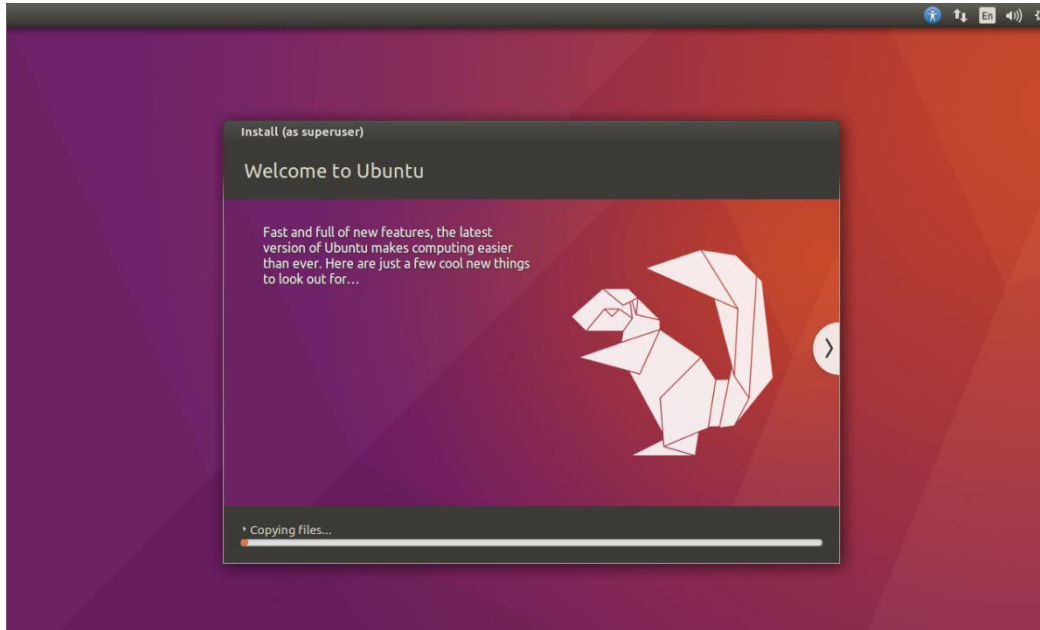
- Select “Create a new virtual disk” and click “Next”.



- Set the disk capacity. It is recommended to set the disk capacity according to the computer's available memory; when sufficient memory is available, 30 GB or more may be used. If additional disk space is required, refer to the subsequent section for expansion. Click “Next”.



- The disk will be named automatically; keep the default and click “Next”.
- Click “Finish” to start the virtual machine and begin installation.
- Wait patiently and follow the wizard until installation is complete.



Note: If the process becomes stuck while copying files, temporarily disconnect the network.

7.2.4 Installing Open-vm-tools (VMware Tools)

If “Virtual Machine” → “Settings” → “Options” → “Shared Folders” shows as disabled, VMware Tools has not been installed correctly.

Note: Ubuntu 24.04 can directly install open-vm-tools. On older versions such as Ubuntu 16.04, “Install VMware Tools” may appear disabled in VMware, in which case VMware Tools must be installed manually.

Press “**Ctrl+Alt+T**” to open the terminal. Enter **sudo apt update** and the password. When connected to the network, the installation will proceed automatically.

```
dwinvm@dwinvm-VMware-Virtual-Platform: ~
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

dwinvm@dwinvm-VMware-Virtual-Platform:~$ sudo apt update
[sudo] dwinvm 的密码:
获取:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
命中:2 http://archive.ubuntu.com/ubuntu noble InRelease
获取:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
获取:4 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [786
kB]
获取:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
获取:6 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [148
kB]
获取:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [2
1.6 kB]
获取:8 http://security.ubuntu.com/ubuntu noble-security/main Icons (48x48) [13.4
kB]
获取:9 http://security.ubuntu.com/ubuntu noble-security/main Icons (64x64) [20.0
kB]
获取:10 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metada
ta [7,068 B]
获取:11 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packag
es [931 kB]
获取:12 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-
```

- Enter: **sudo apt install open-vm-tools open-vm-desktop**

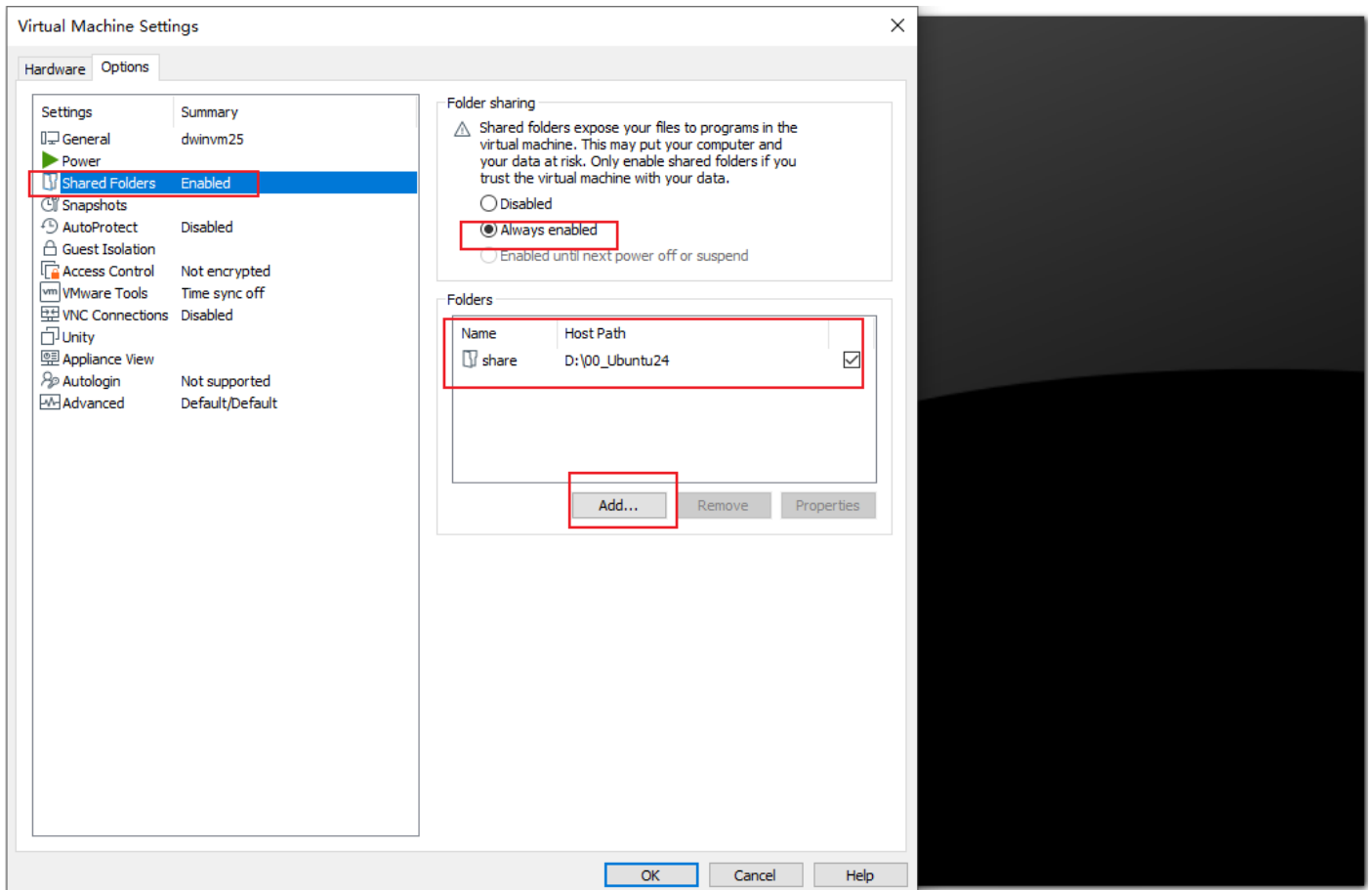
```
dwinvm@dwinvm-VMware-Virtual-Platform:~$ sudo apt install open-vm-tools open-vm-
tools-desktop
正在读取软件包列表... 完成
正在分析软件包的依赖关系树... 完成
正在读取状态信息... 完成
将会同时安装下列软件：
  libatkmm-1.6-1v5 libcairomm-1.0-1v5 libglibmm-2.4-1t64 libgtkmm-3.0-1t64
  libmspack0t64 libpangomm-1.4-1v5 libsigc++-2.0-0v5 libxmlsec1t64
  libxmlsec1t64-openssl zerofree
建议安装：
  open-vm-tools-containerinfo open-vm-tools-salt-minion
下列【新】软件包将被安装：
  libatkmm-1.6-1v5 libcairomm-1.0-1v5 libglibmm-2.4-1t64 libgtkmm-3.0-1t64
  libmspack0t64 libpangomm-1.4-1v5 libsigc++-2.0-0v5 libxmlsec1t64
  libxmlsec1t64-openssl open-vm-tools open-vm-tools-desktop zerofree
升级了 0 个软件包，新安装了 12 个软件包，要卸载 0 个软件包，有 198 个软件包未被
升级。
需要下载 2,050 kB 的归档。
```

The installation is now complete.

7.2.5 Configuring Shared Folders

- Click the power button in the upper-right corner of the Ubuntu desktop to shut down Ubuntu.
- After shutdown, on the corresponding virtual machine page, select “Edit virtual machine settings” → “Options” → “Shared Folders” → “Always enabled” → “Add” . Add a folder to be used for file transfer between the

host and virtual machine, then click “OK” .



