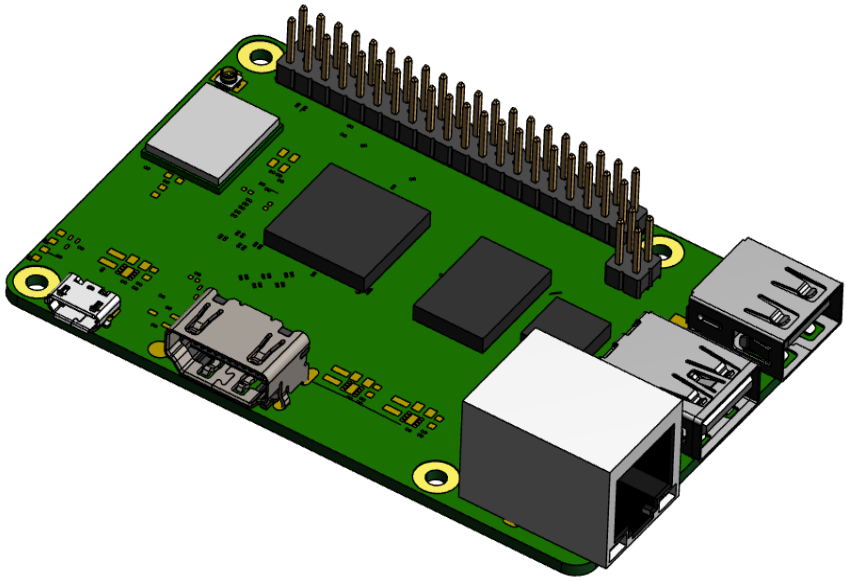


REIMEI1

AN EXCELLENT AND COST-EFFECTIVE SINGLE-BOARD COMPUTER

Shanghai EDA Technology Co.,Ltd
2023-04-10

Copyright Statement

Disclaimers

Shanghai EDA Technology Co., Ltd does not guarantee that the information in this manual is up to date, correct, complete or of high quality. Shanghai EDA Technology Co., Ltd also does not guarantee the further use of this information. If the material or non-material related losses are caused by using or not using the information in this manual, or by using incorrect or incomplete information, as long as it is not proved that it is the intention or negligence of Shanghai EDA Technology Co., Ltd, the liability claim for Shanghai EDA Technology Co., Ltd can be exempted. Shanghai EDA Technology Co., Ltd expressly reserves the right to modify or supplement the contents or part of this manual without special notice.

Contents

1 Product Overview

REIMEI1 is a single-board computer with excellent performance, compactness and high cost performance.

1.1 Target Application

- Multimedia creation
- AI Development
- Developer development
- Smart manufacturing

1.2 Specifications and Parameters

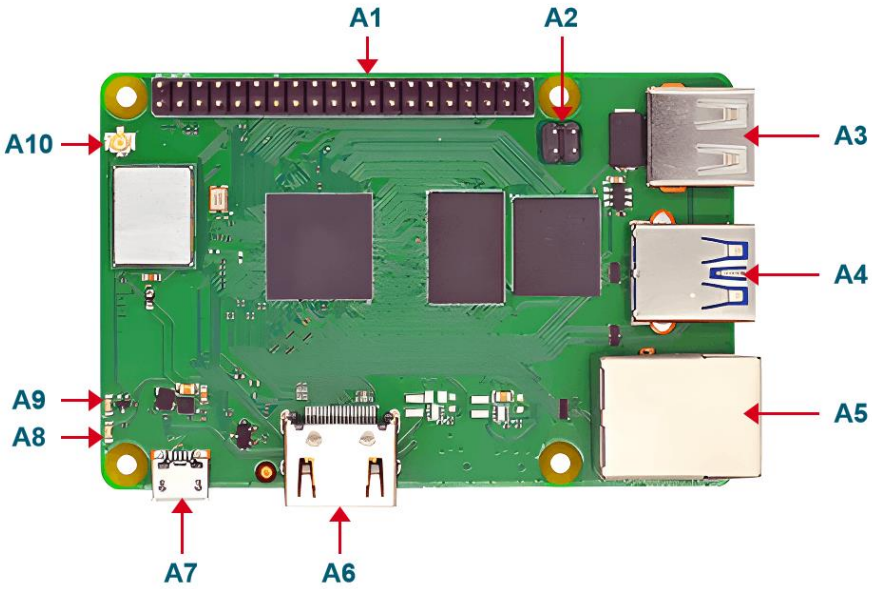
1.2.1 Hardware

Function	Parameters
CPU	Amlogic S905X4 4 core，Cortex-A55 (ARMv8-A)，64 bit，2.0GHz
GPU	ARM Mail-G31MP2
Memory	1GB / 2GB / 4GB option
eMMC	8GB / 16GB / 32GB option
SD card	micro SD card
Ethernet	1x 10/100M Ethernet，support POE HAT
WiFi / Bluetooth	2.4G / 5.8G dual WiFi, bluetooth5.0
HDMI	1x standard HDMI
USB Host	1x USB 2.0 Type A, 1x USB 3.0 Type A
GPIO	28 channels of GPIO are available for users, and some GPIO can be reused as UART, I2C and SPI.
LED	Red (power indicator), green (system status indicator)
Power input	5V@2.5A
Dimensions	85(L) x 56(W) mm
Antenna accessory	Support option WiFi / BT external antenna
Working environment temperature	Option -25 ~ 70°C environment temperature

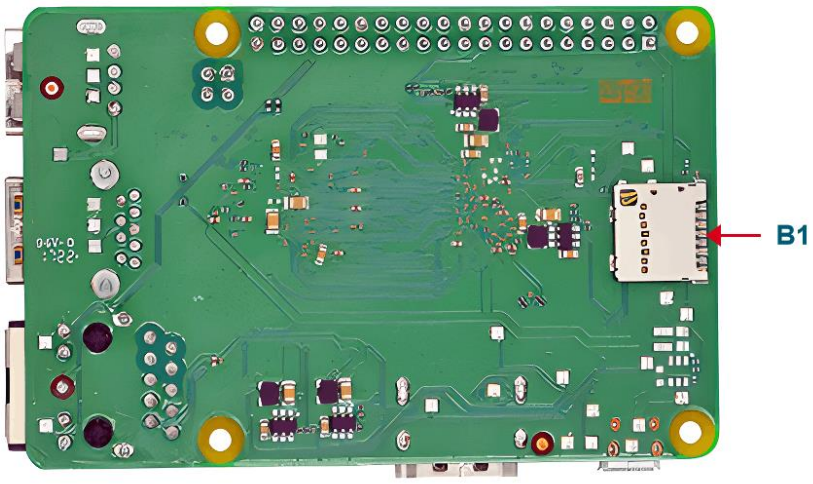
1.2.2 Software

Function	Parameters
OS	Debian 11, 64-bit
Kernel	Linux 5.4.180 64-bit
Video output	HDMI 2.1 to 4Kp75，support CEC、HDR and HDCP 2.2，CVBS
Video decoding	AV1 MP-10 L5.1 up to 4Kx2K @ 60fps
	VP9 Profile-2 up to 4Kx2K @ 60fps
	H.265 HEVC MP-10 @ L5.1 up to 4Kx2K @ 60fps
	AVS2-P2 Profile up to 4Kx2K @ 60fps
	H.264 AVC HP @ L5.1 up to 4Kx2K @ 30fps
	H.264 MVC up to 1080p60
	MPEG-4 ASP @ L5 up to 1080p60 (ISO-14496)
	WMV/VC-1 SP/MP/AP up to 1080p60
	AVS-P16(AVS+) /AVS-P2 JiZhun Profile up to 1080p60
	MPEG-2 MP/HL up to 1080p60 (ISO-13818)
	MPEG-1 MP/HL up to 1080p60 (ISO-11172)
	RealVideo 8/9/10 up to 1080p60
	HDR – HDR10/10+, HLG, Dolby Vision, TCH PRIME
SDK /lib/tool	Mesa Graphics Library with OpenGL ES 3.2, Vulkan 1.0/1.1, and OpenCL 2.0 support
	V4L2 M2M Video Decoder Interface
	QT5 with hardware accelerated Wayland backend
	Gstreamer with hardware decode support

1.3 Functional Layout



Item	Function Description	Item	Function Description
A1	2×20Pin Header	A6	HDMI Type A port
A2	POE header	A7	Micro USB port
A3	USB 2.0	A8	LED red
A4	USB 3.0	A9	LED green
A5	Ethernet RJ45 interface	A10	Antenna IPEX port

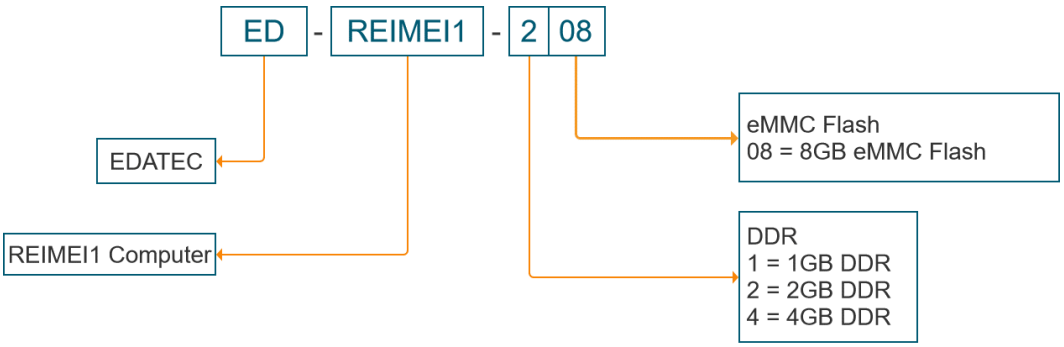


Item	Function Description
B1	Micro SD card slot

1.4 Packing List

- 1x REIMEI 1host
- [option]1x 2.4GHz/5GHz WiFi/BT antenna

1.5 Order Code



Example

Part# : ED-REIMEI1-208
Configration : REIMEI1 Computer
2GB DDR and 8GB eMMC Flash

2 Quick Start

Quick start mainly guides you on how to connect devices, install systems, first-time startup configuration and network configuration.

2.1 Equipment List

- 1x REIMEI 1 host
- 1x 2.4GHz/5GHz WiFi/BT antenna

2.2 Hardware Connection

You need to prepare the following accessories:

- 1x 5V@3A USB micro-B power adapter
- 1x net cable
- 1x HDMI standard cable
- HDMI display
- Mouse
- Keyboard

Hardware connecter:

1. Install the WiFi/BT antenna on the main machine of REIMEI 1.
2. Connect the display and REIMEI 1 with standard HDMI.
3. Insert the network cable
4. Connect the keyboard and mouse
5. Power on, and it is recommended to use 5V@3A USB micro-B power supply.

2.3 First Start

When the device is powered on, it will automatically turn on. When the device is turned on, the red indicator light is constant and the green indicator light flashes. If it is found that the device can't start, the green indicator doesn't flash and the screen doesn't display, it means that the system can't start at this time, probably because there is no system in eMMC. Please refer to Image Download and System Flash to burn eMMC.

