Geniatech Rockchip Series

Android Software Development Instruction

Preface

Overview:

The document is for Geniatech Rockchip series Android software development instruction, it can help software development engineer and technician engineer make development and testing better, the updated products development instruction of Geniatech Rockchip will be updated based on this document persistently.

Product Version：

Product name	Kernel version	Buildroot version
XPI-RK3128	Linux3.10	Android7.1.2

Reader Object:

This document is mainly for software development engineers.

Revision Recording:

Date	Version	Author	Approved by	Modification Instruction
2020-11-23				

Directory

Preface
Directory
1. Support list
2. Document/tool index
3. SDK software construction
4. Edition environment Set up
5. Installation Edition tool
6. SDK Edition
7. SDK Mirror burning
8. GPIO Control
9. I2C Control
10. UART set up and Invoking
11. Hardware specification

1.Support List

1.1Product list

SoC	Model No	Software doc support	Function instruction
Rk3128	XPI-3128	Yes	Support DDR3/DDR3L/LPDDR2

1.2Document development support list:

SoC	H/W type	Function instruction
xpi-rk3128	Dev board	hdmi out、ent、usbx4、wifi、rtc、extend 40Pin ribbon cable

2.Document/tool index

2.1 Document Index:

This Rockchip Linux SDK doc is to help developer to make developing and tuning quickly, the involved content in Document doesn't include all knowledge and questions, document list also will be updated continuingly , if you have any questions and requirement, pls contact me at any time.

Rockchp Android SDK mainly includes Android source code, tool chain, based C library, simulation environment and development environment.

2.2 Tool Index:

The Rockchip Android SDK released tool is used to develop/tune and MP stage, tool version will be updated following SDK updating, if you have any questions and requirement for tool, pls contact us. In Tool directory ,Rockchip Android SDK also includes using tool in Android O/S, Windows O/S.

3.SDK Software Construction

3.1 SDK directory and construction

Android source code root directory	Description
abi	Appliaction binary interface
art	New ART running environment
bionic	System C library
bootable	Start guidance related code
build	Store system compiling rule and generic based development kits configuration
cts	Android compatibility testing suite standard
dalvik	Dalvik machine
developers	Developer directory
development	Appliaction development relation
device	Device related configuration
docs	Referenced document directory
external	Open source module related files
frameworks	Application frameworks, Android system kernel is edited by Jave and C++
hardware	Hardware abstract level code
IMAGE	Compiling generated directory is used to store firmware.
kernel	Dirver and DTS etc…
libcore	Kernel library files
libnativehelper	Dynamic library is JNI library's base.
ndk	NDK code, help developer to embed C/C++ source code in application
out	After compiling, source code is exported through this directory.

packages	Application package
pdk	Plug Development Kit abbreviation, local development suite.
platform_testing	Platform testing
prebuilts	Pre-edit some resource in X86 and ARM
RKDocs	RK related development document
RKTools	Some RK tool, e.g. burning tool, production tool
rockdev	generated firmware by editing.
sdk	SDK and simulated machine.
system	bottom file system library ,application and components.
toolchain	Tool chain files
tools	Tool files
u-boot	Boot guidance related code
vendor	Manufacturer customization code
Makefile	Overall Makefile files is used to define editing rules.

For above several folders, device, build, out, packages, vendor, kernel and frameworks folders often are used during development.

For device: it includes some configuration files, e.g. WiFi, ADB port, you can add property and set some parameters once this port is set to fixed parameters.

For Build, it includes some editing rules, e.g. you can set up some application installation.

For Out: it includes some generated varied files after editing, e.g. all kinds of system applications, these will be installed in Android deivces.

For Package: it includes many Android projects source code, in mobile phone, you can see some clickable icons, actually it is a separated Android item.

4. Edition Environment Set up

4.1.1 JDK installation

Android 7.1 system edition depends on Java 8, before editing, you need to install OpenJDK, installation command is as below:

sudo apt-get install openjdk-8-jdk

Configure Java environment variables, e.g. installing route: /usr/lib/jvm/java-8-openjdk-amd64, you can execute in terminal command and configure environment variables.

export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64

export PATH=$JAVA_HOME/bin:$PATH

export CLASSPATH=.:$JAVA_HOME/lib:$JAVA_HOME/lib/tools.jar

SDK includes Open JDK8 script files, it is named javaenv.sh in engineering root directory.

You can execute below command, and set JDK:

source javaenv.sh Linux

4.1.2 VMware Workstation installation ubuntu

4.1.2.1 You can download ubuntu-14.04.5-desktop-amd64.iso file in www.ubuntu.com

4.1.2.2 Build Virtual machine

Open the Vmware Workstation, click on the graph of the "Create a New Virtual Machine", and open the "New Virtual Machine Wizard".
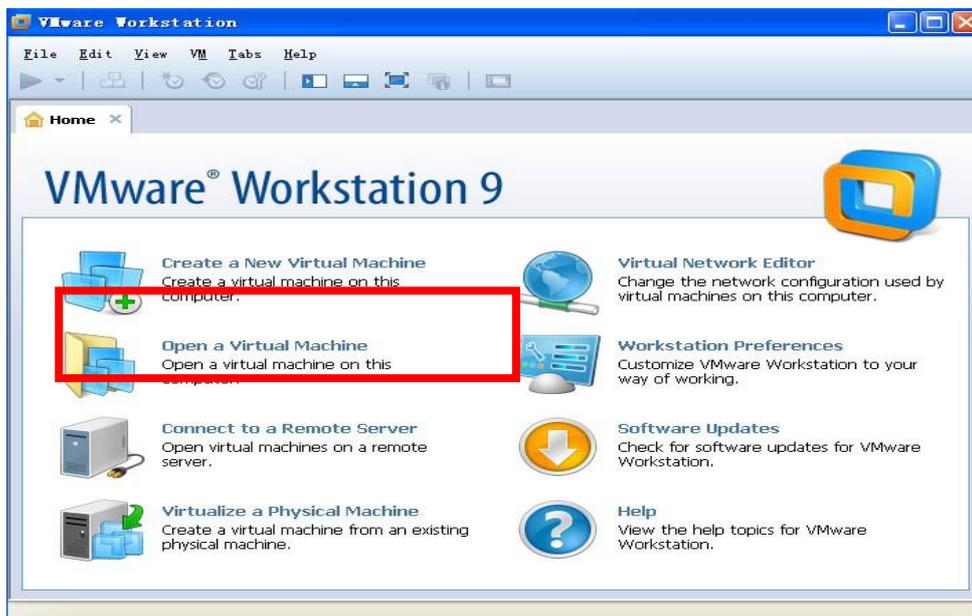
Default select "Typical" configuration and click on "Next" button, and then default select the "I will install the operating system later" item.
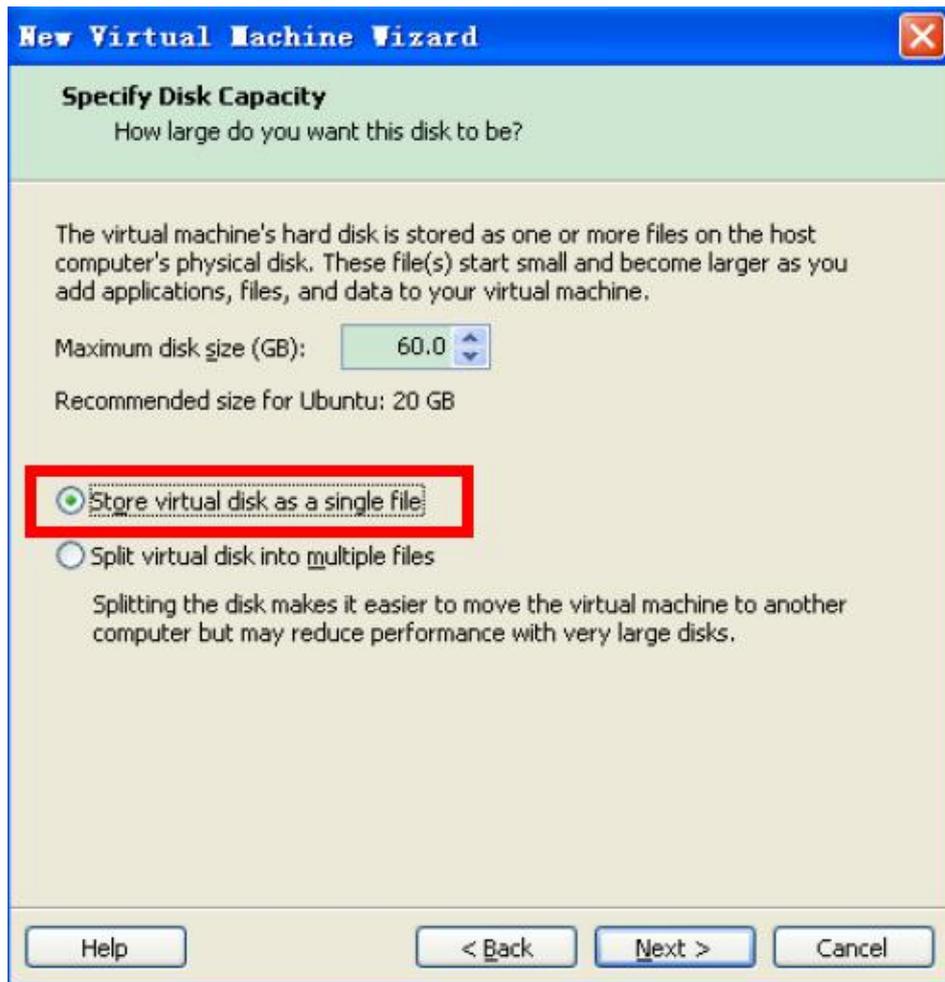
Click on the "Next" button to select a guest operating system, as in the following figure.
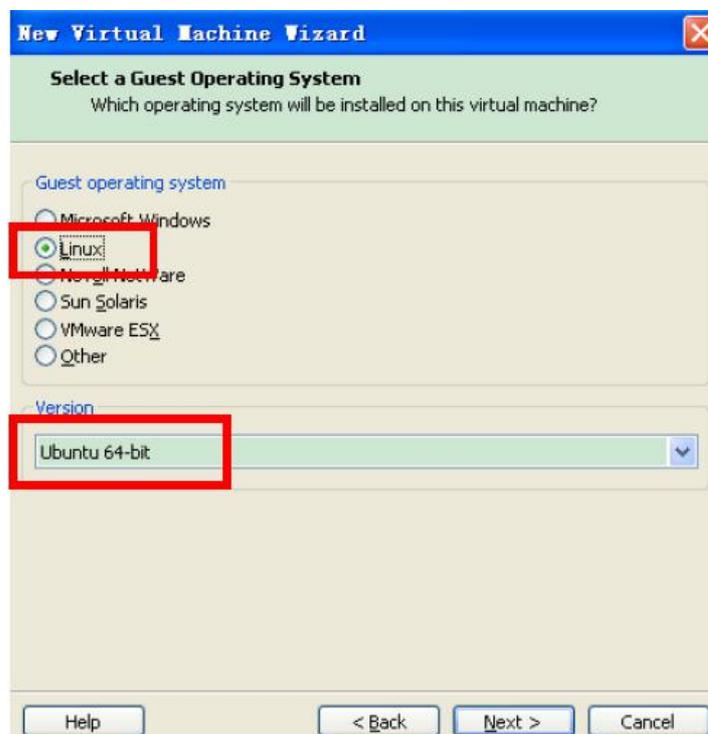
Click on the "Next" button, set the virtual machine name and path as you like

Click on the "Next" button, specify disk capacity, in general 60G is enough. Here select "Store virtual disk as a single file" item.

Click on the "Next" button, confirm setup information, after that click on the "finish" button to complete the configuration.

Now click on the "Edit virtual machine settings", choose "Harware" tab, and set the memery to 2GB or more. You'd better set the processor to 2.