- After configuring the shared folder, enter: `ls /mnt/hgfs/` (press Tab) `***` (folder).

```
dwinvm@dwinvm-VMware-Virtual-Platform:~$ ls /mnt/hgfs/share
36  buildroot-A40i-QT-sdk-20220623.tar.gz  linux.iso  QtDwinTest
dwinvm@dwinvm-VMware-Virtual-Platform:~$ cd /mnt/hgfs/share
dwinvm@dwinvm-VMware-Virtual-Platform:/mnt/hgfs/share$ sudo mv buildroot-A40i-QT-sdk-20220623.tar.gz ~/dwinvm
```

- Next, install the A40i toolchain in Ubuntu.

7.3 Installing the A40i Toolchain

Move the A40i package (buildroot-A40i-QT-sdk-20220623.tar.gz) into Ubuntu. You can use a shared folder or SFTP for transfer.

The following describes the shared-folder method:

Headquarter: F18, Bldg 5, Jia 2, N. Xisanhuan Rd, Haidian Dist, Beijing 100081

Factory: No. 8, Venture Ave, Zhangjiang Innovation Park, Taoyuan, Hunan

Web: www.dwin-global.com | **Tel:** (86) 400 018 9008

- Move the file to the root directory (example path: /home/dwinvm). When using a shared folder, enter: **sudo mv**

buil(Tab) ~ /dwinvm/ and wait briefly for the file to be moved to the root directory.

- Enter **tar -xvf bu(Tab)** to extract the file.

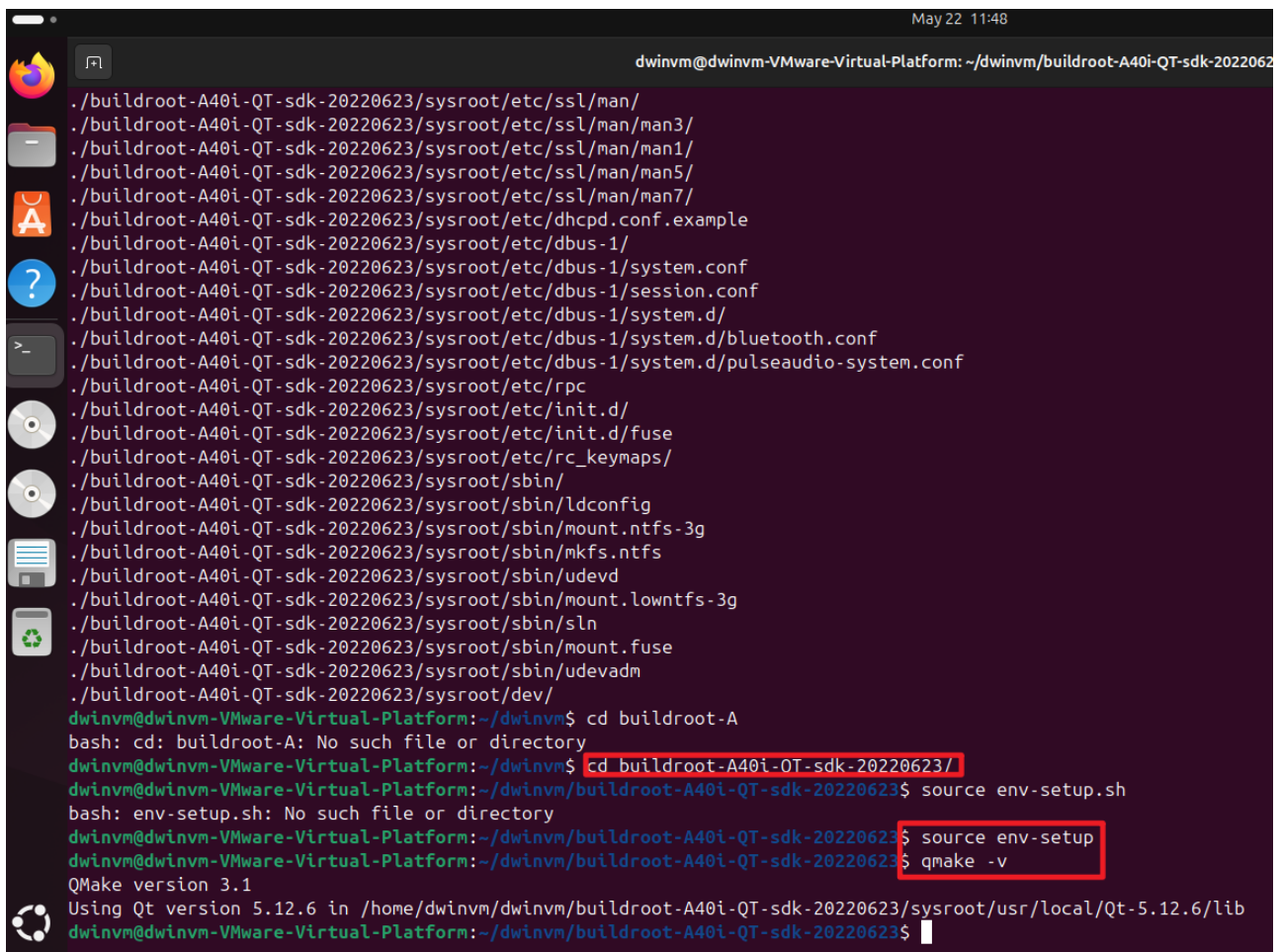
```
dwinvm@dwinvm-VMware-Virtual-Platform:~$ cd dwinvm
dwinvm@dwinvm-VMware-Virtual-Platform:~/dwinvm$ tar -xvf buildroot-A40i-QT-sdk-20220623.tar.gz
```

- Enter the following commands in sequence:

cd bui(Tab)

source env-setup

- Enter **qmake -v** to display the qmake version information and verify that the environment has been set up successfully.



```

May 22 11:48
dwinvm@dwinvm-VMware-Virtual-Platform: ~/dwinvm/buildroot-A40i-QT-sdk-20220623
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/ssl/man/
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/ssl/man/man3/
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/ssl/man/man1/
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/ssl/man/man5/
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/ssl/man/man7/
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/dhcpd.conf.example
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/dbus-1/
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/dbus-1/system.conf
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/dbus-1/session.conf
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/dbus-1/system.d/
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/dbus-1/system.d/bluetooth.conf
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/dbus-1/system.d/pulseaudio-system.conf
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/rpc
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/init.d/
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/init.d/fuse
./buildroot-A40i-QT-sdk-20220623/sysroot/etc/rc_keymaps/
./buildroot-A40i-QT-sdk-20220623/sysroot/sbin/
./buildroot-A40i-QT-sdk-20220623/sysroot/sbin/ldconfig
./buildroot-A40i-QT-sdk-20220623/sysroot/sbin/mount.ntfs-3g
./buildroot-A40i-QT-sdk-20220623/sysroot/sbin/mkfs.ntfs
./buildroot-A40i-QT-sdk-20220623/sysroot/sbin/udev
./buildroot-A40i-QT-sdk-20220623/sysroot/sbin/mount.lowntfs-3g
./buildroot-A40i-QT-sdk-20220623/sysroot/sbin/sln
./buildroot-A40i-QT-sdk-20220623/sysroot/sbin/mount.fuse
./buildroot-A40i-QT-sdk-20220623/sysroot/sbin/udevadm
./buildroot-A40i-QT-sdk-20220623/sysroot/dev/
dwinvm@dwinvm-VMware-Virtual-Platform:~/dwinvm$ cd buildroot-A
bash: cd: buildroot-A: No such file or directory
dwinvm@dwinvm-VMware-Virtual-Platform:~/dwinvm$ cd buildroot-A40i-QT-sdk-20220623/
dwinvm@dwinvm-VMware-Virtual-Platform:~/dwinvm/buildroot-A40i-QT-sdk-20220623$ source env-setup.sh
bash: env-setup.sh: No such file or directory
dwinvm@dwinvm-VMware-Virtual-Platform:~/dwinvm/buildroot-A40i-QT-sdk-20220623$ source env-setup
dwinvm@dwinvm-VMware-Virtual-Platform:~/dwinvm/buildroot-A40i-QT-sdk-20220623$ qmake -v
QMake version 3.1
Using Qt version 5.12.6 in /home/dwinvm/dwinvm/buildroot-A40i-QT-sdk-20220623/sysroot/usr/local/Qt-5.12.6/lib
dwinvm@dwinvm-VMware-Virtual-Platform:~/dwinvm/buildroot-A40i-QT-sdk-20220623$

```

Chapter 8 Cross-Compiling Qt Project Files

This chapter describes the configuration and procedure for cross-compiling with Qt Creator in Ubuntu.