2.3.1 Start the Desktop

REIMEI1 supports weston desktop environment and X11 desktop environment.
The image enters the command line by default. If the user wants to start the desktop, you need to execute the following command.

```
sudo systemctl start weston
```

If the user wants to start the X11 desktop, they need to execute the following command.

```
sudo systemctl start lightdm
```

2.3.2 Check the System Version

```
uname -a
```

2.3.3 Setting the Time Zone

Modify the time zone to set the time to China time:

```
sudo timedatectl set-timezone Asia/Shanghai
```

2.3.4 Check the System Partition

Use the lsblk command to get the current partition situation, as well as the partition size and mount path.

```
lsblk
```

2.3.5 Check the Storage Space

Check current storage space information:

```
df -h
```

2.3.6 Setting Hostname

Modify the /etc/hostname file, change the current hostname to the hostname you want to set, and save after the modification:

```
sudo nano /etc/hostname
```

Modify the/etc/hosts file, change the current hostname in hosts to the hostname you want to set, and save after the modification:

```
sudo nano /etc/hosts
```

After the modification is completed, restart the device:

```
sudo reboot
```

2.3.7 Power Off

```
sudo poweroff
```

2.3.8 Restart

```
sudo reboot
```

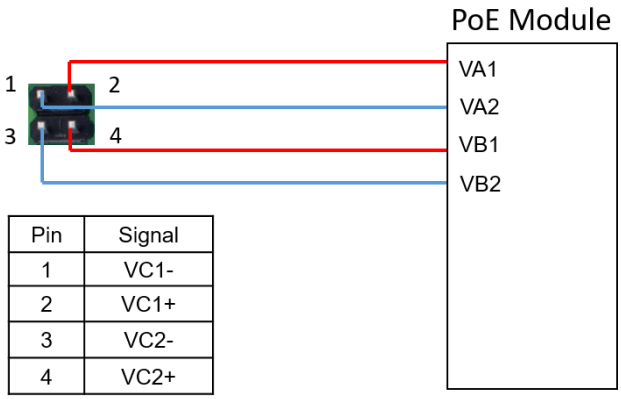
3 Wiring Guide

3.1 IPEX-1

IPEX-1 connector on board REIMEI1 is used for external 2.4GHz/5GHz antenna. Connect the antenna with the female IPEX, facing the male IPEX of the main board, and press down to fix the antenna.

3.2 POE

REIMEI1 supports POE power supply at the network port, but it needs to be equipped with POE HAT module. The wiring diagram with PoE module is as follows:



Pin	Signal
1	VC1-
2	VC1+
3	VC2-
4	VC2+

4 Software Operation Guide

The operating system of REIMEI1 is built on Debian OS, and this chapter contains some basic usage methods of Debian OS.

Debian OS is developed based on the open source system Raspberry Pi OS, with a kernel version of 5.4.180, which supports Weston Desktop (hardware decoding) and QT5 (hardware decoding), and is compatible with most Raspberry Pi OS software ecosystems.

TIP: At present, the system is still in the development stage, and all kinds of raspberry pi system files contained in it have not been removed, including boot images and dts in the /boot directory.

TIP: Raspi-config is still reserved in the system, and users can use raspi-config directly in the system to complete some configurations, such as connecting WiFi and enabling ssh.

4.1 Connect to The Device Using SSH

4.1.1 Enable SSH

Enable SSH automatically at startup:

When the device is started, an empty file named ssh is put into the boot partition before booting, and SSH will be automatically enabled after booting.
To enable SSH on SD card, please refer to the last step of burning system: Flahing SD card.
To enable SSH on eMMC, please refer to the last step of burning system: Flashing eMMC.

Command enables SSH:

```
sudo raspi-config
```

After entering the above command, a command line interface will appear. Configure the third interface. Find SSH and select yes to enable SSH function.

3 Interface Options -> 2 SSH -> Yes

4.1.2 SSH Tool

Windows recommends using putty to realize SSH remote connection.

- Putty Download：Download PuTTY - a free SSH and telnet client for Windows

4.1.3 Get Device IP Address

There are the following ways to get the IP address:

Use Command to check IP

If the device is connected to a monitor and a keyboard, you can use the ifconfig command to check the current IP.

Check router information

Check the IP address of the device listed on the router to find the device. The default hostname of the device is phantom.

Scan with nmap

Install nmap tool: Nmap: the Network Mapper - Free Security Scanner

Execute the following command to scan the network segment 192.168.1.0~255:

```
nmap -sn 192.168.1.0/24
```

After the execution is completed, all scanned devices will be displayed on the screen.

4.1.4 Connecting to The System

```
ssh phantom@<IP>
```

User name ： phantom
Password ： phantom
Port：22

4.2 Connect to The System Through Debugging Serial Port

REIMEI1 has a debugging serial port, which uses USB to serial port (TTL level), and the serial port is connected to pin 6 (GND), pin 8 (TXD) and pin 10 (RXD) of Raspberry 40pin. The other end is connected to the computer USB, and the adapter is found by using the serial port tool. Set the baud rate to 921600,

the data bit to 8 bits, the check bit to none, and the stop bit to 1 bit.

User name：phantom
Password ：phantom

4.3 Using APT tools to Manage Software

The easiest way to manage the installation, upgrade and removal of software is to use Debian's APT (Advanced Packaging Tool). To update the software, you can use the apt tool from the terminal window.

Update deb package list:

```
sudo apt update
```

Install deb package:

```
sudo apt install tree
```

Uninstall deb package:

```
sudo apt remove tree
```

Uninstall the deb package (and delete the corresponding configuration file at the same time)

```
sudo apt purge tree
```

4.4 X11 Desktop Use Introduction

X Window System, usually called X11 or simply X, is a window system with bitmap display, which is widely used in Unix, Unix-like and Linux systems. X11 system is essentially just a toolkit and architecture specification, and it has many different implementations. Among the implementations currently developed according to X11 specification architecture, X.org is the most popular and popular.

X11 system is a hierarchical architecture, which is divided into Server and Client. X Server is responsible for the display of graphical interface and user input, while Client program needs to connect to X Server, then request X Server to draw graphical interface and accept user input from X Server. On the desktop system, X Server and Client programs are often installed on the same machine, and it is basically not felt that it is layered in use.

X11 desktop environment binds various components together to provide common graphical user interface elements, and contains a set of integrated applications and utilities. The most important thing is that the desktop environment provides a window manager. In the window system of graphical user interface, the window manager controls the behavior, position and appearance of windows. REIMEI1 uses PIXEL, the same desktop environment as Raspberry Pi operating system, which is a modified version of LXDE desktop environment.

X Server can be started in two ways, one is through the display manager, and the other is manually. In addition to starting X Server, the display manager also includes starting the corresponding Client program to form a complete desktop environment. The display manager is usually a graphical user interface, which displays at the end of the system desktop startup process to replace the default shell.

The X11 desktop environment of REIMEI1 uses lightdm as the display manager.

4.4.1 Start X11 Desktop

To start the X11 desktop, you need to execute the following command.

```
sudo systemctl start lightdm
```

4.4.2 Configure the System to Boot to X11 Desktop

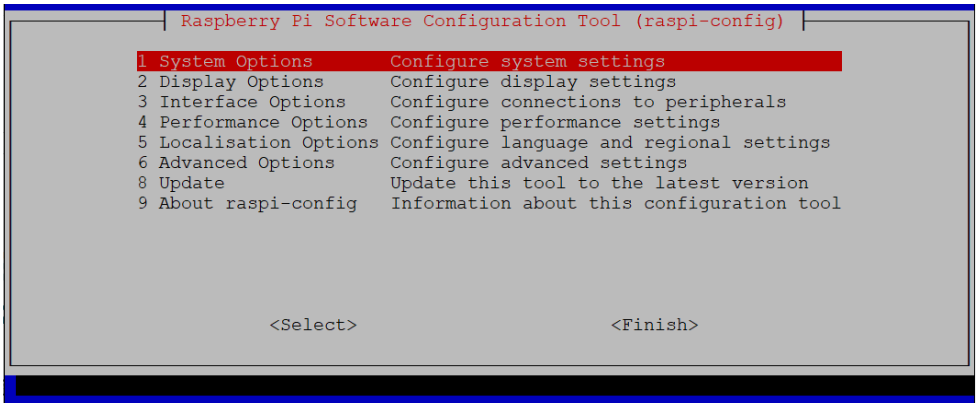
There are two ways to configure the system to automatically boot to X11 desktop: 1) configure it by executing raspi-config command on the command line, and 2) configure it by using the graphical interface Raspberry Pi Configuration tool.

NOTE: Only the sudo systemctl enable lightdm service can't make the system boot to X11 desktop automatically.

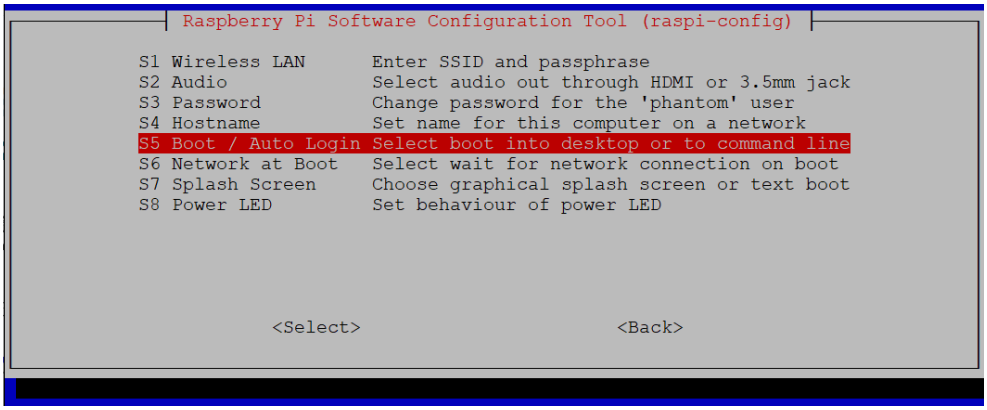
1.Execute the raspi-config command from the command line for configuration.

```
sudo raspi-config
```

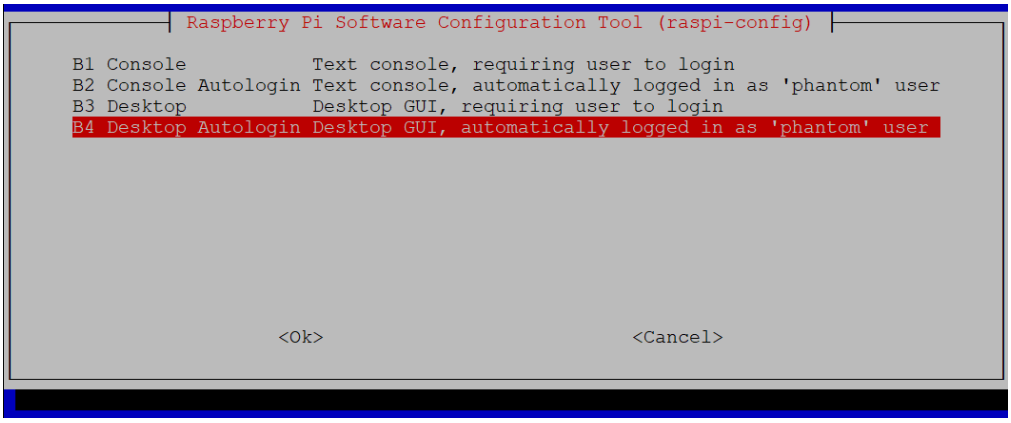
Choose System Options



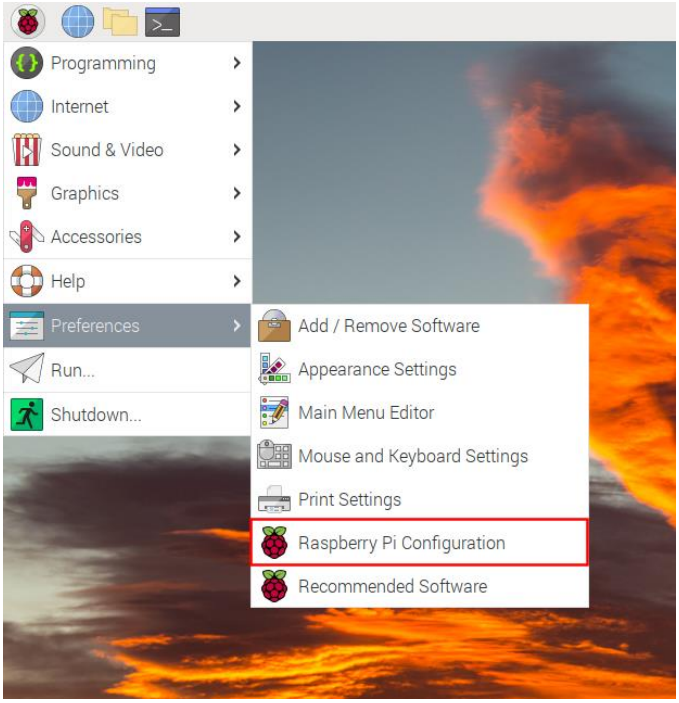
Choose Boot / Auto Login



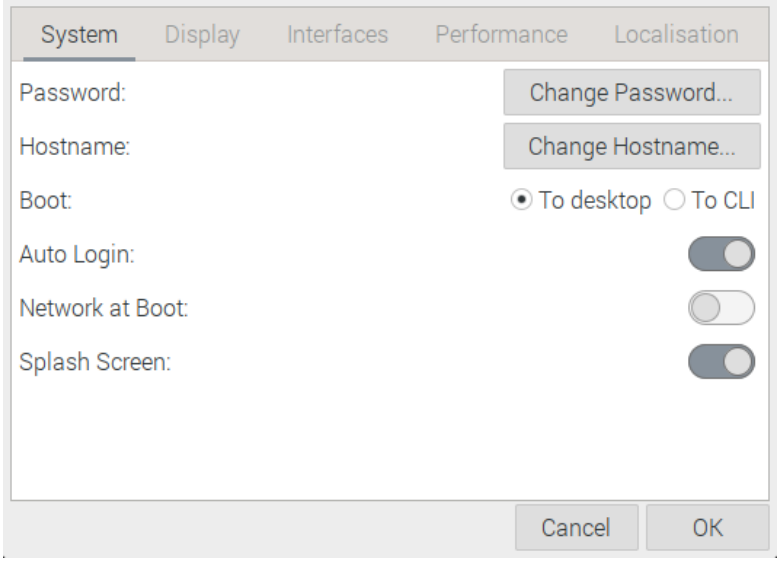
Choose Desktop Autologin，start the desktop and log in automatically.