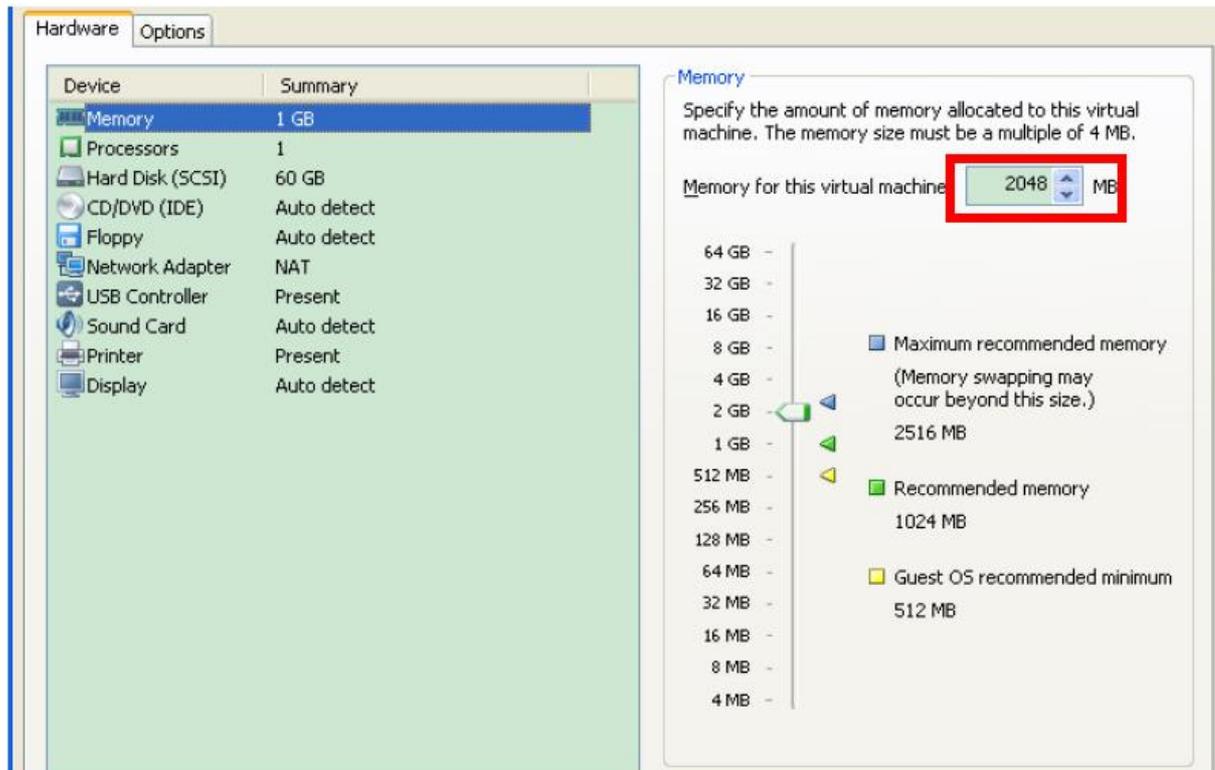select "CD/DVD(IDE)" item on the hardware tab, afterwards choose the path to the ubuntu-12.04-desktop-amd64.iso image file in the right side bar, and then select "Network Adapter" item to choose bridged network connection for the convenience of virtual machines and host access, just as followings. Click on the "ok" button to complete this configuration.
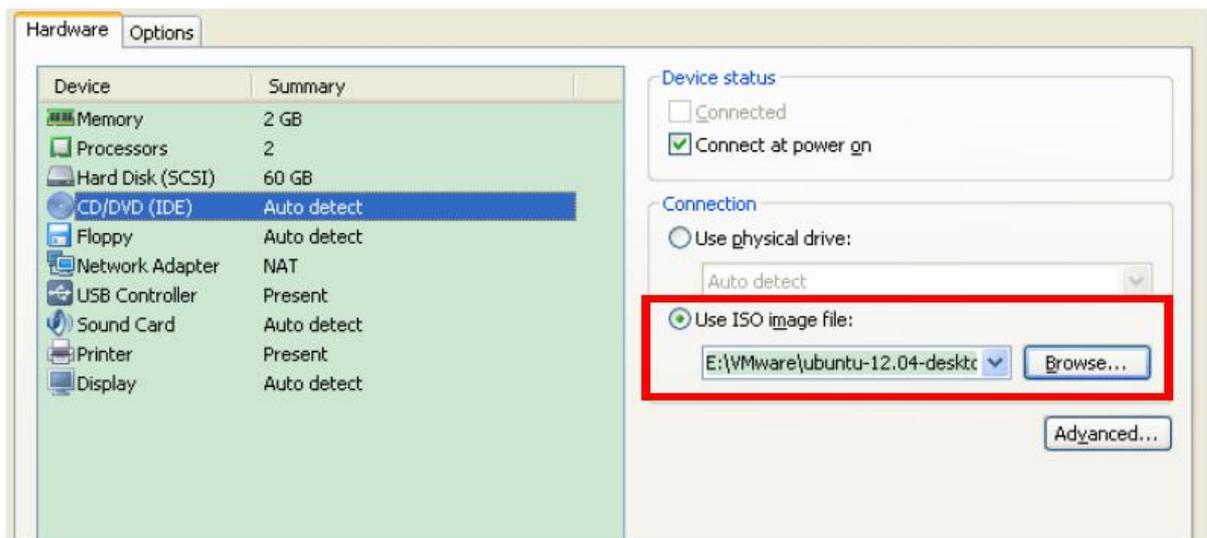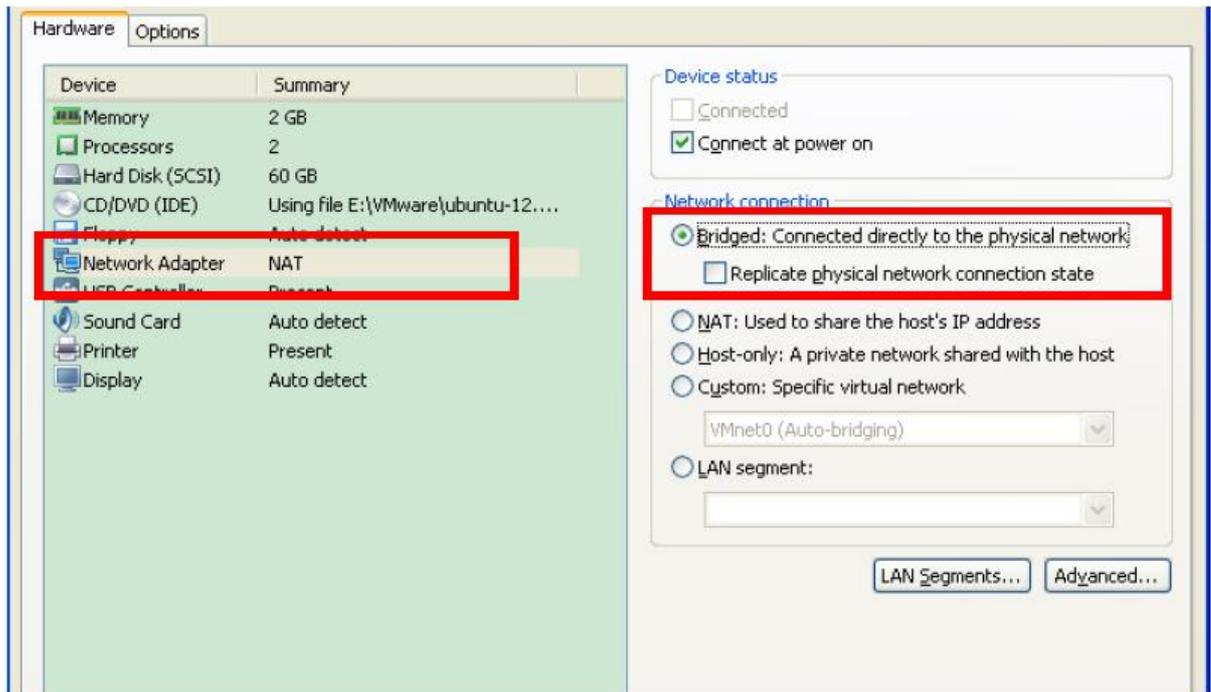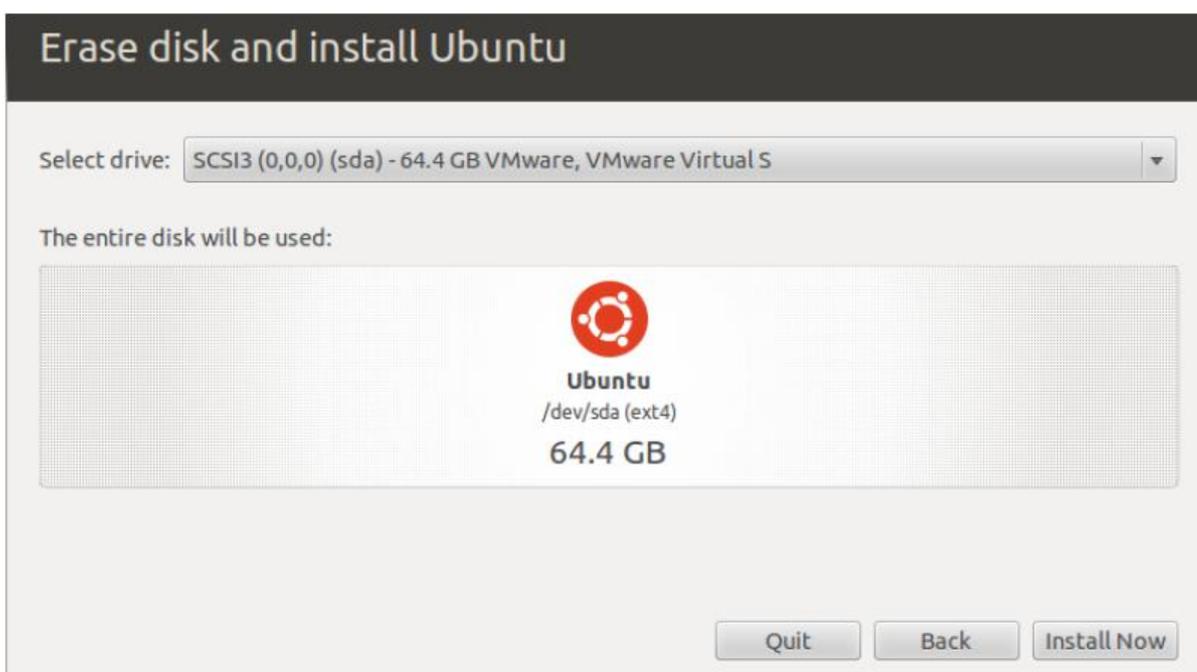
Click on "Power on this virtual machine" to start to install ubuntu operating system. Default select English language and press "Intall Ubuntu" to install the ubuntu operating system.
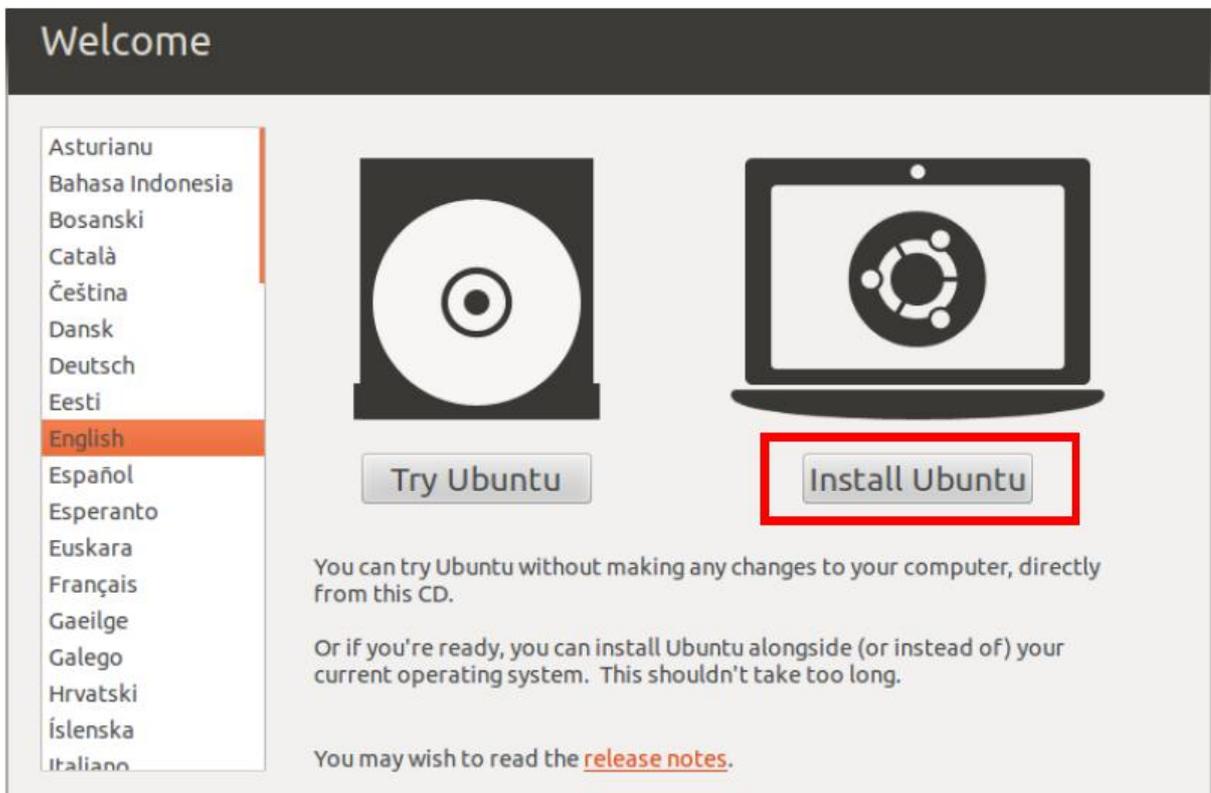
Click on "Continue" button to prepare to install Ubuntu. And prepare disk space, default select "Erase disk and install Ubuntu" item, and then click on the "Continue" button. Select the default driver and click on "Install Now" button to start to install the Ubuntu.