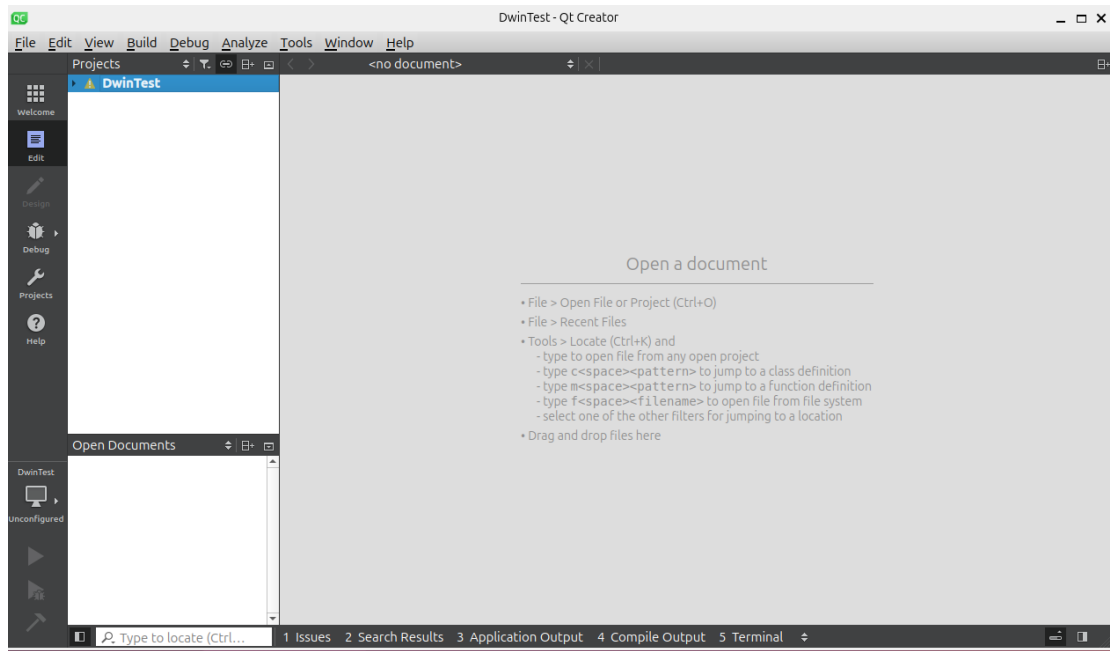
8.1 Installing Qt Creator in Ubuntu

With an active network connection, enter **sudo apt install qtcreator** in the Ubuntu terminal to install the latest version of Qt Creator.

```
qmake-v: command not found
dwinvm@dwinvm-Virtual-Platform:~/dwinvm/buildroot-T113-QT5_12_5-sdk-soft$ qmake -v
dwinvm@dwinvm-Virtual-Platform:~/dwinvm/buildroot-T113-QT5_12_5-sdk-soft$ cd
dwinvm@dwinvm-Virtual-Platform:~$ sudo apt install qtcreator
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  assistant-q6 binutils binutils-common binutils-x86-64-linux-gnu clang-18 clang-tidy clang-tidy-18 clang-tools-18 clangd clangd-18 designer-q6 icu-devtools
  lib32gcc-s1 lib32stdc++6 libasan8 libb2-1 libbinutils libc6-i386 libclang-common-18-dev libclang-cpp15t64 libclang-rt-18-dev libclang1-15t64 libctf-nobfd0
  libctf0 libdouble-conversion3 libffi-dev libgc1 libgcc-13-dev libgprofng0 libgrpc++1.51t64 libgrpc29t64 libgumbo2 libhwasan0 libicu-dev libitm1 liblitehtml0t64
  libllvm15t64 liblsan0 libmd4c0 libmng2 libncurses-dev libobjc-13-dev libobjc4 libpcre2-16-0 libpfm4 libprotoc32t64 libqt6concurrent6t64 libqt6core5compat6
  libqt6core6t64 libqt6dbus6t64 libqt6designer6 libqt6designercomponents6 libqt6gui6t64 libqt6help6 libqt6jsonrpc6 libqt6languageserver6 libqt6network6t64
  libqt6opengl6t64 libqt6openglwidgets6t64 libqt6printsupport6t64 libqt6qml6 libqt6qmlcompiler6 libqt6qmlmodels6 libqt6qmlworkerscript6 libqt6quick6
  libqt6quickcontrols2-6 libqt6quickcontrols2impl6 libqt6quicklayouts6 libqt6quicktemplates2-6 libqt6quicktest6 libqt6quickwidgets6 libqt6serialport6
  libqt6sql6-sqlite libqt6sql6t64 libqt6svg6 libqt6svgwidgets6 libqt6test6t64 libqt6uitools6 libqt6waylandclient6 libqt6waylandcompositor6
  libqt6waylandeglclienthintegration6 libqt6waylandeglcompositorhintegration6 libqt6widgets6t64 libqt6wslshellintegration6 libqt6xml6t64 libquadmth0 libre2-1
  librsframe1 libstdc++-13-dev libstdc++6 libtsan2 libubsan1 libxml2-dev libyaml-cpp0.8 libz3-4 libz3-dev linguist-q6 llvm-18 llvm-18-dev llvm-18-linker-tools
  llvm-18-runtime llvm-18-tools make qdbus-q6 qml-q6 qml6-module-qtqml-models qml6-module-qtqml-workerscript qml6-module-qtquick qml6-module-qtquick-controls
  qml6-module-qtquick-layouts qt6-base-dev-tools qt6-declarative-dev-tools qt6-documentation-tools qt6-gtk-platformtheme qt6-image-formats-plugins qt6-l10n-tools
  qt6-qmltooling-plugins qt6-qpas-plugins qt6-tools-dev-tools qt6-translations-l10n qt6-wayland qtcreator-data qtcreator-doc
Suggested packages:
  binutils-doc gprofng-gui clang-18-doc wasi-libc icu-doc ncurses-doc libstdc++-13-doc pkg-config llvm-18-doc make-doc cmake clazy g++ git meson subversion
  valgrind python3-pylsp
The following NEW packages will be installed:
  assistant-q6 binutils binutils-common binutils-x86-64-linux-gnu clang-18 clang-tidy clang-tidy-18 clang-tools-18 clangd clangd-18 designer-q6 icu-devtools
  lib32gcc-s1 lib32stdc++6 libasan8 libb2-1 libbinutils libc6-i386 libclang-common-18-dev libclang-cpp15t64 libclang-rt-18-dev libclang1-15t64 libctf-nobfd0
  libctf0 libdouble-conversion3 libffi-dev libgc1 libgcc-13-dev libgprofng0 libgrpc++1.51t64 libgrpc29t64 libgumbo2 libhwasan0 libicu-dev libitm1 liblitehtml0t64
  libllvm15t64 liblsan0 libmd4c0 libmng2 libncurses-dev libobjc-13-dev libobjc4 libpcre2-16-0 libpfm4 libprotoc32t64 libqt6concurrent6t64 libqt6core5compat6
  libqt6core6t64 libqt6dbus6t64 libqt6designer6 libqt6designercomponents6 libqt6gui6t64 libqt6help6 libqt6jsonrpc6 libqt6languageserver6 libqt6network6t64
  libqt6opengl6t64 libqt6openglwidgets6t64 libqt6printsupport6t64 libqt6qml6 libqt6qmlcompiler6 libqt6qmlmodels6 libqt6qmlworkerscript6 libqt6quick6
  libqt6quickcontrols2-6 libqt6quickcontrols2impl6 libqt6quicklayouts6 libqt6quicktemplates2-6 libqt6quicktest6 libqt6quickwidgets6 libqt6serialport6
  libqt6sql6-sqlite libqt6sql6t64 libqt6svg6 libqt6svgwidgets6 libqt6test6t64 libqt6uitools6 libqt6waylandclient6 libqt6waylandcompositor6
  libqt6waylandeglclienthintegration6 libqt6waylandeglcompositorhintegration6 libqt6widgets6t64 libqt6wslshellintegration6 libqt6xml6t64 libquadmth0 libre2-1
  librsframe1 libstdc++-13-dev libstdc++6 libtsan2 libubsan1 libxml2-dev libyaml-cpp0.8 libz3-4 libz3-dev linguist-q6 llvm-18 llvm-18-dev llvm-18-linker-tools
  llvm-18-runtime llvm-18-tools make qdbus-q6 qml-q6 qml6-module-qtqml-models qml6-module-qtqml-workerscript qml6-module-qtquick qml6-module-qtquick-controls
  qml6-module-qtquick-layouts qt6-base-dev-tools qt6-declarative-dev-tools qt6-documentation-tools qt6-gtk-platformtheme qt6-image-formats-plugins qt6-l10n-tools
  qt6-qmltooling-plugins qt6-qpas-plugins qt6-tools-dev-tools qt6-translations-l10n qt6-wayland qtcreator-data qtcreator-doc
```

8.2 Opening the Project

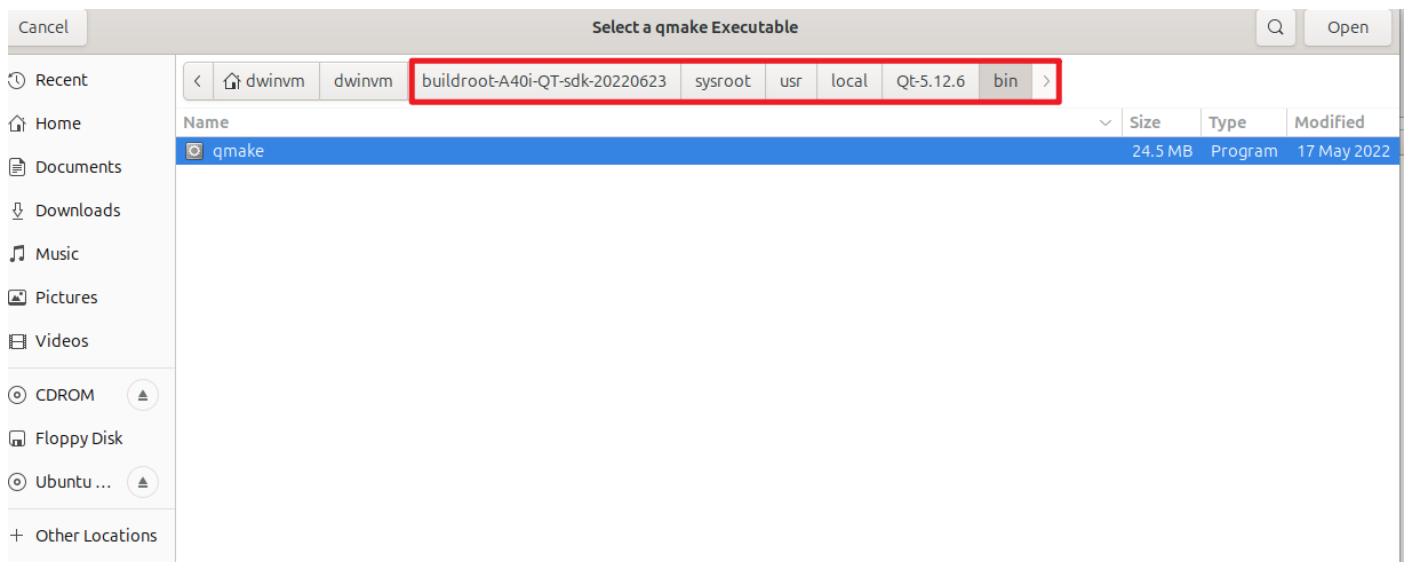
In the Qt Creator interface, select “File” and open the DwinTest project DEMO.