2. Configure through the image interface Raspberry Pi Configuration tool.



Auto Login.On the System page, configure Boot To desktop and enable To desktop.

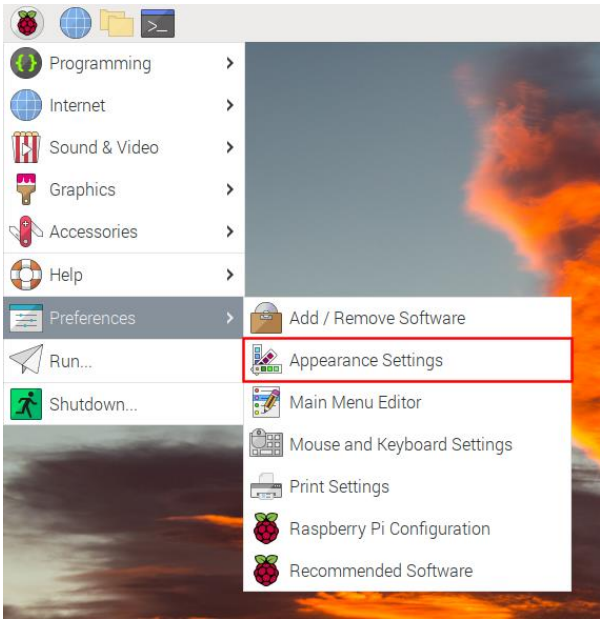


4.4.3 Desktop Taskbar

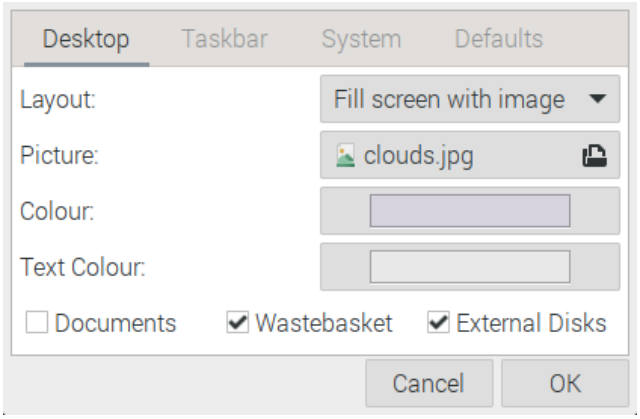
NO.	Function Description
1	Start menu
2	Web browser
3	Primary user folder
4	Terminal shortcut
5	Bluetooth connection icon
6	WiFi connection icon
7	HDMI sound output volume configuration
8	System time display

4.4.4 Desktop Personalization Settings

Choose Perferences->Appearance Settings.



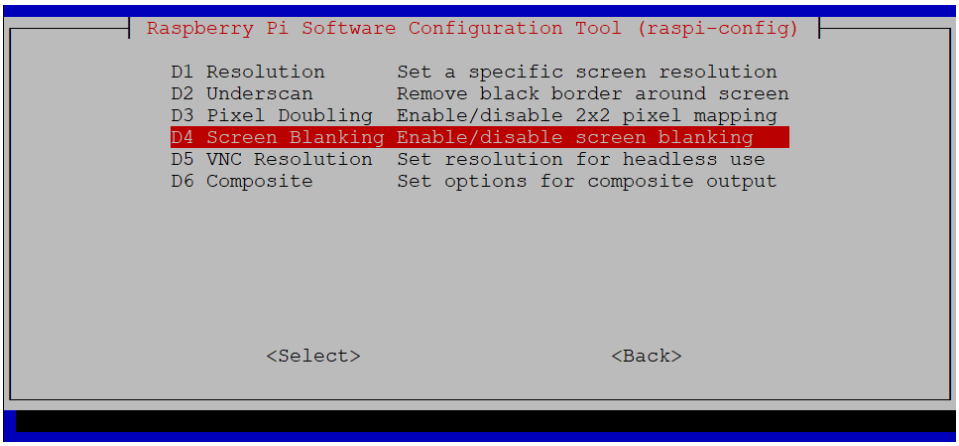
Information such as desktop picture display, taskbar and system font can be configured in Appearance Settings.



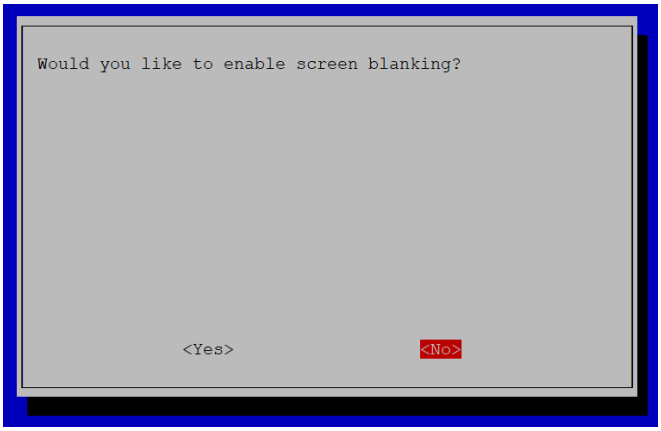
4.4.5 Turn off the automatic screen off function

1. Execute the raspi-config command from the command line for configuration.

Choose Display Options，then choose Screen Blanking，press enter confirm

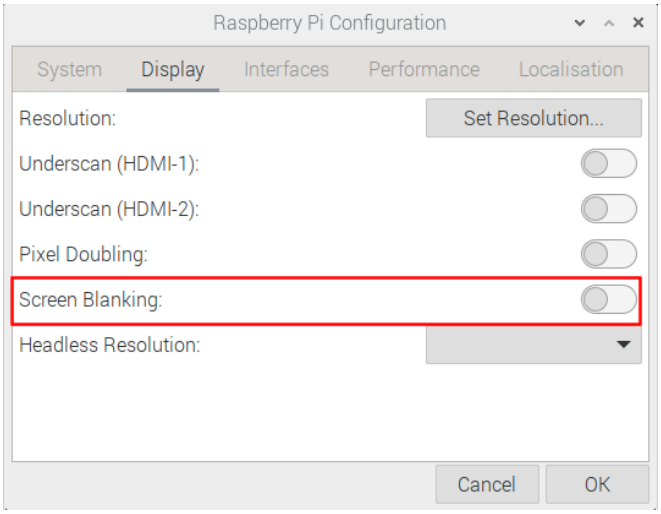


Select No to disable the automatic screen off function.



2. Configure through the image interface Raspberry Pi Configuration tool.

In Display page choose disable Screen Blanking



4.4.6 Check X11 Desktop System Log

You can help debug the problem by checking at the X11 system log.

Check lightdm log

```
# sudo is required to view the lightdm log
sudo cat /var/log/lightdm/lightdm.log
```

Check Xorg log

```
cat /var/log/Xorg.0.log
```

4.4.7 X11 Desktop Screenshot

Scrot tool is pre-installed by default. If it is not installed, please execute the following instructions:

```
sudo apt-get install scrot
```

Press the PrtScn screen print button on the keyboard to capture the screen, and the file is kept in the main user folder /home/phantom.

Screen capture of the current terminal window.

```
scrot -u
```

Use the mouse to select a window or area for screen capture.

```
scrot -s
```

Add a delayed screenshot

```
scrot -d x
```

Indicates a delay of x seconds for screen capture.

Perform a screenshot at the remote SSH command line.
First execute the following command

```
export DISPLAY=:0.0
```

Then execute the scrot command, and the screenshot will be saved in the directory where the current command line terminal is located.

4.4.8 Play Audio

Check the sound card

```
aplay -l
```

Configure sound card

```
alsamixer
```

Press F6 and select AML-AUGESOUND

Move the left and right arrow keys to select configuration options, and move up and down to switch configuration values. 00 indicates that the current volume is normal, while MM indicates that this channel

is mute. You can switch between mute and normal states by pressing the M key on the keyboard.

Configuration Item	Default Parameters
Audio HAL Format	PCM
Audio In Source	TDMIN_A
Audio Out Sink	TDMIN_A
Audio hdmi-out mute	off
Audio spdif format	stereo PCM
Audio spdif mute	off
Audio spdifin source	spdifin pad

NOTE: If there is no sound output from HDMI, please check if Audio hdmi-out mute is on and press the M key to switch to normal mode.

Users can use the following commands to play sounds:

```
aplay test.mp3
```

Of course, it also supports users to specify sound card devices for recording:

```
aplay -Dhw:<sound_card_id> test.mp3
```

4.5 Introduction to Weston Desktop Use

Wayland is a set of communication protocol between the display server (Wayland Composer) and the client, which aims at replacing the X graphics system on Linux. Applications can use this protocol to talk with the display server, so that they can get the user's input at the same time. The display server of Wayland is called a synthesizer, and the application is the client of Wayland. weston is a reference implementation of Wayland Composer, which provides a basic desktop application environment. The weston desktop of REIMEI1 has realized the hard decoding of graphics.

4.5.1 Weston Desktop Startup and Shutdown

Weston application is not enabled by default. You can start, close and configure the startup to automatically start the weston desktop as follows.

- **Start weston desktop**

```
sudo systemctl start weston
```

- **Stop weston desktop**

```
sudo systemctl stop weston
```

- **Restart weston desktop**

```
sudo systemctl restart weston
```

The default desktop display of Weston is shown in the following figure. You can change the desktop

background color, status bar color, desktop picture, add application startup shortcut and display status bar according to your needs. Please refer to Weston Advanced Configuration for details.

4.5.2 Configure System Startup To Weston Desktop.

By default, the system starts by command line. Refer to the following instructions to configure the system for desktop mode.

Modify the weston.service service file

```
sudo nano /etc/systemd/system/weston.service
```

Delete the # before WantedBy to make the WantedBy configuration take effect

```
[Unit]
Description=Weston Wayland Compositor
RequiresMountsFor=/run

[Service]
User=root
PAMName=login
EnvironmentFile=-/etc/default/weston
ExecStart=/usr/bin/weston-start -v -e -- $OPTARGS

[Install]
WantedBy=multi-user.target
```

Enable weston to start automatically

```
sudo systemctl enable weston
```

4.5.3 Weston Desktop System Log

You can help debug the problem by checking weston system logs.

```
cat /var/log/weston.log
```

4.5.4 Screenshot of Weston Desktop

Modify /etc/default/weston

```
sudo nano /etc/default/weston
```

Add optargs = "-debug" in the last line, and then restart weston

```
sudo systemctl restart weston
```

Execute the screenshot command

```
weston-screenshooter
```

The shortcut key of the screenshot command is Win+S. After the screenshot command is executed, the image file of wayland-screenshot-xxx-xxx.png will be generated in the corresponding directory, and the default saving directory is the system root directory/.