Installation type

This computer currently has no detected operating systems. What would you like to do?

○ **Erase disk and install Ubuntu**
Warning: This will delete any files on the disk.

○ **Something else**
You can create or resize partitions yourself, or choose
multiple partitions for Ubuntu.

Quit   Back   Continue

Erase disk and install Ubuntu

Select drive: SCSI3 (0,0,0) (sda) - 64.4 GB VMware, VMware Virtual S ▼

The entire disk will be used:

Ubuntu
/dev/sda (ext4)
64.4 GB

Quit   Back   Install Now

Choose a positon where you are, and click on "Continue" button. Select the default
keyboard layout, just like below figures.

After that set your user name and password, click on the "Continue" button, and then pop up the installing interface. Please wait a moment for this process.

After installation, click on "Restart Now" button to restart into the Ubuntu login screen with the user name and password you set before.

5. Install compilation tools

5.1Install some necessary packages, which will be used later in the compilation process. You can enter the following commands in the terminal. During this process, enter "Y" to continue.

$ sudo apt-get install git-core gnupg flex bison gperf build-essential zip curl libc6-dev libncurses5-dev x11proto-core-dev libx11-dev libreadline6-dev libgl1-mesa-glx libgl1-mesa-dev g++-multilib mingw32 tofrodos python-markdown libxml2-utils xsltproc zlib1g-dev lib32z1 libxml2-utils lzop

```
root@jolie-ubt:/home/jp# apt-get install git-core gnupg flex bison  gperf build-e
ssential zip curl libc6-dev libncurses5-dev x11proto-core-dev libx11-dev libreadl
ine6-dev libgl1-mesa-glx libgl1-mesa-dev g++-multilib mingw32 tofrodos python-mar
kdown libxml2-utils xsltproc zlib1g-dev
```

5.2Download the cross compilation tool through the link as below

http://www.geniatech.net/down-eng/tools/aarch64-linux-gnu.tar.xz.zip

5.3 Switch to the directory where the file you just downloaded, directly delete the .zip of the file name, without decompression, and then command to decompress the aarch64-linux-gnu.tar.xz file, and configure the environment variables as follows:

$ xz -d aarch64-linux-gnu.tar.xz

$ tar -xvf aarch64-linux-gnu.tar

$ export PATH=/opt/aarch64-linux-gnu/bin:$PATH

5.4 Jack-server configuration

The Android7.1 system uses jack-server as the java code compiler, and the following similar errors may be encountered during the compilation process:

Jack server already installed in "/home/yhx/.jack-server"

Communication error with Jack server (1), try 'jack-diagnose' or see Jack server log

Communication error with Jack server 1. Try 'jack-diagnose'

Communication error with Jack server 1. Try 'jack-diagnose'

This situation is mainly due to the limitation of the jack-server compiler, and the same network port number cannot be used by multiple users at the same time.That is, in the process of collaborative development on the server, when multiple users compile Android 7.1 at the same time, they need to configure each to use a different network port number.