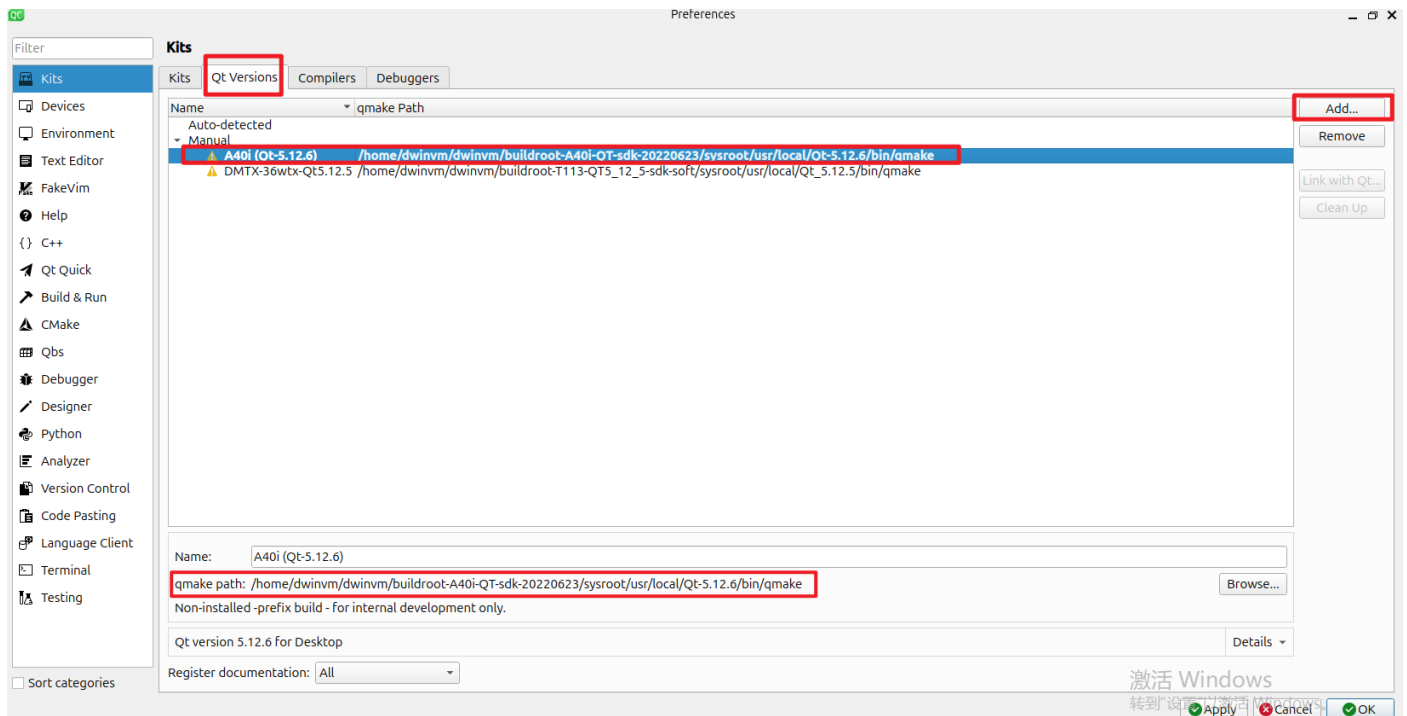


8.3 Setting the Qmake Path

Select “Edit” → “Preferences” → “Qt Versions” → “Add” .

Qmake is located in the buildroot-A40i-QT-sdk-20220623.tar.gz package at sysroot/usr/local/Qt-5.12.6/bin/.

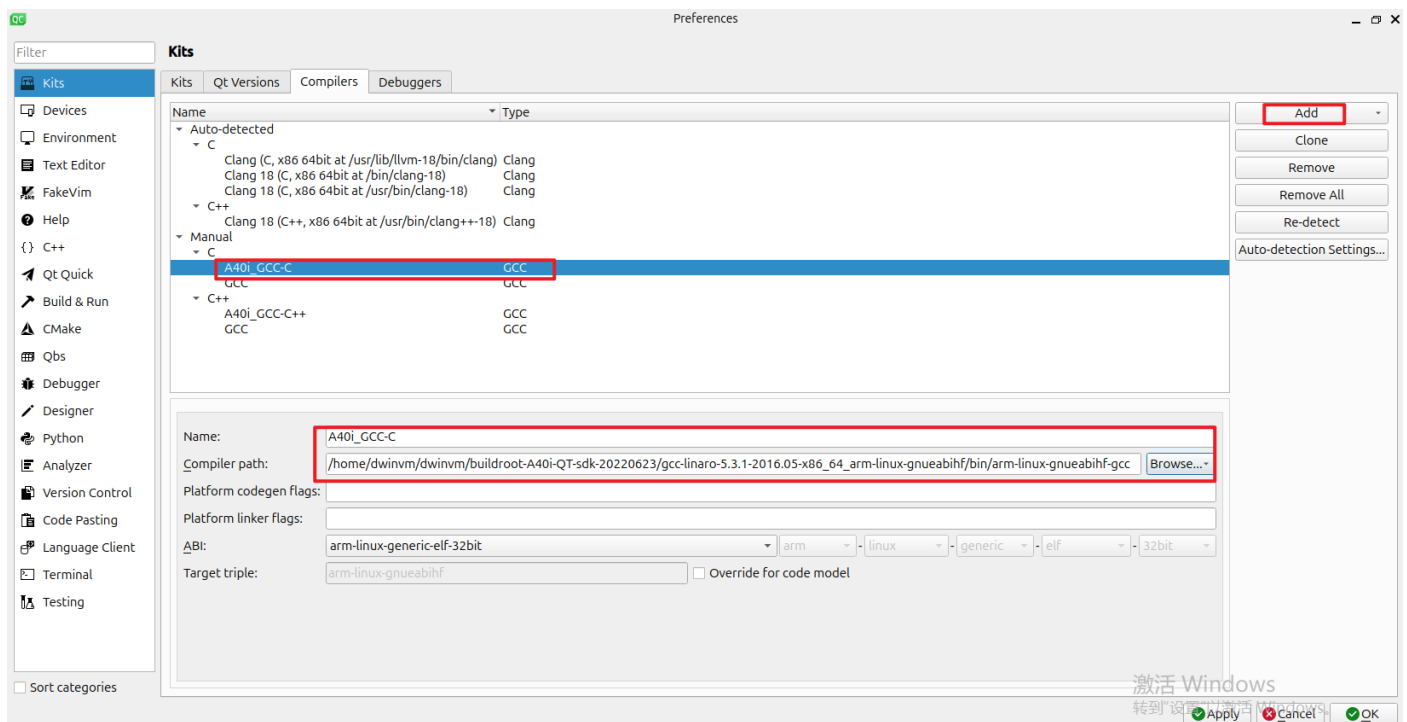




8.4 Configuring the Compiler Toolchain

Select “Compilers”. It is recommended to add both C and C++ compilers.

- Click Add GCC-C.