4.5.5 Screen Recording on Weston Desktop

The shortcut key Win+r performs start/stop screen recording and generates a file in. wcap format, which is a low-loss weston proprietary format and can be decoded by wcap tools:

```
wcap-decode --yuv4mpeg2 capture.wcap > capture.y4m
```

Y4m is an original format, which can be opened by vlc or encoded by ffmpeg:

```
ffmpeg -y -i capture.y4m -c:v libx264 -pix_fmt yuv420p capture.mp4
```

4.5.6 Play Video

REIMEI1 system supports video hard decoding, and the video hardware depends on Wayland. Please keep the weston desktop open. Please refer to Gstreamer for detailed operation of video playback.

4.6 System Configuration

4.6.1 Network Configuration

4.6.1.1 Scan Available WiFi Networks.

```
sudo iwlist wlan0 scan
```

4.6.1.2 Connected to WiFi.

Method 1：

```
sudo raspi-config
```

Select 1 System Options to find S1 Wireless LAN. For the first time, you need to select a country, and China is CN. Then you will be asked to enter the WiFi name, then enter the WiFi password, and then save and exit. If the country code is set, it needs to be restarted.

Method 2：

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Add the following to the file

```
country=CN
network={
      ssid="WiFi_SSID"
      psk="Password"
}
```

Ctrl+X exits and press return to save.

4.6.1.3 Set Static IP.

Configure static IP, set the static IP of eth0 network card to 192.168.168.108, set the default route to 192.168.168.1, and set DNS to 192.168.168.1(DNS can be omitted):

```
sudo nano /etc/dhcpcd.conf

interface eth0
static ip_address=192.168.168.108/24
static route=192.168.168.1
static domain_name_servers=192.168.168.1
```

4.6.1.4 Set Network Priority

When multiple networks are available at the same time, if you want to specify the network priority, you need to follow the following methods.

Set the network priority of interface wlan0 to 200. The smaller the value, the higher the priority:

```
sudo nano /etc/dhcpcd.conf

#Add the following to the file
interface wlan0
metric 200
```

Ctrl+X exits and press return to save.

NOTE:Please connect the 2.4GHz/5GHz WiFi/BT antenna, otherwise the WiFi connection may fail due to weak signal.

4.6.2 Bluetooth Configuration

REIMEI1 enables the Bluetooth function by default. If you need to set Bluetooth, you can use the bluetoothctl command to set Bluetooth.

4.6.2.1 Bluetoothctl USE

Scan

```
bluetoothctl scan on/off
```

Find device

```
bluetoothctl discoverable on/off
```

Trust device

```
bluetoothctl trust [MAC]
```

Pair

```
bluetoothctl pair [MAC]
```

Connect

```
bluetoothctl connect [MAC]
```

Disconnect

```
bluetoothctl disconnect [MAC]
```

More about Bluetooth command configuration

```
bluetoothctl
help
```

NOTE:Please connect the 2.4GHz/5GHz WiFi/BT antenna, otherwise Bluetooth may not be able to scan all surrounding devices due to weak signal.

4.6.3 Add External Storage

When mounting the SD card, it is necessary to pay attention that the image cannot be burned in SD, otherwise the system will boot from the SD card.

Mount SD card to /mnt directory

```
sudo mount /dev/mmcblk0 /mnt
```

Uninstall SD card

```
sudo umount /mnt
```

4.6.3.1 Set Up Automatic Mount.

1. Get the UUID of the storage device:

```
lsblk
```

2. Find the UUID corresponding to the device, assuming that the UUID of the device is 5C24-1453.
3. Open fstab file：

```
sudo nano fstab
```

4. Write UUID into fstab file:

```
UUID=5C24-1453 /mnt/mydisk fstype defaults,auto,users,rw,nofail 0 0
```

4.6.4 User Management

4.6.4.1 sudo Mechanism

Phantom is added to the sudoer user group by default, and is allowed to use root privilege. When executing a command, you can execute the command with root privilege by adding sudo before the command.

4.6.4.2 Switch to Root.

Switch to root

```
sudo -s
```

4.6.4.3 Create New User

```
sudo adduser <username>
```

After executing the command, you will be asked to enter the password and other information. After the creation, a new folder will be generated for the new user in the home directory.

4.6.4.4 Disable default user

```
sudo passwd -l phantom
```

4.6.4.5 Enable default user

```
sudo passwd -u phantom
```

4.7 X11 Desktop Advanced Configuration

4.7.1 Lightdm Config File

/usr/share/lightdm/lightdm.conf.d/01_debian.conf is the system configuration and cannot be edited by ordinary users.
/etc/lightdm/lightdm.conf Global common configuration modification file
/etc/lightdm/pi-greeter.conf greeter Configuration file
/etc/lightdm/lightdm.conf Configuration can override system configuration parameters.

lightdm.conf default config：

```
[Seat:*]
```

```
greeter-session=pi-greeter
greeter-hide-users=false

display-setup-script=/usr/share/dispsetup.sh

autologin-user=phantom
```

greeter-hide-users Indicates whether to hide the user list.
greeter-setup-script Specifies the command to run before the welcome interface starts.
session-setup-script Run before the user session starts. If it fails, the user session will not start.
session-cleanup-script Run after the welcome interface or user session exits.
display-setup-script Used to specify the command to be executed after X server is started.
display-stopped-script Used to specify the command to be executed after X server exits.

The greeter-session field in the lightdm.conf is configured as pi-greeter, which corresponds to the /etc/lightdm/pi-greeter.conf file.

Default configuration for pi-greeter.conf

```
[greeter]
default-user-image=/usr/share/raspberrypi-artwork/raspberry-pi-logo.png
desktop_bg=#d6d6d3d3dede
wallpaper=/usr/share/rpd-wallpaper/clouds.jpg
wallpaper_mode=crop
gtk-theme-name=PiXflat
gtk-icon-theme-name=PiXflat
gtk-font-name=PibotoLt 12
```

It mainly completes the configuration of default desktop, including desktop background picture, default font and theme.

4.7.2 Disable Screen Blanking

Edit /etc/lightdm/lightdm.conf config file

```
sudo nano /etc/lightdm/lightdm.conf
```

Modify xserver-command parameter value

```
xserver-command=X -s 0 -dpms
```

It takes effect after restarting the device.

4.8 Weston Advanced Configuration

According to the actual application scenario, you may need to adjust the configuration of the existing

weston desktop, such as changing the desktop background color, changing the desktop picture, removing the status bar, adding desktop shortcuts and so on.

4.8.1 Weston Configuration File Introduction

Weston obtains the configuration information from its startup command line parameters and configuration files. The configuration file of the desktop of the REIMEI1 development board Weston is/etc /etc/aml-weston.ini

The aml-weston.ini file consists of multiple section, which can appear in any order or be omitted to use the default configuration values. The format of each section is as follows:

```
[SectionHeader]
Key1=Value1
Key2=Value2
```

Comment out a line with #, and it will be ignored.

```
#Key2=Value2
```

The SectionHeader section can be the following fields:

core	The core modules and options
libinput	Input device configuration
shell	Desktop customization
launcher	Add launcher to the panel
output	Output configuration
input-method	Onscreen keyboard input
keyboard	Keyboard layouts
terminal	Terminal application options
xwayland	XWayland options
screen-share	Screen sharing options
autolaunch	Autolaunch options

Possible Value types of value include strings, signed and unsigned 32-bit integers, and Boolean values. The string cannot be referenced, does not support any escape sequence, and runs until the end of the line. Integers can be given in decimal (e.g. 123), octal (e.g. 0173) and hexadecimal (e.g. 0x7b) forms. Boolean values can only be true or false.

Please refer to the Official document of configuration file for a detailed description of Key and Value in each section.

4.8.2 Customization of Weston Desktop

- **Change the desktop background color**

Set the desktop background color to blue, and modify the value of the background-color field in the [shell] block.

```
[shell]
background-color=0xff0000ff
```

32bits Hexadecimal numbers represent transparent, red, green and blue in sequence in groups of 8bits:

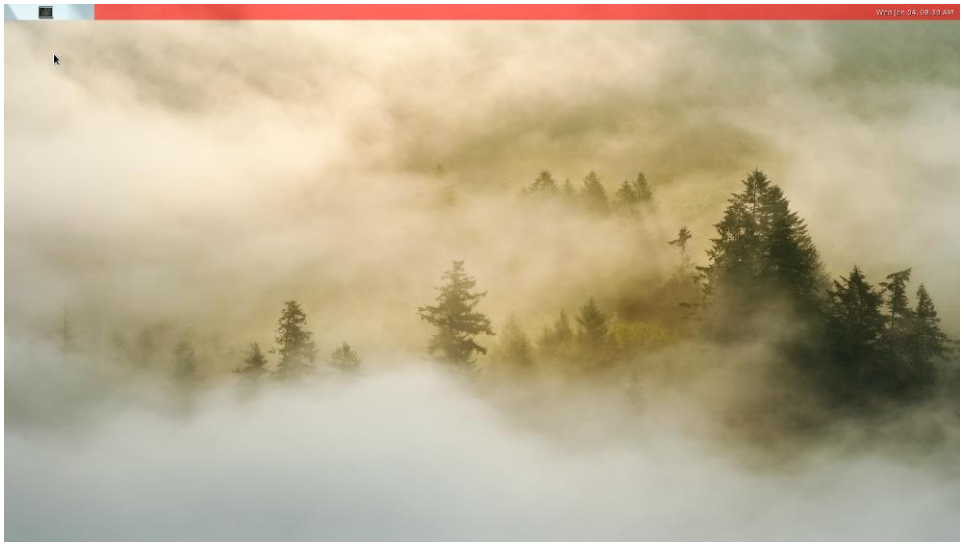
0xffff0000	red
0xff00ff00	green
0xff0000ff	blue
0x00ffffff	Fully transparent

- **Add a desktop background picture**

The location of the desktop picture is/home/phantom/pictures/desktop.jpg. Add the background-image field in the [shell] block and fill in the correct desktop picture path.

```
[shell]
background-image=/home/phantom/Pictures/desktop.jpg
```

After restarting the device, a new desktop picture can be displayed. Examples are as follows:



- **Configure status bar display**

Set the status bar to not show.

Add a panel-position field in the [shell] block with a value of none, and specify the background color as fully transparent.

```
[shell]
panel-position=none
background-color=0x00FFFFFF
```

In addition, the values of the panel-position field can be configured as top, bottom, left and right, which means that the status bar is displayed at the top, bottom, left and right in turn.

4.8.3 Add Desktop Shortcut

Add an icon field and a path field to the [launcher] block. The icon field is used to specify the display picture of the shortcut of the executable program, and the path field specifies the path of the executable

file.

At present, the system has defined three shortcuts in the status bar, which can be modified or added according to your needs:

```
[launcher]
icon=/usr/share/weston/icon_flower.png
path=/usr/bin/weston-flower

[launcher]
icon=/usr/share/weston/icon_ivi_smoke.png
path=/usr/bin/weston-smoke

[launcher]
icon=/usr/share/icons/gnome/32x32/apps/utilities-terminal.png
path=/usr/bin/weston-terminal --shell=/usr/bin/bash
```

4.9 Compilation Tool Chain

By default, the system has installed the gcc compiler with version 10.2.1.

```
phantom@phantom:/ $ gcc --version
gcc (Debian 10.2.1-6) 10.2.1 20210110
Copyright (C) 2020 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

If you need to use cross-compilation, you can install linaro cross-compilation tool, which is a provider of free licensed version of ARM cross-compilation tool. You can get the officially compiled cross tool chain on theArm GNU Downloads PAGE.
Select gcc-arm-10.2-2020.11-x86_64-aarch64-none-linux-gnu.tar.xz to download, decompress the compressed package and add the tool chain directory to the user profile.
Take ubuntu20 system as an example, assuming that gcc-arm-10.2-2020.11-x86_64-aarch64-none-linux-gnu.tar.xz has been placed in the ~/tools directory.