The two configuration files of jack-server (yhx is the user name of the corresponding user) determine the port number it uses:

/home/yhx/.jack-server/config.properties

/home/yhx/.jack-settings

These two configuration files need to be configured with two port numbers, namely the server port number and the client port number. The port numbers in the two configuration files must match.

jack.server.service.port=8074

jack.server.admin.port=8075

And

SERVER_PORT_SERVICE=8074

SERVER_PORT_ADMIN=8075

The configuration steps are as follows:

5.4.1 Make sure that two configuration files are existed and the permissions are set to 0600：

chmod 0600 /home/yhx/.jack-server/config.properties

chmod 0600 /home/yhx/.jack-settings

5.4.2 If the two configuration files do not exist, please refer to the following text to create these two configuration files.

config.properties The file example is as follows (the port number needs to be modified according to the actual situation)：

jack.server.max-jars-size=104857600

jack.server.max-service=4

jack.server.service.port=8074

jack.server.max-service.by-mem=1\=2147483648\:2\=3221225472\:3\=42949

67296

jack.server.admin.port=8075

jack.server.config.version=2

jack.server.time-out=7200

.jack-settings The file example is as follows (the port number needs to be modified according to the actual situation)：

Server settings

SERVER_HOST=127.0.0.1

SERVER_PORT_SERVICE=8074

SERVER_PORT_ADMIN=8075

Internal, do not touch

SETTING_VERSION=4

5.4.3 To modify the port number, please change the service port and admin port to other port numbers. The port numbers in the two configuration files need to match. Examples are as follows:

jack.server.service.port=8023

jack.server.admin.port=8024

SERVER_PORT_SERVICE=8023

SERVER_PORT_ADMIN=8024

5.4.4 Recompile Android to see if it reports an error. If it still reports an error, please try to change other port numbers until the compilation passes.

5.4.5 If the compilation fails after 5 changes, you can execute the jack-admin dump-report command to decompress the compressed package generated by the command and analyze the log log. If the following log appears, you can reinstall libcurl:

$ JACK_EXTRA_CURL_OPTIONS=-v jack-admin list server

* Protocol https not supported or disabled in libcurl

* Closing connection -1

Communication error with Jack server 1. Try „jack-diagnose“

6.SDK Compile

The SDK is compiled in userdebug mode by default. When using ADB, you need to execute adb root first to make the shell obtain root privileges, and then perform other operations like adb remount, adb push, etc

6.1 Uboot compile

■ XPI3128

$make rk3128_box_defconfig

$make -j12

After compiling, two files rk3128_loader_v2.05.240bin and uboot.img will be generated

6.2 Kernel compile

■ XPI3128

$makerockchip_defconfig

$make rk3128-box-rk88.img

After the compilation is complete, the kernel root directory will generate two image files, kernel.img and resource.img.

6.3 Android compilation and firmware generation

■ XPI3128

After the customer configures the JDK environment variables according to the actual compilation environment, follow the steps below and execute make.

$ source build/envsetup.sh

$ lunch rk3128_box-userdebug

$ make -j4

After compiling, execute the mkimage.sh script in the root directory of the SDK to generate the firmware, and all the images required for programming will be copied to the rockdev/Image-RK3128_box directory.

Or directly use the script to compile, the command is as follows:

$./device/rockchip/rk312x/build-rk3128_box.sh

7.SDK image burning

7.1 Overview

This chapter mainly introduces the process of how to program the completed image file (image) and run it on the hardware device. Several mirror programming tools provided by Rockchip platform are introduced as shown in the following table. You can choose a suitable programming method for programming. Before programming, you need to install the latest USB driver, see 4.3.2 USB Driver Installation for details.

tool	system	description
Rockchip Development Tools	**Windows**	Discrete upgrade firmware and the entire update upgrade firmware tool

7.2 Introduction to programming mode

Several modes of Rockchip platform hardware operation are shown in the table below. Only when the device is in Maskrom and Loader mode, can the firmware be programmed or the on-board firmware can be updated.

| mode	Tool burn	instruction
Maskrom	support	When Flash has not burned the firmware, the chip will boot into Maskrom mode, and the firmware can be burned for the first time; during development and debugging, if the Loader fails to start normally, you can also enter Maskrom mode to burn the firmware.
Loader	support	In Loader mode, firmware can be programmed and upgraded. A certain partition image file can be programmed separately through the tool to facilitate debugging.

7.3 Rockchip Development Tool Programming

For the programming instructions, please refer to "Rockchip Development Tools Manual.pdf" under the docs\RKTools manuals directory.

The SDK provides programming tools, as shown below. After compiling and generating the corresponding firmware, enter the programming mode and you can flash the machine. For machines that have already burned other firmware, you can choose to burn the firmware again, or choose a low-level device, erase idb, and then flash.
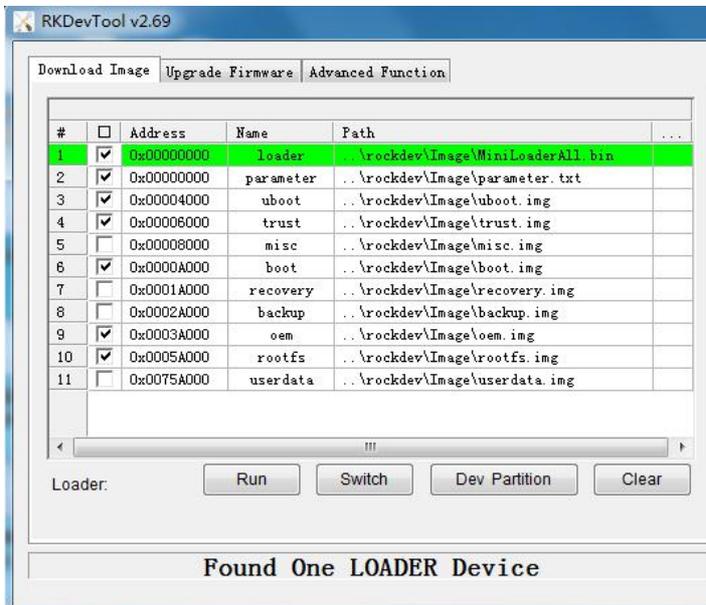
7.3.1 Enter programming mode