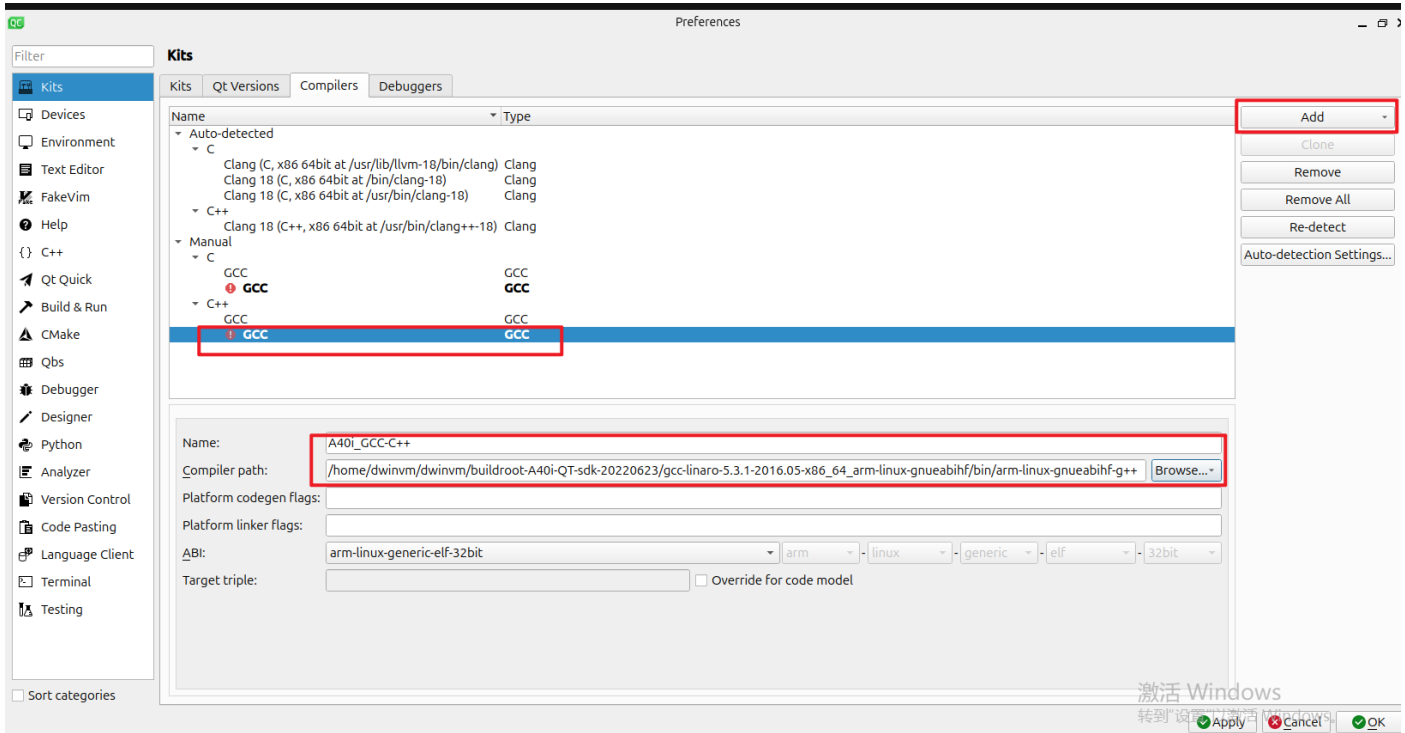


Headquarter: F18, Bldg 5, Jia 2, N. Xisanhuan Rd, Haidian Dist, Beijing 100081

Factory: No. 8, Venture Ave, Zhangjiang Innovation Park, Taoyuan, Hunan

Web: www.dwin-global.com | **Tel:** (86) 400 018 9008

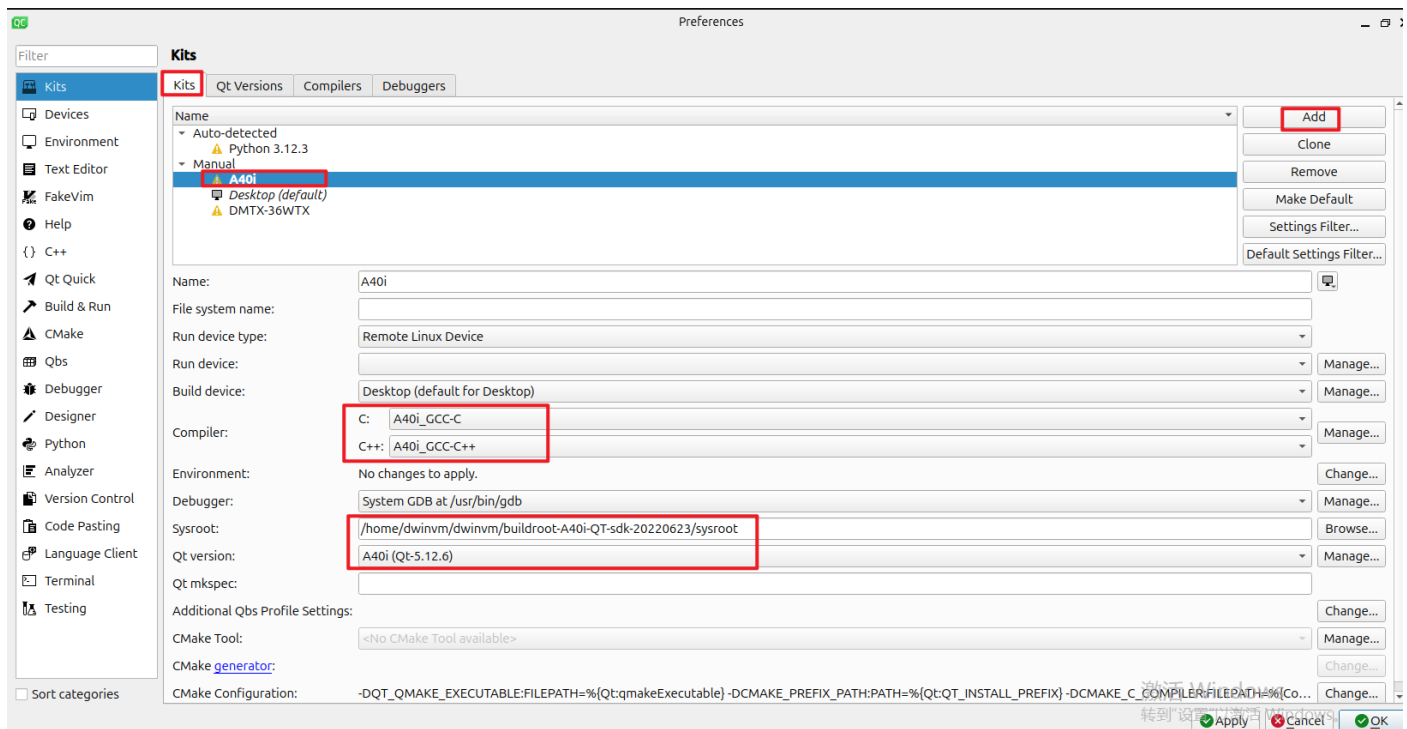
- Click Browse on the right side of the compiler path and select the compiler ending in gcc in the bin directory of the toolchain package.
- Repeat the above two steps and click Add-GCC-C++.
- In the bin directory of the toolchain folder, select the compiler ending in g++.



8.5 Configuring Kits

Configure the Kits page according to the original figure.

The sysroot folder is generally located in the buildroot-A40i-QT-sdk-20220623 folder under the toolchain directory.

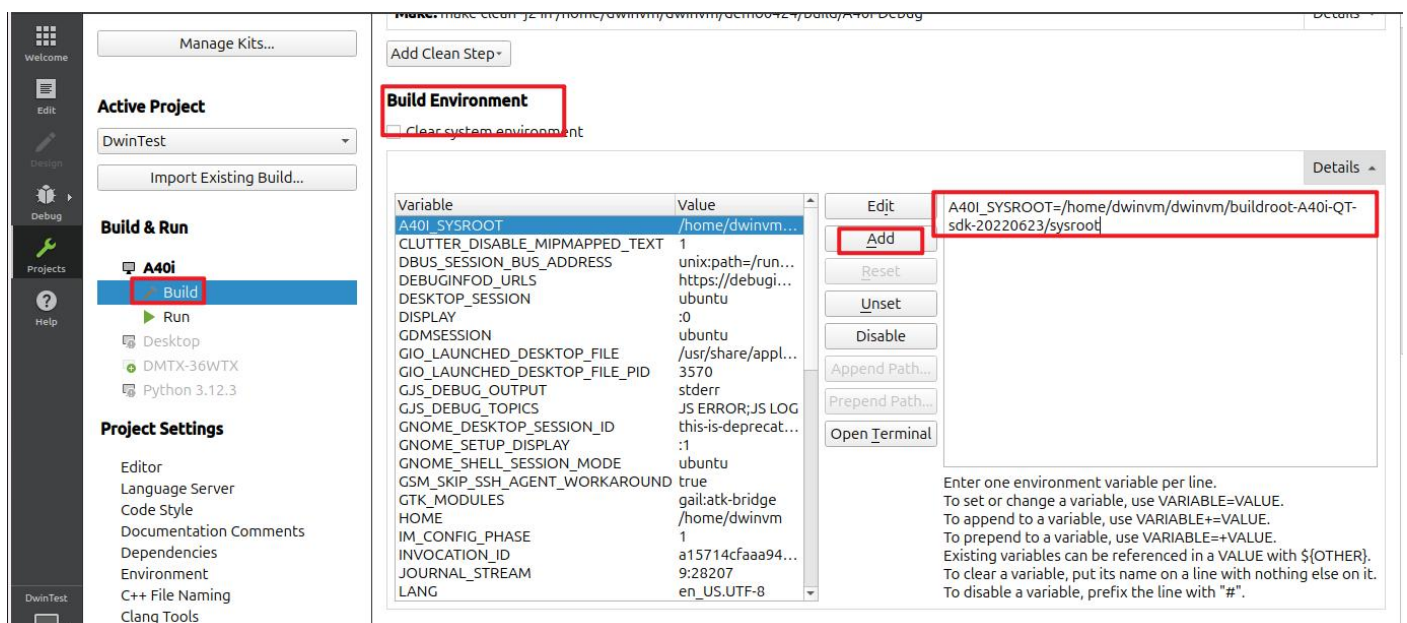


8.6 Configuring Environment Variables

Open the project, select “Build Environment” → “Add”, and add the following variable:

Variable name: A40I_SYSROOT

Variable value: the sysroot directory in the buildroot-A40i-QT-sdk-20220623 folder