```
cd ~/tools
xz -d gcc-arm-10.2-2020.11-x86_64-aarch64-none-linux-gnu.tar.xz
tar xvf gcc-arm-10.2-2020.11-x86_64-aarch64-none-linux-gnu.tar

echo PATH=$PATH:~/tools/gcc-arm-10.2-2020.11-x86_64-aarch64-none-linux-gnu/bin >> ~/.bashrc
```

4.10 Device Files Interface

Function	Peripheral device	Linux device file
eMMC	MMC1	/dev/mmcblk1
Micro SD	MMC0	/dev/mmcblk0

Uart	UART0	/dev/ttyS0
i2c0	I2C	/dev/i2c-0
i2c1	I2C	/dev/i2c-1
spi dev0	SPI dev	/dev/spidev0.0
spi dev1	SPI dev	/dev/spidev0.1

4.11 40-Pin GPIO Programming Guide

REIMEI1 has a pin arrangement with a spacing of 2X20P of 2.54mm, which leads out 28 GPIO of the main control chip. Users can control these GPIO through software. At present, it has supported 2 I2C channels, 1 UART channel, 1 SPI channel and multi-channel input/output configurable GPIO.

TIP: REIMEI1 40-Pin pin is compatible with Raspberry Pie 40-Pin pin (I2C, UART, SPI).

wiringPi	GPIO	Function	Physical Pins		Function	GPIO	wiringPi
		3V3	1	2	5V		
8	GPIO70	PIN3	3	4	5V		
9	GPIO71	PIN5	5	6	GND		
7	GPIO83	PIN7	7	8	PIN8	GPIO37	15
		GND	9	10	PIN10	GPIO38	16
0	GPIO79	PIN11	11	12	PIN12	GPIO33	1
2	GPIO75	PIN13	13	14	GND		
3	GPIO76	PIN15	15	16	PIN16	GPIO77	4
		3V3	17	18	PIN18	GPIO72	5
12	GPIO45	PIN19	19	20	GND		
13	GPIO46	PIN21	21	22	PIN22	GPIO73	6
14	GPIO48	PIN23	23	24	PIN24	GPIO47	10
		GND	25	26	PIN26	GPIO78	11
30	GPIO0	PIN27	27	28	PIN28	GPIO1	31
21	GPIO36	PIN29	29	30	GND		
22	GPIO32	PIN31	31	32	PIN32	GPIO35	26
23	GPIO82	PIN33	33	34	GND		
24	GPIO40	PIN35	35	36	PIN36	GPIO80	27
25	GPIO74	PIN37	37	38	PIN38	GPIO81	28
		GND	39	40	PIN40	GPIO31	29

4.11.1 Using libgpiod to Control GPIO

Install ligpiod

```
sudo apt-get update
#Install the static library and header file of libgpiod.
sudo apt-get install libgpiod-dev
```

```
#Install command-line tools based on libgpiod
sudo apt-get install gpiod
```

Libgpiod supports six command-line test commands, namely:

- gpiodetect- List all gpiochips existing on the system, their names, tags and number of gpio lines.
- gpioinfo- List all lines of the specified gpiochips, their names, consumers, directions, activity status and additional flags.
- GPIOget- Read the value of the specified gpio line.
- GPIOset- Set the value of the specified gpio lines, which may be kept for export and waiting for timeout, user input or signal.
- gpiofind- Find the row offset of the gpiochip name and the given row name.
- gpiomon- Wait for the event on the GPIO line, specify the event to watch, how many events to deal with before exiting or how many events should be reported to the console.

Check gpiochip information

```
phantom@phantom:~ $ gpioinfo
gpiochip0 - 87 lines:
        line   0:      "PIN27"         kernel   input  active-high [used]
        line   1:      "PIN28"         kernel   input  active-high [used]
        line   2:  "EMMC_DAT0"         kernel   input  active-high [used]
        line   3:  "EMMC_DAT1"         kernel   input  active-high [used]
        line   4:  "EMMC_DAT2"         kernel   input  active-high [used]
        line   5:  "EMMC_DAT3"         kernel   input  active-high [used]
        line   6:  "EMMC_DAT4"         kernel   input  active-high [used]
        line   7:  "EMMC_DAT5"         kernel   input  active-high [used]
        line   8:  "EMMC_DAT6"         kernel   input  active-high [used]
        line   9:  "EMMC_DAT7"         kernel   input  active-high [used]
        line  10:   "EMMC_CLK"         kernel   input  active-high [used]
        line  11:   "NAND_ALE"         unused   input  active-high
        line  12:   "EMMC_CMD"         kernel   input  active-high [used]
        line  13:          "-"         unused   input  active-high
        line  14:   "EMMC_RST"         unused   input  active-high
        line  15: "EMMC_NAND_DQS" kernel input active-high [used]
        line  16:          "-"         unused   input  active-high
        line  17:          "-"         unused   input  active-high
        line  18:    "SD_DAT0"         kernel   input  active-high [used]
        line  19:    "SD_DAT1"         kernel   input  active-high [used]
        line  20:    "SD_DAT2"         kernel   input  active-high [used]
        line  21:    "SD_DAT3"         kernel   input  active-high [used]
        line  22:     "SD_CLK"         kernel   input  active-high [used]
        line  23:     "SD_CMD"         kernel   input  active-high [used]
        line  24:      "SD_CD"           "cd"   input  active-high [used]
        line  25:    "USB_PSU" "fe03a080.usb3phy" output active-low [used]
        line  26:  "VDDEE_PWM"         unused   input  active-high
```

```
line  27: "VDDCPU_PWM"          kernel    input   active-high [used]
line  28:          "LED"          "act"   output   active-high [used]
line  29:    "DEBUG_TX"          kernel    input   active-high [used]
line  30:    "DEBUG_RX"          kernel    input   active-high [used]
line  31:       "PIN40"          unused    input   active-high
line  32:       "PIN31"          unused    input   active-high
line  33:       "PIN12"          unused    input   active-high
line  34:            "-"         unused    input   active-high
line  35:       "PIN32"          unused    input   active-high
line  36:       "PIN29"          unused    input   active-high
line  37:        "PIN8"          kernel    input   active-high [used]
line  38:       "PIN10"          kernel    input   active-high [used]
line  39:            "-"         unused    input   active-high
line  40:       "PIN35"          unused    input   active-high
line  41:    "HDMI_SDA"          kernel    input   active-high [used]
line  42:    "HDMI_SCL"          kernel    input   active-high [used]
line  43:    "HDMI_HPD"          kernel    input   active-high [used]
line  44:    "HDMI_CEC"          kernel    input   active-high [used]
line  45:       "PIN19"          kernel    input   active-high [used]
line  46:       "PIN21"          kernel    input   active-high [used]
line  47:       "PIN24"       "spi0.0"   output   active-high [used]
line  48:       "PIN23"          kernel    input   active-high [used]
line  49: "PCIE_RESET"           unused    input   active-high
line  50: "WIFI_SD_D0"           kernel    input   active-high [used]
line  51: "WIFI_SD_D1"           kernel    input   active-high [used]
line  52: "WIFI_SD_D2"           kernel    input   active-high [used]
line  53: "WIFI_SD_D3"           kernel    input   active-high [used]
line  54: "WIFI_SD_CLK" kernel input active-high [used]
line  55: "WIFI_SD_CMD" kernel input active-high [used]
line  56:            "-"         unused    input   active-high
line  57:            "-"         unused    input   active-high
line  58:            "-"         unused    input   active-high
line  59:            "-"         unused    input   active-high
line  60:       "BT_ON"     "shutdown"   output   active-high [used]
line  61:       "WL_ON"          unused    input   active-high
line  62: "BTUART_A_TX" kernel input active-high [used]
line  63: "BTUART_A_RX" kernel input active-high [used]
line  64: "BTUART_A_CTS_N" kernel input active-high [used]
line  65: "BTUART_A_RTS_N" kernel input active-high [used]
line  66:            "-"         unused    input   active-high
line  67:            "-"         unused    input   active-high
line  68:            "-"         unused    input   active-high
line  69:            "-"         unused    input   active-high
line  70:         "PIN3"          kernel    input   active-high [used]
```

```
        line  71:          "PIN5"          kernel     input   active-high [used]
        line  72:         "PIN18"          unused     input   active-high
        line  73:         "PIN22"          unused     input   active-high
        line  74:         "PIN37"          unused     input   active-high
        line  75:         "PIN13"          unused     input   active-high
        line  76:         "PIN15"          unused     input   active-high
        line  77:         "PIN16"          unused     input   active-high
        line  78:         "PIN26"       "spi0.1"    output   active-high [used]
        line  79:         "PIN11"          unused     input   active-high
        line  80:         "PIN36"          unused     input   active-high
        line  81:         "PIN38"          unused     input   active-high
        line  82:         "PIN33"          unused     input   active-high
        line  83:          "PIN7"          unused     input   active-high
        line  84:     "LAN_LEDG"            kernel     input   active-high [used]
        line  85:     "LAN_LEDY"            kernel     input   active-high [used]
        line  86:            "-"          unused     input   active-high
```

It can be seen that the system has only one gpiochip0 with 87 GPIO pins, and the GPIO that has been driven or occupied by the system will be displayed as [used] in the last column.

According to the name of GPIO pin in the datasheet **ED-REIMEI1_Datasheet_CN.pdf** and the return result of gpioinfo, the corresponding line number is found. Taking PIN7 as an example, the pin name of PIN7 in the data book is PIN 7, and the line number corresponding to PIN 7 is 83.

```
phantom@phantom:~ $ gpioinfo | grep PIN7
        line  83:          "PIN7"          unused     input   active-high
```

Config GPIO output

```
#Set the line83 pin of chip0 to low level.
gpioset 0 83=0

#Set the line83 pin of chip0 to high level.
gpioset 0 83=1
```

Config GPIO input

```
#Read the state of line83 pin of chip0.
gpioget 0 83
```

When the return value is 1, it means that the pin of line83 is high, and when the return value is 0, it means that the pin of line83 is low.

4.11.2 Using the wiringPi Library to Control GPIO

We have made modifications to the wiringPi library based on Raspberry Pi. You can use our modified

wiringPi library to control the GPIO pins of REIMEI1.

4.11.2.1 Installing wiringPi

```
git clone https://github.com/edatec/phantom-wiringPi.git
cd phantom-wiringPi
./build clean
./build
```

If you wish to uninstall the wiringPi library, please execute