Take xpi-rk3128 as an example: first press and hold the button next to the infrared head, power on, and connect to the USB port for programming.
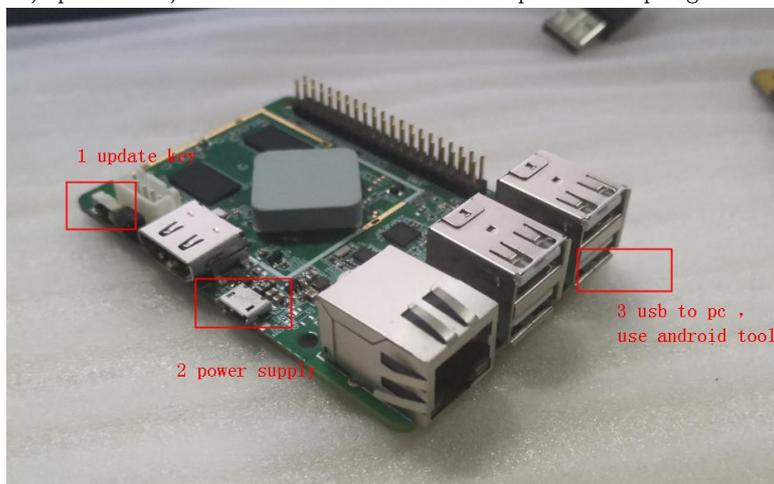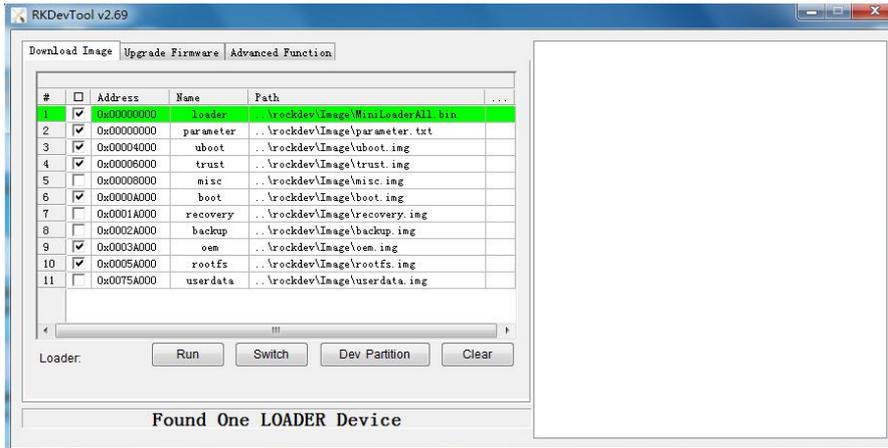
7.3.2 Download mirror

1) USB connection hardware board enters programming mode.

2) Open the tool and click the Download Mirror menu, and click on the… column will pop up a file selection box, you can select the img of the corresponding partition, and configure several other items at once. geniatech rockchip series selection: 1, 2, 3, 4, 6, 9, 10

3) After the configuration is complete, click "Execute" and you can see the blank box on the right to enter the download prompt. The "Low" button is used to erase the device, and the "Clear" button is to clear the text in the edit box.
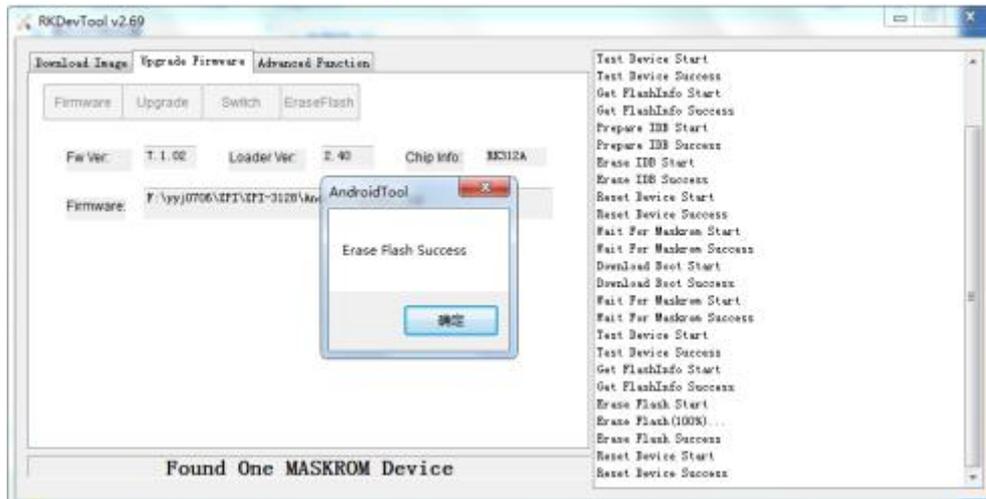
7.3.3 Upgrade firmware (update.img exists only in buildroot)
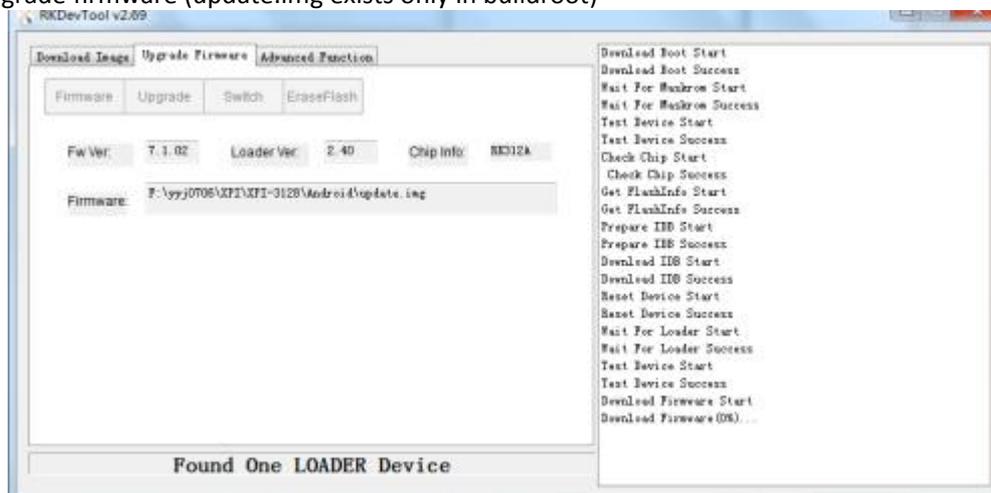
1) Click on the firmware to select the newly packaged update.img file, and click the upgrade button to download. (Note that the device must be in download mode).

8.GPIO control

8.1 Driver files and DTS configuration

The location of the driver file: kernel/drivers/pinctrl/pinctrl-rk3368.c (note that because this set of drivers is inherited from rk3368, the pinctrl driver is the name.)

Drive DTS node to configure pinctrl, the driver will analyze and store the 3 sets of pinctrl configuration corresponding to "default", "idle", and "sleep" for use in specific occasions. Three commonly used states are given in Pinctrl-state.h: The default state represents the state of the device when it is active, which is generally configured in the .resume of the device driver. In addition, the pin will also be set to the default state at startup.

The idle state represents the pin state that needs to be configured when the system is idle, and the system does not enter deep sleep at this time.