Headquarter: F18, Bldg 5, Jia 2, N. Xisanhuan Rd, Haidian Dist, Beijing 100081

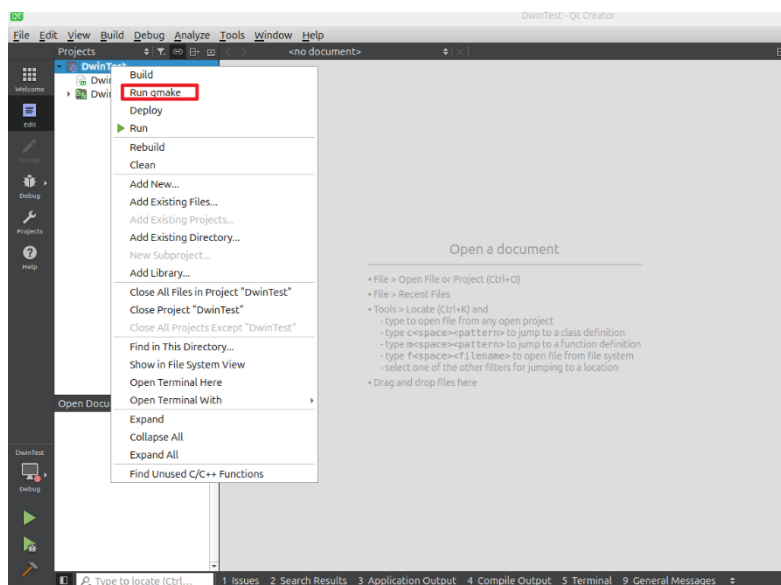
Factory: No. 8, Venture Ave, Zhangjiang Innovation Park, Taoyuan, Hunan

Web: www.dwin-global.com | **Tel:** (86) 400 018 9008

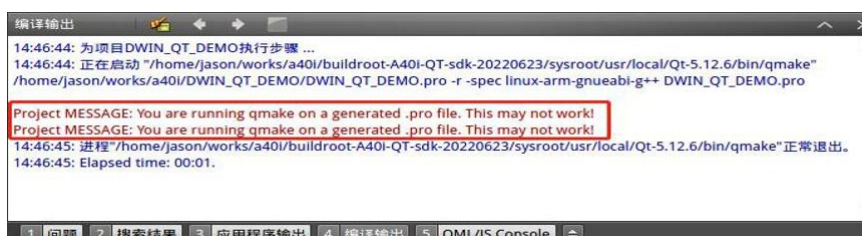
8.7 Building and Downloading/Deploying the Project

8.7.1 Running qmake

Click “Edit” → select the DEMO project → right-click → select “Run qmake” from the menu.

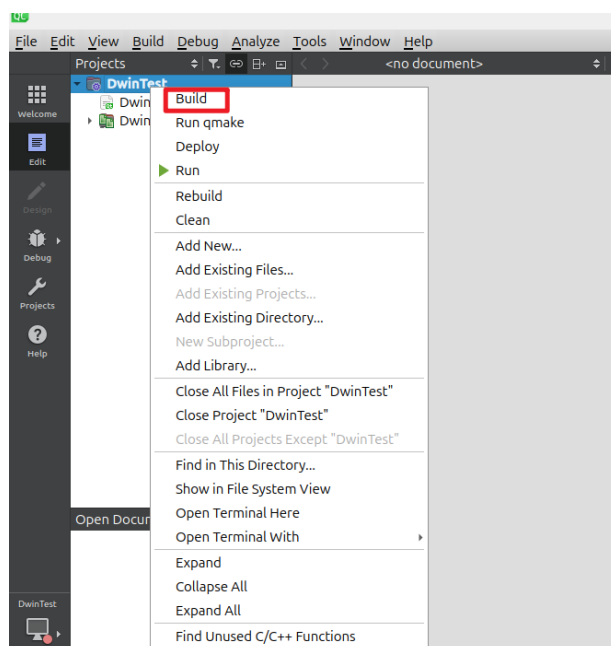


Example of successful qmake execution (the red text does not affect the result):

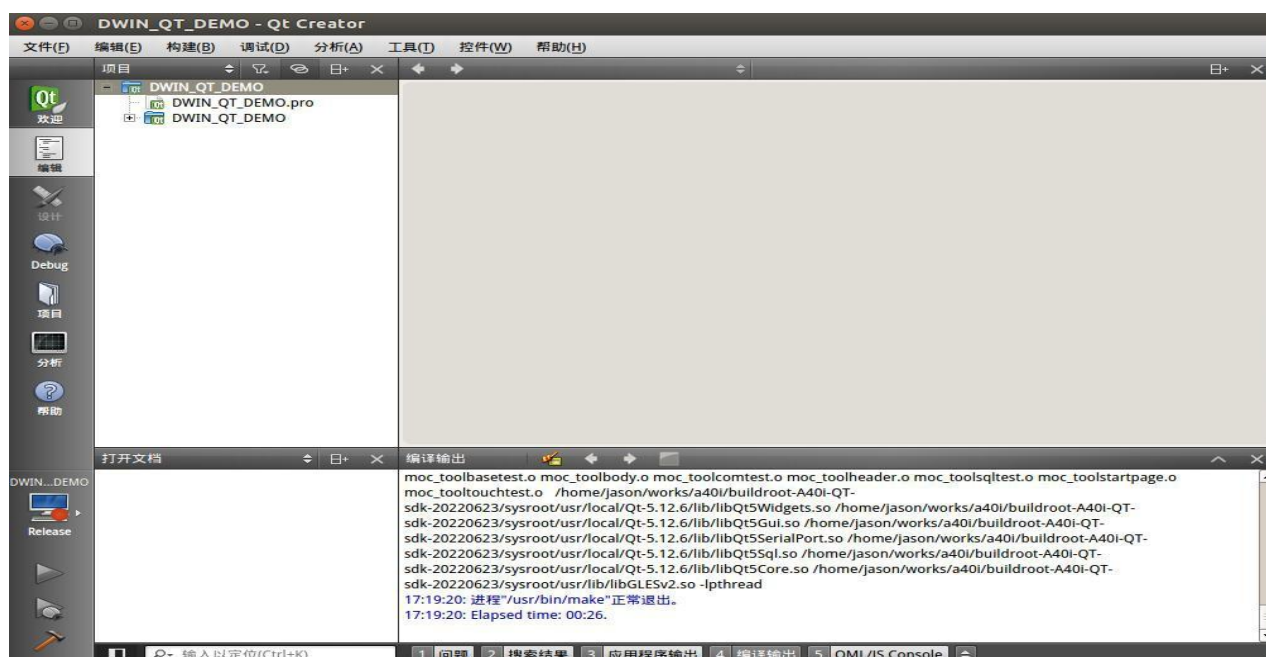


8.7.2 Build

Edit → select the project → right-click → Build.



Build completed.



At this point, the target file has been generated in the project directory and can be copied to the display for execution.



8.7.3 Downloading/Deploying via USB Flash Drive

- Insert a USB flash drive into the computer and copy the executable file DwinTest to the USB flash drive.
- Power on the display and connect the computer using either a serial cable or Ethernet cable.
- Insert the USB flash drive into the USB port of the display.
- Launch MobaXterm, connect to the display, and enter **root**.
- Enter: **cd /mnt/usb/ (press Tab)**, enter the usb folder, and select the sdaX folder.
- Copy or move the target file to the target directory. You may use any custom folder to avoid clutter caused by too many files later. Enter: **cp /mnt/usb/sda1/DwinTest /**
- Run the DEMO with: **vi /etc/init.d/runqt**
- Note: The path may differ between firmware versions. If runqt does not exist in vi /etc/init.d/runqt, try **vi /etc/runqt**.

```
# vi /etc/init.d/runqt
#!/bin/sh
export LD_LIBRARY_PATH=/usr/local/Qt-5.12.6/lib:$LD_LIBRARY_PATH
export QT_PLUGIN_PATH=/usr/local/Qt-5.12.6/plugins
export QT_QPA_FONTDIR=/usr/local/Qt-5.12.6/fonts
export QT_QPA_PLATFORM_PLUGIN_PATH=$QT_PLUGIN_PATH/platforms
export QT_QPA_PLATFORM="linuxfb:fb=/dev/fb0:size=1024x600:mmssize=169x179"
export PATH=/usr/local/Qt-5.12.6/bin:$PATH
export QTDIR=/usr/local/Qt-5.12.6
export QT_ROOT=/usr/local/Qt-5.12.6
export QT_QPA_GENERIC_PLUGINS=tslib,evdevmouse:/dev/input/event4
export QT_QPA_FB_HIDE_CURSOR=1
export QT_SCREEN_ROTATE=0
#export QT_QPA_USE_DWINTOUCH=1
qtesttool &
~
~
~
~
~
~
```