```
./build uninstall
```

4.11.2.2 read gpio

```
gpio readall
```

Correspondence between GPIO and wiringPi encoding

```
+------+-----+---------+------+---+--Phantom-+---+------+---------+-----+-----+
| GPIO | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | GPIO |
+------+-----+---------+------+---+----++----+---+------+---------+-----+------+
|      |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |      |
|   70 |   8 |   SDA.1 |   IN | 0 |  3 || 4  |   |      | 5v      |     |      |
|   71 |   9 |   SCL.1 |   IN | 0 |  5 || 6  |   |      | 0v      |     |      |
|   83 |   7 |    PIN7 |   IN | 0 |  7 || 8  | 0 |  IN  | TxD     | 15  | 37   |
|      |     |      0v |      |   |  9 || 10 | 0 |  IN  | RxD     | 16  | 38   |
|   79 |   0 |   PIN11 |   IN | 0 | 11 || 12 | 0 |  IN  | PIN12   | 1   | 33   |
|   75 |   2 |   PIN13 |   IN | 0 | 13 || 14 |   |      | 0v      |     |      |
|   76 |   3 |   PIN15 |   IN | 0 | 15 || 16 | 0 | IN   | PIN16   | 4   | 77   |
|      |     |    3.3v |      |   | 17 || 18 | 0 | IN   | PIN18   | 5   | 72   |
|   45 |  12 |    MOSI |   IN | 0 | 19 || 20 |   |      | 0v      |     |      |
|   46 |  13 |    MISO |   IN | 0 | 21 || 22 | 0 | IN   | PIN22   | 6   | 73   |
|   48 |  14 |    SCLK |   IN | 0 | 23 || 24 | 0 | IN   | CE0     | 10  | 47   |
|      |     |      0v |      |   | 25 || 26 | 0 | IN   | CE1     | 11  | 78   |
|    0 |  30 |   SDA.0 |   IN | 0 | 27 || 28 | 0 | IN   | SCL.0   | 31  | 1    |
|   36 |  21 |   PIN29 |   IN | 0 | 29 || 30 |   |      | 0v      |     |      |
|   32 |  22 |   PIN31 |   IN | 0 | 31 || 32 | 0 | IN   | PIN32   | 26  | 35   |
|   82 |  23 |   PIN33 |   IN | 0 | 33 || 34 |   |      | 0v      |     |      |
|   40 |  24 |   PIN35 |   IN | 0 | 35 || 36 | 0 | IN   | PIN36   | 27  | 80   |
|   74 |  25 |   PIN37 |   IN | 0 | 37 || 38 | 0 | IN   | PIN38   | 28  | 81   |
|      |     |      0v |      |   | 39 || 40 | 0 | IN   | PIN40   | 29  | 31   |
+------+-----+---------+------+---+----++----+---+------+---------+-----+------+
| GPIO | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | GPIO |
+------+-----+---------+------+---+--Phantom-+---+------+---------+-----+------+
```

4.11.2.3 Configure Pins

Taking PIN15 as an example, PIN15 corresponds to wiringPi encoding pin 3.

Configure PIN15 as output:

```
gpio export 3 out

#output high level
gpio write 3 1

#output low level
gpio write 3 0
```

Configure PIN15 as input:

```
gpio export 3 in

#read PIN15 level
gpio read 3
```

4.11.3 i2c_dev

The device has supported two i2c buses.
/dev/i2c-0
/dev/i2c-1

Device	PIN NO.	PIN Name	Function
i2c-0	27	PIN27	SDA
	28	PIN28	SCL
i2c-1	3	PIN3	SDA
	5	PIN5	SCL

I2c-tools is installed in the system.
You can use i2cdetect to view devices on the i2c bus.

```
#View devices on i2c-0 bus
i2cdetect -y 0

#View devices on i2c-1 bus
i2cdetect -y 1
```

Development of i2c-dev Device Application Program Example.

4.11.4 spi_dev

The device already supports two spidev.
/dev/spidev0.0
/dev/spidev0.1

Device	PIN NO.	PIN Name	Function
spidev	24	PIN24	SPI_CE0
	26	PIN26	SPI_CE1
	19	PIN19	SPI_MOSI
	21	PIN21	SPI_MISO
	23	PIN23	SPI_CLK

Reference examples of spidev device application development.Example

4.12 QT Programming Guide

Qt, version 5.15.2, OpenGL ES library, version 3.2 have been installed in the system by default. The graphical display backend of this Qt development environment is based on Wayland, which supports hardware decoding. As a synthesizer of Wayland, weston needs to be kept on, and the weston desktop needs to be turned on and enabled.Please refer to Weston desktop start and shoutdown.

4.12.1 Quick Application of Qt Environment

The Qt environment of this system has provided a large number of test sample programs, which have been compiled for direct use. The directory of test samples is/usr/lib/aarq64-Linux-GNU/qt5/examples/.

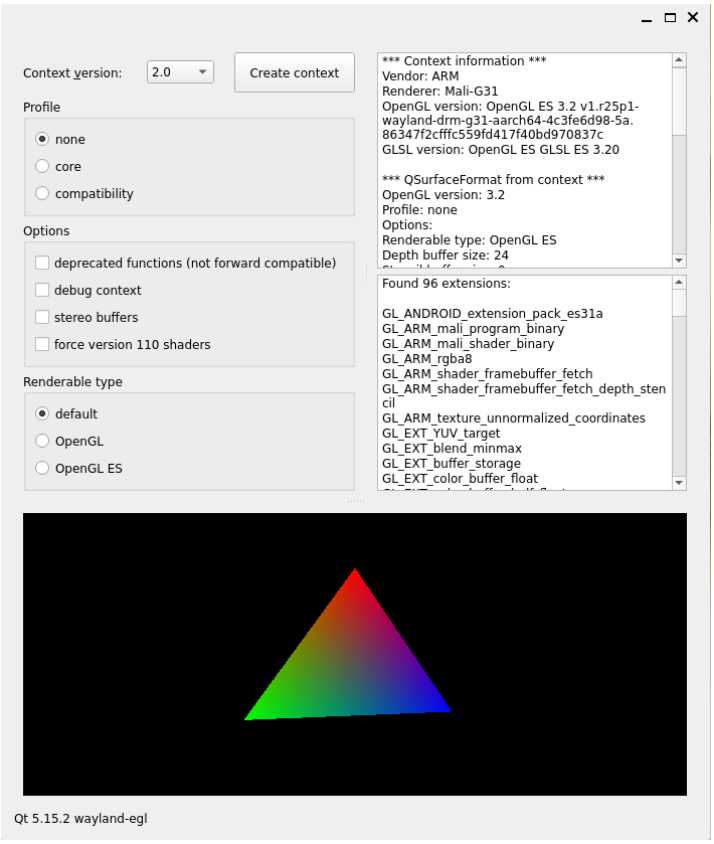
If you want to recompile, please refer to Compile Qt Widgets Application In Command Line and Compile Qt Quick(QML) Application in Command Line.

Execute Qt openGL ES example

```
/usr/lib/aarch64-linux-gnu/qt5/examples/opengl/contextinfo/contextinfo
```

TIP：If you want to launch Qt graphics program within weston from SSH, you need to prepend with sudo WAYLAND_DISPLAY=wayland-0 XDG_RUNTIME_DIR=/run/user/0 QT_QPA_PLATFORM=wayland-egl

Renderable type choose default or OpenGL ES，click Create context，result as follows:



4.12.2 Install Other Dependency Libraries

Depending on your application software needs, you may need to install some specific libraries:

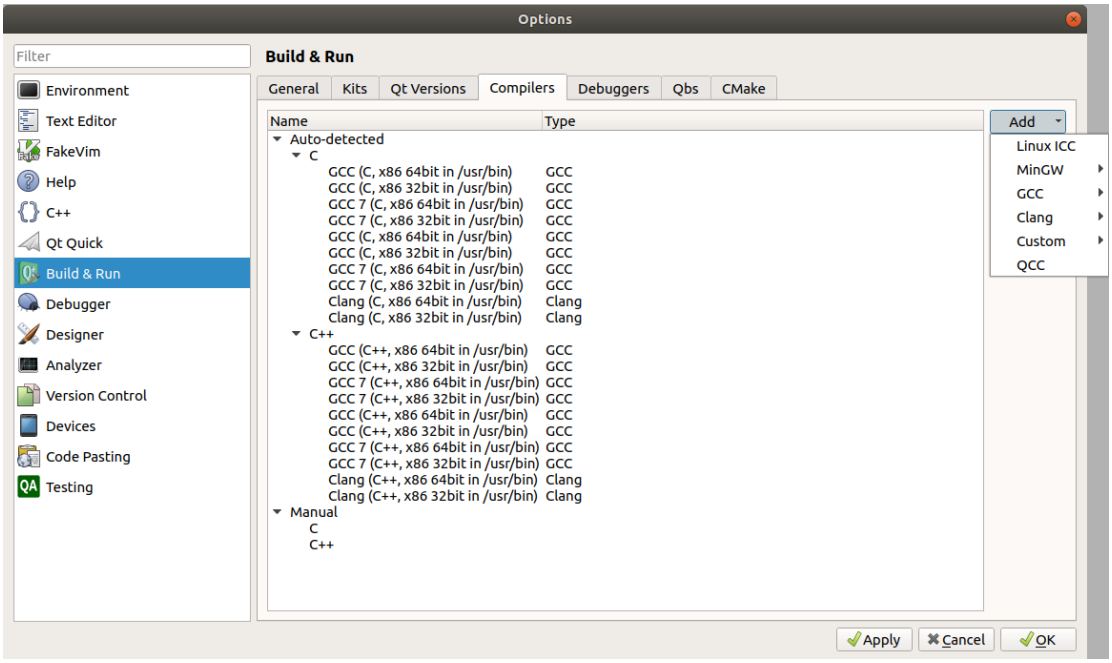
```
#Install qml develop environment
sudo apt-get install qtdeclarative5-dev

#Install QtMultimedia
sudo apt-get install qtmultimedia5-dev

#Install Qtserialport
sudo apt-get install libqt5serialport5-dev

#Install opengl develop environment
sudo apt-get install libgles2-mesa-dev

#Install QtMySQL
sudo apt-get install libqt5sql5-mysql
```

TIP：If the error of a library is missing when running the program, it can be replaced by the corresponding library name in the above way for installation.

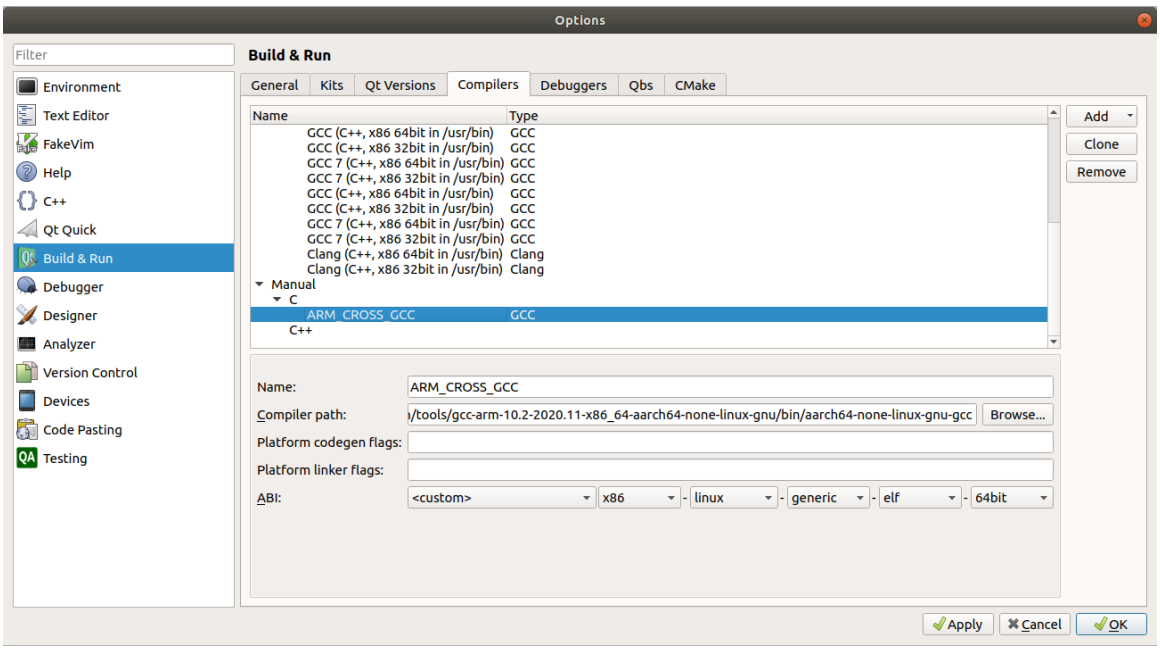
4.12.3 Configure Qt Creator Cross Compilation Environment.

At present, the system does not support the installation of Qt Creator on the device side through apt install. You can complete the development of application software interface by installing Qt Creator on the Host PC, and then copy the whole project to the REIMEI1 development board, and complete the overall compilation by command line. Except for opengl-based applications, other simple applications can also be compiled on the Host PC by cross-compilation, and then copy the executable files to the REIMEI1 development board to run.

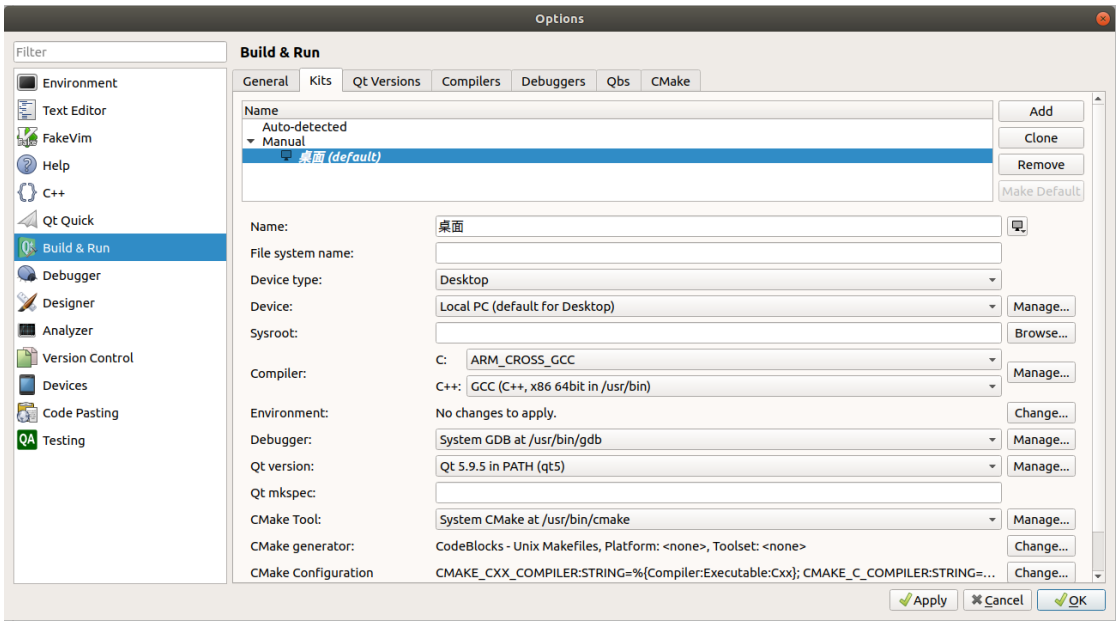
Open Qt Creator to enter the integrated development environment, select the Options option under the menu bar Tools, and open the Build & Run menu on the left. Select C under Manual, and then click Add to select GCC.



In the configuration box that appears below, enter a name that is easy to distinguish, then configure the path of the tool chain, select Browse... below, and select the location of aarch64-none-linux-gnu-gcc executable file. Then click OK to complete the configuration.



Finally, modify the building Kits (Kits), select the desktop (default), take the C compiler as an example, select the name of the cross-compiler tool just added in the drop-down list, and complete the addition of the C++ cross-compiler tool in a similar way.



4.12.4 Compile Qt Widgets Application In Command Line

Takes hellogles3 as an example