The sleep state represents the pin state when the system is in deep sleep, which is generally configured in the .suspend of the device driver. For example, the configuration of the "default" group of pins will be configured in the register when the probe is driven, and the configuration of other groups need to be parsed out in the code, and then switch to use. For example, use in the DTS configuration and code of HDMI pinctrl:

hdmi: hdmi@ff3c0000 {

compatible = "rockchip,rk322xh-hdmi";

reg = <0x0 0xff3c0000 0x0 0x20000>,

<0x0 0xff430000 0x0 0x10000>;

pinctrl-names = "default", "gpio";

pinctrl-0 = <&hdmi_cec &hdmii2c_xfer &hdmi_hpd>;

pinctrl-1 = <&i2c3_gpio &hdmi_cec>;

rockchip,grf = <&grf>;

rockchip,hdmi_audio_source = <0>;

rockchip,hdcp_enable = <0>;

rockchip,cec_enable = <0>;

status = "disabled";

};

Used in the code:

The driver analyzes the gpio state and switches to gpio mode:

gpio_state = pinctrl_lookup_state(hdmi_dev->dev->pins->p, "gpio");

pinctrl_select_state(hdmi_dev->dev->pins->p, gpio_state);

default 状态不需要驱动解析，直接切换成 default 模式：

pinctrl_select_state(hdmi_dev->dev->pins->p,

hdmi_dev->dev->pins->default_state);

8.2 IOMUX configuration

The iomux configuration is to switch the mux value corresponding to the pin, which has the following definitions, which correspond to the corresponding
Register value:

#define RK_FUNC_GPIO 0

#define RK_FUNC_1 1

#define RK_FUNC_2 2

```
#define RK_FUNC_3 3
#define RK_FUNC_4 4
#define RK_FUNC_5 5
#define RK_FUNC_6 6
#define RK_FUNC_7 7
```

The following is still an example of hdmii2c_xfer; first, we find it in the GRF chapter of trm of 3228H

The two pins i2c3hdmi_scl and i2c3hdmi_sda correspond to gpio0a5 and gpio0a6. Both function pins are binary '01' as the function value, that is, use RK_FUNC_1;

How to modify if the pin pin defined in the hardware schematic diagram is different from the given reference code or the mux value is different. Suppose there is a specific product that uses i2c2 to connect to HDMI for communication function.

The rk3228h-xxx.dts quoted coverage to achieve, the following is an example:

```
&hdmi_rk_fb {
status = "okay";
pinctrl-names = "default", "gpio";
pinctrl-0 = <&i2c2_xfer &hdmi_cec>;
pinctrl-1 = <&i2c2_gpio>;
};
```

8.3 Drive strength configuration

The drive strength configuration, that is, the drive strength current value corresponding to the configuration, respectively corresponds to the corresponding register value, similar to the usage of mux, the following is an example:

```
pcfg_pull_down_12ma: pcfg-pull-down-12ma {
bias-pull-down;
drive-strength = <12>;
};
```

If you want to increase or decrease the drive strength, but it is different from the drive strength defined by dtsi, how to modify it. It is similar to the modification of mux, after being quoted in the product DTS file, the modification is overwritten. Each pin has its own corresponding drive current intensity range, so when configuring it, select its effective configurable current value. If it is not the effective current value corresponding to the pin, the configuration will be wrong and cannot take effect.

9.I2C control

The I2C (Inter-Integrated Circuit) bus is a two-wire serial bus developed by PHILIPS, which is used to connect the microcontroller and its peripherals; the I2C bus controller uses the serial data (SDA) line and the serial clock (SCL)) The wire transfers information between devices connected to the bus. Each device has a unique address identification (whether it is a microcontroller-MCU, LCD driver, memory or keyboard interface), and can be used as a transmitter or receiver (determined by the function of the device).

Rockchip I2C controller supports the following functions:

● Compatible with I2C and SMBus bus
● Support I2C bus in master mode
● Software programmable clock frequency and transmission rate up to 1000kbps
● Support 7-bit and 10-bit addressing modes
● Interrupt or poll up to 32 bytes of data transmission at a time
● Clock stretching and waiting state

The following figure shows the hardware connection method of the I2C bus. A pull-up resistor is required. Changing the size of the pull-up resistor can adjust the pull-up strength of the I2C bus:

I²C 总线连线图
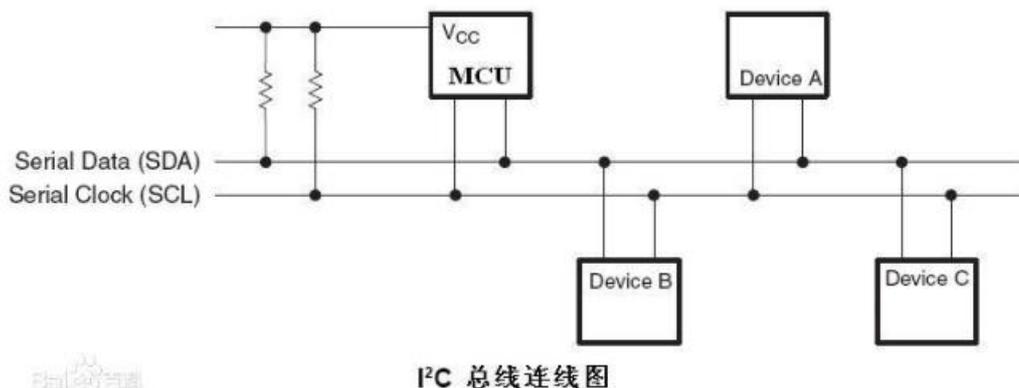
9.1dts node configuration

You can refer to the files in the Linux kernel directory:

Documentation/devicetree/bindings/i2c/i2c-rk3x.txt Dts node configuration can refer to the file in the

Linux kernel directory: Documentation/devicetree/bindings/i2c/i2c-rk3x.txt

9.2 Need configuration items:

9.2.1 I2C rate configuration: Please refer to the datasheet of the device to determine the adaptable clock, generally 400k, 100k (default, optional), 200k, 1000k;

400k example:

clock-frequency = <400000>;

9.2.2 i2c_clk rising edge time, falling edge time;

When the I2C rate configuration is required to exceed 100k, the rising edge and falling edge time of i2c_clk must be measured by an oscilloscope; because the I2C protocol standard has regulations on the rising edge and falling edge time, especially the rising edge time, if it exceeds the protocol specified The maximum value, the I2C communication may fail. The following is the rising and falling edge time range specified by the protocol:

Symbol	Parameter	Standard-mode		Fast-mode		Fast-mode Plus		unit
		Min	Max	Min	Max	Min	Max	
fSCL	SCL clock frequency		100		400		1000	KHZ
Tr	rise time of both SDA and SCL signals		1000	20	300		120	ns
Tf	fall time of both SDA and SCL signals		300	20× (VDD/5.5V	300	20× (VDD/5.5V	300	ns

Rising edge Tr, falling edge Tf, respectively take 30%~70% of the waveform time:
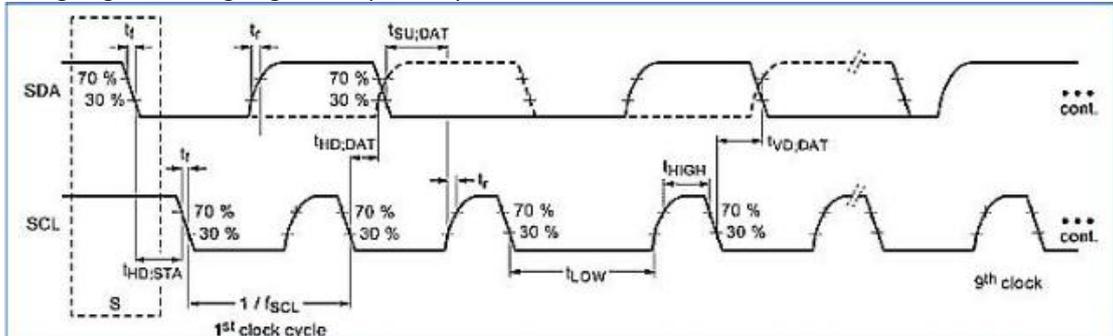
9.2.3 If the above two items are not configured, the calculation will be based on the max value of the rising edge and the falling edge by default, and the clk speed obtained will be nearly 90k. The default max value of 100k is 1000ns, which is in line with most hardware, so if the requirement is not high, it can not be configured.