- Move the cursor to the beginning of the qtesttool line, press i to enter insert mode, enter # to comment out the line, then move the cursor to the end of the line (using the keyboard arrow keys), press Enter, move to the next line, and enter the absolute path to the dwinqtdemo executable followed by a space and &. Press Esc to exit insert mode, enter :wq, and save the changes.

```
# vi /etc/init.d/runqt
#!/bin/sh
export LD_LIBRARY_PATH=/usr/local/Qt-5.12.6/lib:$LD_LIBRARY_PATH
export QT_PLUGIN_PATH=/usr/local/Qt-5.12.6/plugins
export QT_QPA_FONTDIR=/usr/local/Qt-5.12.6/fonts
export QT_QPA_PLATFORM_PLUGIN_PATH=$QT_PLUGIN_PATH/platforms
export QT_QPA_PLATFORM="linuxfb:fb=/dev/fb0:size=1024x600:mmssize=169x179"
export PATH=/usr/local/Qt-5.12.6/bin:$PATH
export QTDIR=/usr/local/Qt-5.12.6
export QT_ROOT=/usr/local/Qt-5.12.6
export QT_QPA_GENERIC_PLUGINS=tslib,evdevmouse:/dev/input/event4
export QT_QPA_FB_HIDE_CURSOR=1
export QT_SCREEN_ROTATE=0
#export QT_QPA_USE_DWINTOUCH=1
qtesttool &
~
~
~
~
~
~
```

```
#!/bin/sh
export LD_LIBRARY_PATH=/usr/local/Qt-5.12.6/lib:$LD_LIBRARY_PATH
export QT_PLUGIN_PATH=/usr/local/Qt-5.12.6/plugins
export QT_QPA_FONTDIR=/usr/local/Qt-5.12.6/fonts
export QT_QPA_PLATFORM_PLUGIN_PATH=$QT_PLUGIN_PATH/platforms
export QT_QPA_PLATFORM="linuxfb:fb=/dev/fb0:size=1024x600:mmsize=169x179"
export PATH=/usr/local/Qt-5.12.6/bin:$PATH
export QTDIR=/usr/local/Qt-5.12.6
export QT_ROOT=/usr/local/Qt-5.12.6
export QT_QPA_GENERIC_PLUGINS=tslib,evdevmouse:/dev/input/event4
export QT_QPA_FB_HIDE_CURSOR=1
export QT_SCREEN_ROTATE=0
#export QT_QPA_USE_DWINTOUCH=1
#qttesttool &
/DwinTest &
```

- The program can then be launched by running runqt.

```
# cd /etc/init.d/
# ./run
runhmi      runqt      runupdate
# ./runqt
```

Chapter 9 Cross-Compiling LVGL Project Files

This chapter describes how to build an LVGL project by cross-compilation in Ubuntu and deploy it to a 35 Series Linux display.

9.1 Software Requirements

Supported LVGL version: 9.1.0

Download: <https://github.com/lvgl/lvgl/releases/tag/v9.1.0>

LVGL documentation: <https://docs.lvgl.io/master/>

Build environment: Ubuntu 14.04 and other versions. For environment setup, refer to Chapter 7.

Example resources: LvglDemo, buildroot-A40i-LVGL9_1_0-sdk-20241219.tar, lvgl_demo_source

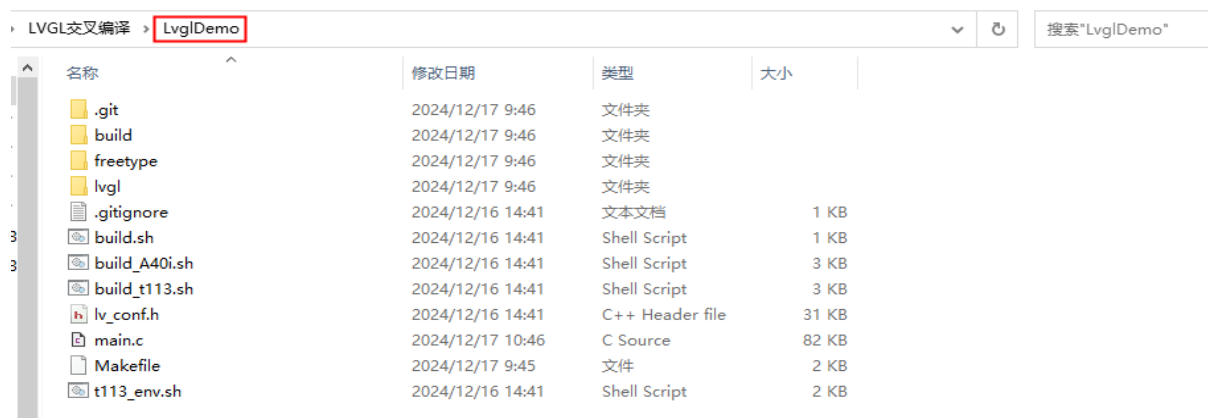
9.2 Hardware

Currently, only 35 Series capacitive touch screen models are supported.

9.3 Cross-Compilation

9.3.1 Pre-Compilation Preparation

Open main.c in LvglDemo and search for main(void). In the input device initialization section, select A40i for this example because the 35 Series model uses the A40i platform.



名称	修改日期	类型	大小
.git	2024/12/17 9:46	文件夹	
build	2024/12/17 9:46	文件夹	
freetype	2024/12/17 9:46	文件夹	
lvgl	2024/12/17 9:46	文件夹	
.gitignore	2024/12/16 14:41	文本文档	1 KB
build.sh	2024/12/16 14:41	Shell Script	1 KB
build_A40i.sh	2024/12/16 14:41	Shell Script	3 KB
build_t113.sh	2024/12/16 14:41	Shell Script	3 KB
lv_conf.h	2024/12/16 14:41	C++ Header file	31 KB
main.c	2024/12/17 10:46	C Source	82 KB
Makefile	2024/12/17 9:45	文件	2 KB
t113_env.sh	2024/12/16 14:41	Shell Script	2 KB

Headquarter: F18, Bldg 5, Jia 2, N. Xisanhuan Rd, Haidian Dist, Beijing 100081

Factory: No. 8, Venture Ave, Zhangjiang Innovation Park, Taoyuan, Hunan

Web: www.dwin-global.com | **Tel:** (86) 400 018 9008

```

1959 int main(void)
1960 {
1961     //Informations before all
1962     #if 1
1963         LV_LOG_USER("\n\ntlvg1_demo start");
1964         LV_LOG_USER("\n\t sizes: char(%x(-1:%d)), short(%x), int(%x), long(%x), size_t(%x)"
1965                     ", sizeof(char), (char)-1, sizeof(short), sizeof(int), sizeof(long), sizeof(size_t));");
1966     #endif//
1967
1968     //LVGL Init
1969     lv_init();
1970     LV_LOG_USER("lv_init done.");//
1971
1972     //Linux frame buffer device init
1973     lv_display_t * disp = lv_linux_fbdev_create();
1974     lv_linux_fbdev_set_file(disp, "/dev/fb0");
1975
1976     LV_LOG_USER("lv display init done.");//
1977
1978     //Input device init
1979
1980     //const char inputFile[] = "/dev/input/event2";//R11
1981     const char inputFile[] = "/dev/input/event4";//A40i ←
1982     //const char inputFile[] = "/dev/input/event1";//t113
1983     lv_evdev_create(LV_INDEV_TYPE_POINTER, inputFile);
1984
1985     LV_LOG_USER("lv indev init done. (%s)", inputFile);//

```

9.3.2 Compilation Procedure

- Copy the LvglDemo folder and buildroot-A40i-LVGL9 1 0-sdk-20241219.tar toolchain into Ubuntu.

lvgl_demo_source	2024/12/19 9:14	文件夹	
LvglIDemo	2024/12/24 16:42	文件夹	
压缩存档文件夹			
buildroot-A40i-LVGL9 1 0-sdk-20241219.tar	2024/12/25 13:34	压缩存档文件夹	2,858,715 KB

- Open Terminal and extract the SDK (toolchain) buildroot-A40i-LVGL9 1 0-sdk-20241219.tar using:

```
tar -xvf buildroot-A40i-LVGL9_1_0-sdk-20241219.tar
```

```
dwinvm@ubuntu:~$ tar -xvf buildroot-A40i-LVGL9_1_0-sdk-20241219.tar
```

- After extraction is complete, run the script, set the environment variables, open the project file, and run make to generate the executable.

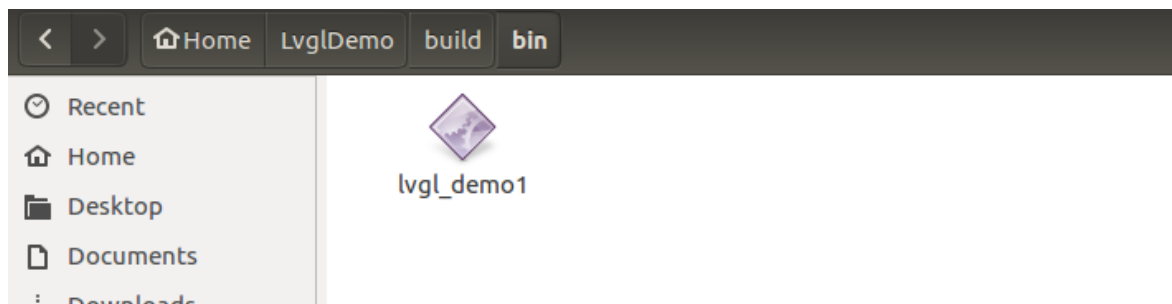
```
source buildroot-A40i-LVGL9_1_0-sdk-20241219/env-setup
cd LvglDemo/
make
```

```
buildroot-A40i-LVGL9_1_0-sdk-20241219/sysroot/usr/share/zsh/site-functions/
buildroot-A40i-LVGL9_1_0-sdk-20241219/sysroot/usr/share/zsh/site-functions/_pulseaudio
buildroot-A40i-LVGL9_1_0-sdk-20241219/sysroot/var/
buildroot-A40i-LVGL9_1_0-sdk-20241219/sysroot/var/cache/
buildroot-A40i-LVGL9_1_0-sdk-20241219/sysroot/var/cache/fontconfig/
buildroot-A40i-LVGL9_1_0-sdk-20241219/sysroot/var/lib/
buildroot-A40i-LVGL9_1_0-sdk-20241219/sysroot/var/lib/alsa/
buildroot-A40i-LVGL9_1_0-sdk-20241219/sysroot/var/lib/dbus/
buildroot-A40i-LVGL9_1_0-sdk-20241219/sysroot/var/run/
buildroot-A40i-LVGL9_1_0-sdk-20241219/sysroot/var/run/dbus/
dwinvm@ubuntu:~$ source buildroot-A40i-LVGL9_1_0-sdk-20241219/env-setup
dwinvm@ubuntu:~$ cd LvglDemo
dwinvm@ubuntu:~/LvglDemo$ make
```