```
cd /usr/lib/aarch64-linux-gnu/qt5/examples/opengl/hellogles3
sudo qmake hellogles3.pro
sudo make
```

Execute ./hellogles3 it can run。

4.12.5 Compile Qt Quick(QML) Application in Command Line

- Install Qt Creator on the HOST PC side.
- For a new project, Projects selects Application(Qt Quick).
- Copy the generated project as a whole to the development board.
- Install qml application dependency library qtdeclarative5-dev on the device side.

he generated main.qml is the qml source file, and xxxx.pro is the project file.

```
#Install qml dependency library
sudo apt-get install qtdeclarative5-dev

#Generate makefile of the project, with xxxx as the corresponding project name.
qmake xxxx.pro
make
```

4.13 Gstreamer

GStreamer is a very powerful and universal framework for developing streaming media applications. Applications can connect all the steps of multimedia processing in series through Pipeline to achieve the expected results. Each step is realized by means of Element based on the GoObject object system and plugins, which facilitates the expansion of various functions.

gst-launch-1.0 is used to create and execute a pipline, so gst-launch is usually used to verify related functions before writing corresponding applications.

Install gstreamer tool

```
sudo apt-get install gstreamer1.0-tools
```

4.13.1 H264 Mkv Format Video Decoding

Test video download link.

If you want to execute gst-launch-1.0 through SSH remote terminal, you need to add sudo wayland _ display = wayland-0xdg _ runtime _ dir =/run/user/0qt _ qpa _ platform = wayland-egl before the command. If it is executed directly through the terminal window on weston desktop, you don't need to add this pre-instruction.

```
sudo WAYLAND_DISPLAY=wayland-0 XDG_RUNTIME_DIR=/run/user/0
QT_QPA_PLATFORM=wayland-egl gst-launch-1.0 -vvv filesrc
location=/home/phantom/Videos/jellyfish-5-mbps-hd-h264.mkv ! matroskademux ! h264parse !
v4l2h264dec ! video/x-raw,format=NV12 ! waylandsink
```

4.13.2 Mp4 Format Decoding

At present, the system gst-launch-1.0 does not support MP4 decoding, and it needs to be transcoded to H264 format through ffmpeg

```
ffmpeg -i input.mp4 -c:v copy -bsf h264_mp4toannexb output.h264
```

And then decoded by gst-launch-1.0

```
gst-launch-1.0 -vvv filesrc location=output.h264 ! h264parse ! v4l2h264dec ! video/x-raw,format=NV12 ! waylandsink
```

4.14 Docker

REIMEI1 supports Docker service.

4.14.1 Installation of Docker

1. Update the apt package index and install the package to allow apt to use the repository through HTTPS.

```
sudo apt-get update
sudo apt-get install ca-certificates curl gnupg lsb-release
```

2. Add Docker's official GPG key.

```
sudo mkdir -m 0755 -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

3.Setting repository

```
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/debian $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

4. Update apt package index again

```
sudo apt-get update
```

5. Install Docker Engine, containerd and Docker Compose

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

4.14.2 Uninstall of Docker

```
sudo apt-get remove docker docker-engine docker.io containerd runc
```

For more details, please refer to the official documents Install Docker Engine on Debian。

4.14.3 Check Docker

Check Docker version

```
docker --version
```

Check Docker system information

```
sudo docker info
```

```
Client:
 Context:    default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.10.2
    Path:     /usr/libexec/docker/cli-plugins/docker-buildx
  compose: Docker Compose (Docker Inc.)
    Version:  v2.16.0
    Path:     /usr/libexec/docker/cli-plugins/docker-compose
```

```
Server:
 Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
 Images: 0
 Server Version: 23.0.1
 Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Using metacopy: false
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
 Cgroup Driver: systemd
 Cgroup Version: 2
 Plugins:
  Volume: local
  Network: bridge host ipvlan macvlan null overlay
  Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
 Swarm: inactive
 Runtimes: io.containerd.runc.v2 runc
 Default Runtime: runc
 Init Binary: docker-init
 containerd version: 2456e983eb9e37e47538f59ea18f2043c9a73640
 runc version: v1.1.4-0-g5fd4c4d
 init version: de40ad0
 Security Options:
  apparmor
  seccomp
   Profile: builtin
  cgroupns
 Kernel Version: 5.4.180-phantom
 Operating System: Debian GNU/Linux 11 (bullseye)
 OSType: linux
 Architecture: aarch64
 CPUs: 4
 Total Memory: 1.937GiB
 Name: phantom
 ID: cfebd162-a5bc-49cd-a20c-63f114b333ec
 Docker Root Dir: /var/lib/docker
 Debug Mode: false
 Registry: https://index.docker.io/v1/
 Experimental: false
 Insecure Registries:
  127.0.0.0/8
 Live Restore Enabled: false
```

Docker installation is successful only when both Client and Server are running normally. Check that Docker service is running.

Check if the docker service is running.

```
sudo systemctl status docker
```

```
phantom@phantom:~ $ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2023-03-09 11:47:34 GMT; 13h ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 717 (dockerd)
      Tasks: 10
     Memory: 85.2M
        CPU: 1.477s
     CGroup: /system.slice/docker.service
             └─717 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
```

If not, please execute.

```
sudo systemctl start docker
```

4.14.4 Use Docker

Pull image

```
sudo pull <image_name>
```

List all local images

```
sudo docker image ls
```

Run container

```
sudo run <image_name>
```

Enter the container and use bash as the shell.

```
sudo docker exec -it <container_id> /bin/bash
```

Create a container by mirroring and enter the container

```
sudo docker run -it <image_name> /bin/bash
```

It is equivalent to executing the following command:

```
sudo docker run <image_name>
sudo docker exec -it <container_id> /bin/bash
```

Background running container

```
sudo docker run -d -it <image_name> /bin/bash
```

Specify a new name for the container.

```
sudo docker run -it --name <container_id> <image_name> /bin/bash
```

Specify host and container port mappings

```
sudo docker run -it --name <container_id> -p [host port]:[container port] <image_name> /bin/bash
```

Check all containers

```
sudo docker ps
```

Stop the running container and delete the container.

```
sudo docker stop <container_id> && docker rm <container_id>
```

Delete image

```
sudo docker image rm <image_name>
```

4.15 Bootloader Configuration Guide

The system uses u-boot as the BootLoader. In the boot stage, the boot configuration parameters are obtained by reading the boot.conf file under the boot partition, and the loading path of the system kernel, boot parameters and device tree files can be specified.

4.15.1 Specify the Configuration File Path

There is a boot.conf file in the root directory of the boot partition. load_prefix, dtb and bootargs respectively specify the loading path, device tree file and startup parameter file of the system configuration.

/boot/boot.conf

```
# boot.conf - boot configuration file for phantom/pm3

# Set to 1 to print the proprerties to the uart.
config_print=0

# Search for Image, dtb.img and bootargs.txt under this sub-directory
load_prefix=linux/

[board_type=1]
dtb=phantom.dtb

[board_type=1 boot_mode=0]
bootargs=bootargs-phantom-sd.txt

[board_type=1 boot_mode=1]
bootargs=bootargs-phantom-emmc.txt
```

Load_prefix specifies the search path for the file that started the configuration.
Dtb specifies the name of the device tree file.
Bootargs specifies the name of the startup parameter configuration file.
Boot_mode specifies the loading order of startup parameter configuration file loading.
A config_print of 1 will output the current startup configuration to the debugging serial port.

4.15.2 Modify bootargs Parameters

Bootargs-phantom-emmc.txt and bootargs-phantom-sd.txt correspond to the startup parameters of emmc and sd card respectively.

```
phantom@phantom:/ $ cat /boot/linux/bootargs-phantom-emmc.txt
earlycon=aml_uart,0xfe078000,921600 console=ttyS0,921600 console=tty1 loglevel=8 jtag=apao
root=/dev/mmcblk1p2 rootfstype=ext4 rootwait
```

You can modify the baud rate of debugging serial port.