- The executable lvgl_demo is generated in LvglDemo/build/bin.

```
cd build/bin/
```

```
dwinvm@ubuntu:~/LvglDemo$ cd build/bin
dwinvm@ubuntu:~/LvglDemo/build/bin$ ls
lvgl_demo1
dwinvm@ubuntu:~/LvglDemo/build/bin$
```



9.3.3 Running the Test

- Copy the lvgl_demo executable generated in the previous step and the provided lvgl_demo_source folder to a USB flash drive, then insert the USB flash drive into the USB port of the display.

 lvgl_demo_source	2024/12/19 9:14	文件夹
 LvglDemo	2024/12/24 16:42	文件夹
▼ 压缩存档文件夹		
 buildroot-A40i-LVGL9_1_0-sdk-20241219.tar	2024/12/25 13:34	压缩存档文件夹 2,858,715 KB

- Connect the display and computer with an Ethernet cable. For details, refer to the Ethernet Telnet connection procedure. Using MobaXterm, copy the fonts and images folders from the USB flash drive to /var/setting/lvgl.

```

3. 192.168.10.202
? MobaXterm Personal Edition v23.3 ?
(SSH client, X server and network tools)

> Telnet session to 192.168.10.202
> Your DISPLAY is set to 169.254.125.1:0.0
> For more info, ctrl+click on help or visit our website.

kunos login: rooy
Password:
Login incorrect
kunos login: root
Password: 密码Dwin123, 输入不显示

```

cd /mnt/usb/sdaXXX/lvgl_demo_source (sdaXXX is the mount location of the USB flash drive on the display; enter cd /mnt/usb/ and press Tab to obtain it.)

cp -rf fonts/ /var/setting/lvgl/

cp -rf images/ /var/setting/lvgl/

```

# cd /mnt/usb/sda1/lvgl_demo_source/
# ls
fonts  images
# cp -rf fonts/ /var/setting/lvgl/
# cp -rf images/ /var/setting/lvgl/
# cd /var/setting/lvgl/
# ls
fonts  images
#

```

Copy the executable to the ~ directory via the USB flash drive and run the program for testing.

cd /mnt/usb/sda1/ (sda1 通过 Tab 键获得)

cp lvgl_demo ~

cd ~

./lvgl_demo

```
# cd /mnt/usb/sda1/
# cp lvgl_demo ~
# cd ~
# ls
lvgl_demo
# ./lvgl_demo
[User] (0.000, +0) main:
lvgl_demo start main.c:1963
[User] (0.000, +0) main:
sizes: char(1(-1:255)), short(2), int(4), long(4), size_t(4) main.c:1965
[Info] (0.000, +0) lv_init: begin lv_init.c:139
[User] (0.000, +0) main: lv_init done, main.c:1970
[Info] (4293318.922, +4293318922) lv_obj_create: begin lv_obj.c:100
[Info] (4293318.922, +0) lv_obj_create: begin lv_obj.c:100
[Info] (4293318.922, +0) lv_obj_create: begin lv_obj.c:100
[Info] (4293318.922, +0) lv_obj_create: begin lv_obj.c:100
[Info] (4293318.922, +0) lv_linux_fbdev_set_file: The framebuffer device was opened successfully lv_linux_fbdev.c:134
[Info] (4293318.922, +0) lv_linux_fbdev_set_file: 800x480, 32bpp lv_linux_fbdev.c:180
[Info] (4293318.923, +1) lv_linux_fbdev_set_file: The framebuffer device was mapped to memory successfully lv_linux_fbdev.c:195
```

The DEMO is now installed on the display.

9.4 Notes

9.4.1 Makefile

If you develop your own application based on the DEMO, modify the Makefile according to your project requirements.

```
Makefile
1 #
2 # Makefile
3 #
4
5 CC           ?= gcc
6 LVGL_DIR_NAME ?= lvgl ← 1、LVGL源码相对目录
7 LVGL_DIR     ?= .
8 include $(LVGL_DIR)/lvgl/lvgl.mk
9
10 #Flags
11 #WARNINGS := -Wall -Wshadow -Wundef -Wno-unused-function -Wno-error=strict-prototypes -Wpointer-arith -Wno-error=pedantic \
12             -fno-strict-aliasing -Wno-error=cpp -Wmaybe-uninitialized -Wno-unused-parameter -Wno-missing-field-initializers \
13             -Wno-format-nonliteral -Wno-cast-qual -Wunreachable-code -Wno-switch-default -Wreturn-type -Wstack-usage=2048 \
14             -Wno-sign-compare -Wdouble-promotion -Wclobbered -Wdeprecated -Wempty-body -Wtype-limits -Wshift-negative-value \
15             -Wno-unused-value -Wuninitialized -Wextra -Wno-ignored-qualifiers -Wsizeof-pointer-memaccess -Wformat-security \
16             -Wmultichar
17
18 #FLAGS_FOR_C := $(CFLAGS) -std=c99 -g0 -I$(LVGL_DIR)/$(WARNINGS) \
19             -Wmissing-prototypes -Wno-error=missing-prototypes -Wno-discarded-qualifiers
20 FLAGS_FOR_CXX := $(CXXFLAGS) -std=c++11 -g0 -I$(LVGL_DIR)/$(WARNINGS)
21
22 #Build
23 BIN = lvgl_demo ← 2、项目名称
24 BUILD_DIR = ./build
25 BUILD_OBJ_DIR = $(BUILD_DIR)/obj
26 BUILD_BIN_DIR = $(BUILD_DIR)/bin
27
28 #Collect the files to compile
29 MAINSRC = ./main.c ← 3、自己编写的代码，文件名
30 CSRCS += $(MAINSRC)
31
32 #Objects
```

9.4.2 Third-Party Libraries

If you need to load a third-party library (such as freetype in the example project), open lvgl.mk in the lvgl directory

and add the source path, header files, and other required information.

```
lvgl.mk  x
1  LVGL_PATH ?= ${shell pwd}/lvgl
2  FREETYPE_PATH ?= ${shell pwd}/freetype ← freetype源码路径
3
4  ASRCS += $(shell find $(LVGL_PATH)/src -type f -name '*.S')
5  CSRCS += $(shell find $(LVGL_PATH)/src -type f -name '*.c')
6  CSRCS += $(shell find $(LVGL_PATH)/demos -type f -name '*.c')
7  # CSRCS += $(shell find $(LVGL_PATH)/examples -type f -name '*.c')
8  CXXEXT := .cpp
9  CXXSRCS += $(shell find $(LVGL_PATH)/src -type f -name '*${CXXEXT}')
10
11  AFLAGS += "-I$(LVGL_PATH)"
12  CFLAGS += "-I$(LVGL_PATH)"
13  CXXFLAGS += "-I$(LVGL_PATH)"
14
15  # FreeType custom configuration header file
16  CFLAGS += -DFT2_BUILD_LIBRARY
17  CFLAGS += -DFT_CONFIG_MODULES_H=\<lvgl/src/libs/freetype/ftmodule.h> ← 头文件
18  CFLAGS += -DFT_CONFIG_OPTIONS_H=\<lvgl/src/libs/freetype/ftoption.h> ←
19
20  # FreeType include path
21  CFLAGS += -I"${FREETYPE_PATH}/include/" ← include路径
22
23  # FreeType C source file ← C source路径
24  FT_CSRCS += ${FREETYPE_PATH}/src/base/ftbase.c
25  FT_CSRCS += ${FREETYPE_PATH}/src/base/ftmm.c
26  FT_CSRCS += ${FREETYPE_PATH}/src/base/ftbitmap.c
27  FT_CSRCS += ${FREETYPE_PATH}/src/base/ftdebug.c
28  FT_CSRCS += ${FREETYPE_PATH}/src/base/ftglyph.c
29  FT_CSRCS += ${FREETYPE_PATH}/src/base/ftinit.c
30  FT_CSRCS += ${FREETYPE_PATH}/src/cache/ftcache.c
31  FT_CSRCS += ${FREETYPE_PATH}/src/gzip/ftgzip.c
32  FT_CSRCS += ${FREETYPE_PATH}/src/sfnt/sfnt.c
33  FT_CSRCS += ${FREETYPE_PATH}/src/smooth/smooth.c
34  FT_CSRCS += ${FREETYPE_PATH}/src/truetype/truetype.c
35  CSRCS += $(FT_CSRCS)
```

Chapter 10 Revision History

Version	Date	Description	Editor
00	2022.09.26	Initial release	Yu Yihe
01	2022.09.29	Added qtdemo project and webviewtest browser project example	Yu Yihe
02	2022.10.13	Added application update package upgrade guide	Yu Yihe
03	2023.04.25	Added boot logo modification example	Yu Yihe
04	2024.02.19	Added limodbus example	Chen Yan
05	2024.03.18	Added brightness adjustment and buzzer switch examples	Chen Yan
06	2024.03.27	Added instructions for the dwinqtdemo execution path	Chen Yan
07	2024.08.12	Added Qt Creator cross-compilation configuration and added Chapter 1	Chen Yan
08	2025.05.22	Restructured chapters; updated to the new Ubuntu and Qt Creator	Chen Yan
09	2026.09.28	Added LVGL project file compilation configuration	Chen Yan