```
earlycon=aml_uart,0xfe078000,115200 console=ttyS0,115200 console=tty1 loglevel=8 jtag=apao
root=/dev/mmcblk1p2 rootfstype=ext4 rootwait
```

Modify the log output level

```
earlycon=aml_uart,0xfe078000,921600 console=ttyS0,921600 console=tty1 loglevel=1 jtag=apao
root=/dev/mmcblk1p2 rootfstype=ext4 rootwait
```

4.16 Using dtoverlay

REIMEI1 supports the Device Tree overlay function by configuring the boot.conf file.

4.16.1 dtoverlay Configuration Description

Overlay is a patch applied to base dtb to extend or modify it. They are stored as. dtbo files in the overlay subdirectory, and the system will load the .dtbo file from the <load_prefix>overlays/ directory.
The system default base dtb is phantom.dtb, and the default overlay path is /boot/linux/overlays/.

You can add dtoverlay to support by configuring the boot.conf file

```
sudo nano /boot/boot.conf
```

Examples

```
dtoverlay=<overlay_name>,<param=value>
```

It is equivalent to

```
dtoverlay=<overlay_name>
dtparam=<param=value>
```

Open dtdebug debugging information output

```
dtdebug=1
```

After enabling the output of dtdebug debugging information, you will be able to see the dtoverlay loading information at startup through the debug serial port.

NOTE: After modifying the boot.conf file, please execute the sync command to synchronize the changes to the FLASH storage media before restarting the device.

4.16.2 Currently Supported Device Tree Overlay

- Enable onboard uartA serial port

```
[board_type=1]
dtb=phantom.dtb
dtoverlay=uartA
```

After enabling uartA, it corresponds to the /dev/ttyS1 device

NOTE: By default, uartA is used as a debugging serial port. After enabling uartA, it will be used as a normal serial port instead of a debugging serial port.

- Enable onboard uartC serial port

```
[board_type=1]
dtb=phantom.dtb
dtoverlay=uartC
```

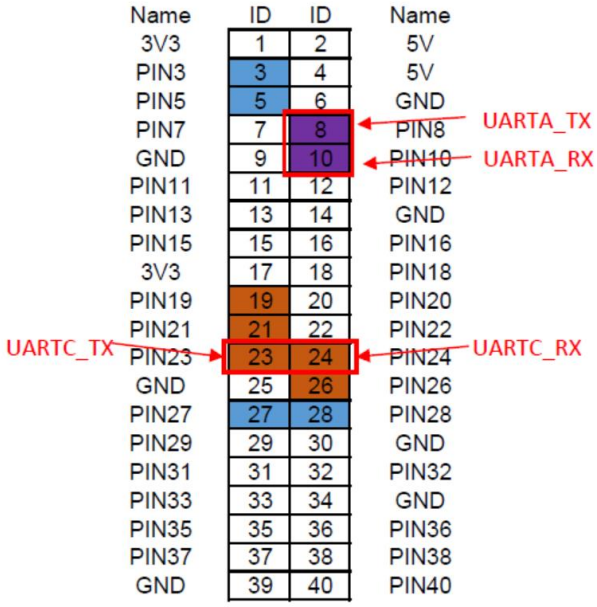
After enabling uartC, it corresponds to the/dev/ttyS2 device

NOTE: The uartC pin of REIMEI1 conflicts with the pin of spi. After enabling the uartC serial port, spi cannot be used.

- Enable uartA and uartC simultaneously

```
[board_type=1]
dtb=phantom.dtb
dtoverlay=uartA
dtoverlay=uartC
```

Pin definitions for uartA and uartC:



- Enable serial port expansion based on SC16IS752 i2c mode

```
[board_type=1]
dtb=phantom.dtb
dtoverlay=sc16is752-i2c,int_pin=72,addr=0x48
```

int_pin corresponds to the actual connected interrupt GPIO pin, and the addr corresponds to the actual i2c device address of the expansion board.

The Serial Expansion HAT is a serial port expansion board based on SC16IS752, which extends 2-way serial ports and 8-way GPIO through the I2C interface.
The Serial Expansion HAT expansion board uses i2c-1, which can be directly connected to the 40PIN of the REIMEI1. The two extended serial port devices are /dev/ttySC0, /dev/ttySC1, and gpiochip1.

- Enable serial port expansion based on SC16IS752 spi mode

```
[board_type=1]
dtb=phantom.dtb
dtoverlay=sc16is752-spi0,int_pin=77
```

int_pin corresponds to the actual connected interrupt GPIO pin

The 2-CH RS232 HAT is a dual channel isolated RS232 expansion board using the SC16IS752+SP3232 scheme.

NOTE: The pin of the 2-CH RS232 HAT expansion board is not compatible with the 40PIN of the REIMEI1. Connecting the expansion board directly to the 40PIN will not work. Please connect the spi0 interface of the REIMEI1 with it by using the DuPont wires.

NOTE: When using the SC16IS752 spi based serial port extension overlay, please do not enable uartC, as uartC will disable the spi0 interface.

The expansion board based on the SC16IS752 supports stacking. When configuring multiple dtoverlays based on the SC16IS752 at the same time, pay attention to the defined interrupt pin int_pin, which cannot be the same name. The extended multi-channel serial port devices are /dev/ttySC0, /dev/ttySC1, /dev/ttySC2, /dev/ttySC3, and so on.

You can view the serial port and gpio corresponding to each expansion board by using the following instructions:

```
sudo cat /sys/kernel/debug/gpio
```

```
gpiochip2: GPIOs 409-416, parent: spi/spi0.0, spi0.0, can sleep:

gpiochip1: GPIOs 417-424, parent: i2c/1-0048, 1-0048, can sleep:

gpiochip0: GPIOs 425-511, parent: platform/fe000000.apb4:pinctrl@4000, periphs-banks:
```

gpiochip0 is the native gpio on board, gpiochip1 corresponds to the GPIO of the I2C extension with the device address 0x48, and gpiochip2 corresponds to the GPIO of the SPI extension. Based on the number of gpiochip, it can be determined that /dev/ttySC0 and /dev/ttySC1 are the serial ports for I2C expansion with the device address of 0x48, and /dev/ttySC2 and /dev/ttySC3 are the serial ports for SPI expansion.

- Enable CAN controller expansion based on mcp2515

```
[board_type=1]
dtb=phantom.dtb
dtoverlay=mcp2515-can0,oscillator=12000000,interrupt=73,spimaxfrequency=2000000
```

oscillator corresponds to the actual crystal oscillator frequency, oscillator=12000000 indicates that the onboard crystal oscillator is 12M, and interrupt corresponds to the GPIO that interrupts the pin connection.

NOTE: mcp2515-can0 and mcp2515-can1 respectively use spiev0.0 and spiev0.1 to extend the CAN controller。

NOTE: Please refer to the 40-Pin GPIO Programming Guide for the corresponding relationship of GPIO.

Using CAN interface

Open can0

```
sudo ip link set can0 down
sudo ip link set can0 type can bitrate 1000000
sudo ifconfig can0 up
```

CAN communication test
Install the can-utils tool

```
sudo apt-get install can-utils
```

Send

```
cansend can0 123#DEADBEEF
```

Receive

```
candump can0 -L
```

5 OS Installation

5.1 Image Download

We have burned the system in eMMC before leaving the factory. Users can skip this section and section 3.4 and use it directly.
We have provided the factory image. If the system is restored to factory settings, please click the following link to download the factory image.

Download link: https://1drv.ms/f/s!Au060HUAtEYBgRl4XvZeFGCVrZvt?e=H91zTs

5.2 System Flash

REIMEI1 supports dual booting of SD card and eMMC system, and SD card has higher priority.

If you want to burn the system to eMMC, you need to start the system through SD card, and then indirectly burn eMMC through dd command.

5.2.1 Flash SD card

Install the flash tool, and recommend balenaEtcher:

- balenaEtcher：https://www.balena.io/etcher/
- SD card: use an SD card with a capacity of at least 8GB (if you plan to burn eMMC with an SD card, the capacity of the SD card should be at least 16GB).

Flashing steps:

1. Open balenaEtcher and select the file to flash.
2. Select the SD card to flash.
3. Wait for the flashing to be completed

Enable SSH:

By default, the image disables ssh function. If you want to connect to the device remotely by SSH after booting, you need to create an empty file named SSH in the boot partition before booting, so as to ensure that the SSH function is automatically enabled after the device boots.

5.2.2 Flash eMMC

At present, eMMC only supports flashing from SD card. By default, the image has been flashed in eMMC when leaving the factory, and users can use it directly. If the device cannot be started and the green indicator light does not flash, it means that the system cannot be started at this time, and the image needs to be flashed into eMMC with SD card.

```
lsblk
```

```
NAME            MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
mmcblk0         179:0    0 14.8G  0 disk
 ├─mmcblk0p1    179:1    0   256M  0 part /boot
 └─mmcblk0p2    179:2    0 14.6G  0 part /
mmcblk1         179:32   0   7.3G  0 disk
 └─mmcblk1p1    179:33   0   7.3G  0 part
mmcblk1boot0 179:64   0     4M  0 disk
mmcblk1boot1 179:96   0     4M  0 disk
```

The partition name of SD card is mmcblk0. You can see that SD card has two partitions, one is mmcblk0p1 and the other is mmcblk0p2.
The second partition is eMMC. Because there is no flashing system by default, there is only one partition mmcblk1p1.
If the second partition has burned the system, the following will be displayed after using the lsblk command.

```
lsblk

NAME            MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
mmcblk0         179:0    0 14.8G  0 disk
 ├─mmcblk0p1    179:1    0   256M  0 part /boot
 └─mmcblk0p2    179:2    0 14.6G  0 part /
mmcblk1          179:32   0   7.3G  0 disk
 ├─mmcblk1p1    179:33   0   256M  0 part
 └─mmcblk1p2    179:34   0   5.9G  0 part
mmcblk1boot0    179:64   0     4M  0 disk
mmcblk1boot1    179:96   0     4M  0 disk
```

Flash preparation
EMMC flashing can only be written through SD card, so first you need an SD card that has flashed the REIMEI system, start the system, and put the system to be burned into the SD card. In the example, the image is directly placed in the folder of the default user phantom, and the absolute path of the folder is /home/phantom.

Flash the system to eMMC:

```
sudo dd if=<image path> of=/dev/mmcblk1 bs=4MiB
#示例
sudo dd if=/home/phantom/phantom_2022-12-03.img of=/dev/mmcblk1 bs=4MiB
sync
```

Wait patiently for the order to be executed.
After the execution, the following contents will be displayed:

```
1483+1 records in
1483+1 records out
```

Using lsblk, we can see that mmcblk1 has two partitions, p1 and p2:

```
lsblk

NAME          MAJ:MIN RM  SIZE  RO TYPE MOUNTPOINT
mmcblk0       179:0    0  14.8G  0 disk
 ├─mmcblk0p1  179:1    0  256M   0 part /boot
 └─mmcblk0p2  179:2    0  14.6G  0 part /
mmcblk1       179:32   0  7.3G   0 disk
 ├─mmcblk1p1  179:33   0  256M   0 part
 └─mmcblk1p2  179:34   0  5.9G   0 part
mmcblk1boot0  179:64   0  4M     0 disk
mmcblk1boot1  179:96   0  4M     0 disk
```

Enable SSH:

SSH service is not enabled for the default image. If you want to connect to the device remotely by SSH when you start the machine, please follow the following steps:

```
sudo mount /dev/mmcblk1p1 /mnt
sudo touch /mnt/ssh
sudo umount /mnt
```

6 Trouble Shooting

6.1 HDMI No Display

There may be no display on HDMI after power-on. At this time, first check whether the screen connection is correct, then you can directly use SSH login interface (how to know the IP address of the device), and then enable the desktop service to see if there is a screen display.

```
sudo systemctl start weston.service
```

6.2 The Device Cannot Startup and Green LED on

This situation is basically because there is no mirror in eMMC and there is no available system in SD card. At this time, you should install and burn SD card with reference to OS Installation, or burn a system for eMMC.

6.3 SSH Not Available

Because the SSH function is disabled by default, if you want to use SSH, please refer to using SSH to connect to the device.

7 FAQ

7.1 Default Username and Password

User name：phantom
password： phantom

7.2 Does It Support Docker Service

The latest system image supports docker service.

7.3 Does it pre-install Linux Header package

The latest system has pre-installed Linux Header package. Please do not install Linux Header package by apt install.

8 Known Issues

At present, the REIMEI1 system is still being optimized. At present, we know that there are the following problems:

#	Category	Description	Comment
1	HDMI	There is no image output on the connected HDMI display.	Currently, some non-standard HDMI monitors are not supported.
2		You can't wake up by mouse or keyboard after the screen automatically turns off.	At present, it is necessary to manually disable the automatic screen blanking function.

9 About Us

9.1 About EDATEC

EDATEC, located in Shanghai, is one of Raspberry Pi's global design partners. Our vision is to provide hardware solutions for Internet of Things, industrial control, automation, green energy and artificial intelligence based on Raspberry Pi technology platform.

We provide standard hardware solutions, customized design and manufacturing services to speed up the development and time to market of electronic products.

9.2 Contact Us

Mail – sales@edatec.cn / support@edatec.cn
Phone – +86-18621560183
Website – https://www.edatec.cn
Address – Room 301, Building 24, No.1661 Jialuo Highway, Jiading District, Shanghai