9.2.4 The i2c1+es8316 codec example shows that a 400k i2c clock is needed, and the oscilloscope measured Tr=164ns, Tf=15ns:

&i2c1 {

status = "okay";

i2c-scl-rising-time-ns = <164>;

i2c-scl-falling-time-ns = <15>;

clock-frequency = <400000>;

es8316: es8316@10 {

#sound-dai-cells = <0>;

compatible = "everest,es8316";

reg = <0x10>;

clocks = <&cru SCLK_I2S_8CH_OUT>;

clock-names = "mclk";

spk-con-gpio = <&gpio0 11 GPIO_ACTIVE_HIGH>;

hp-det-gpio = <&gpio4 28 GPIO_ACTIVE_LOW>;

};

};

10 Serial port settings and calls

If you are developing U-Boot or kernel, the USB serial adapter (short for USB-to-serial TTL adapter) is very useful for checking the system boot log, especially when there is no graphical desktop display.

uart2 is generally used as a debugging serial port, but there is also multiplexing, which means that the TF card and the debugging serial port cannot be used at the same time:

● SDMMC_D0/UART2_TX

● SDMMC_D1/UART2_RX

10.1 Rockchip Uart features

UART (Universal Asynchronous Receiver/Transmitter), the following are some features supported by linux 3.10 uart driver:

Up to 4M baud rate

Support interrupt transmission mode

Support DMA transfer mode

Support 5, 6, 7, 8 data bits

Support automatic flow control

10.2 DTS node configuration

Serial DTS node:

uart0: serial@ff110000 {

compatible = "rockchip,serial";

reg = <0x0 0xff110000 0x0 0x100>;

interrupts = <GIC_SPI 55 IRQ_TYPE_LEVEL_HIGH>;

clock-frequency = <24000000>;

clocks = <&clk_uart0>, <&clk_gates16 11>;

clock-names = "sclk_uart", "pclk_uart";

reg-shift = <2>;

reg-io-width = <4>;

dmas = <&pdma 2>, <&pdma 3>;

#dma-cells = <2>;

pinctrl-names = "default";

pinctrl-0 = <&uart0_xfer &uart0_cts &uart0_rts>;

```
status = "disabled";

};
```

Add the following code to the board-level DTS file：

```
&uart0 {

dma-names = "tx", "rx"; Enable DMA receive and send

status = "okay";

};
```

Enable after /dev/ttys0 devices can be used

10.3Change the serial port for printing logs

```
fiq_debugger: fiq-debugger {
        compatible = "rockchip,fiq-debugger";
        rockchip,serial-id = <2>; Set the serial port ID

        rockchip,signal-irq = <182>;
        rockchip,wake-irq = <0>;
        rockchip,irq-mode-enable = <1>; /* If enable uart uses irq instead
of fiq */
        rockchip,baudrate = <1500000>; /* Only 115200 and 1500000 */
pinctrl-names = "default";
        pinctrl-0 = <&uart2c_xfer>; configuration  iomux
};
```

The above required information can be obtained through the DTS node of the serial device, for

example:

```
uart1: serial@ff190000 {

compatible = "rockchip,rk3399-uart", "snps,dw-apb-uart";

reg = <0x0 0xff190000 0x0 0x100>;

clocks = <&cru SCLK_UART1>, <&cru PCLK_UART1>;

clock-names = "baudclk", "apb_pclk";

interrupts = <GIC_SPI 98 IRQ_TYPE_LEVEL_HIGH>;

dmas = <&dmac_peri 2>, <&dmac_peri 3>;

dma-names = "tx", "rx";

reg-shift = <2>;

reg-io-width = <4>;

pinctrl-names = "default";

/*pinctrl-0 = <&uart1_xfer>;*/

status = "disabled";

};
```

Note, to ensure that the CLK of other serial ports is not turned off, otherwise the switching process may
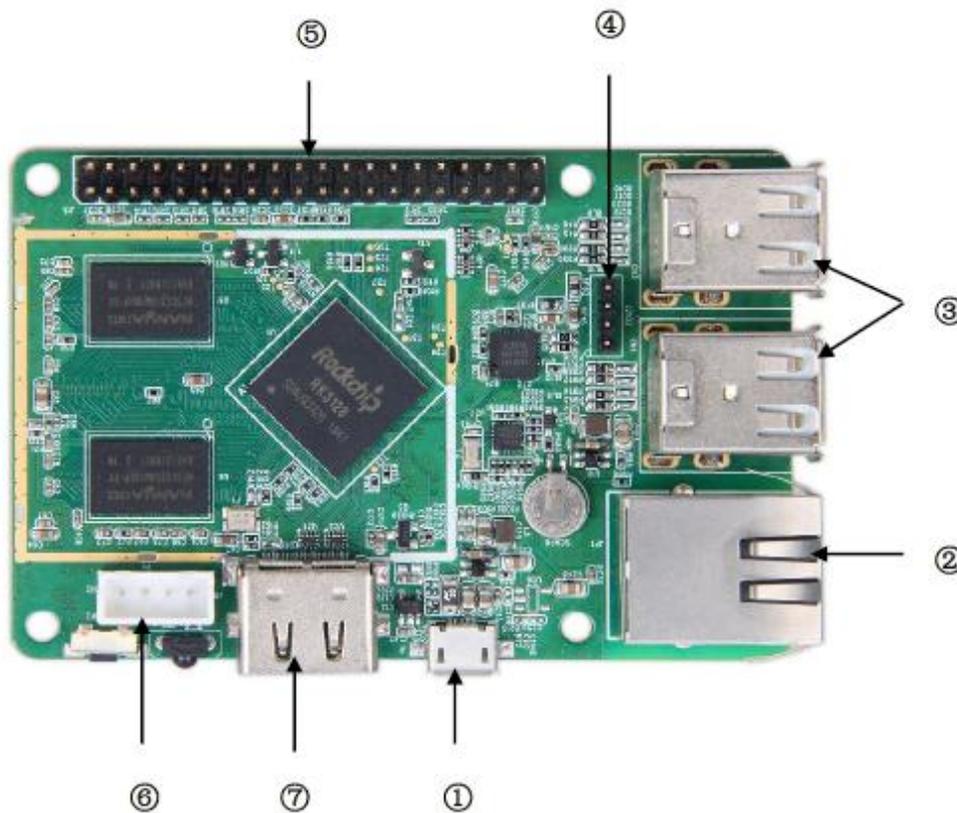
cause a crash.

10.4 Debug serial device

It is best not to use commands such as echo cat to debug the serial port device rudely, it is better to use the tested apk software, or contact our FAE to obtain the ts_uart test bin file
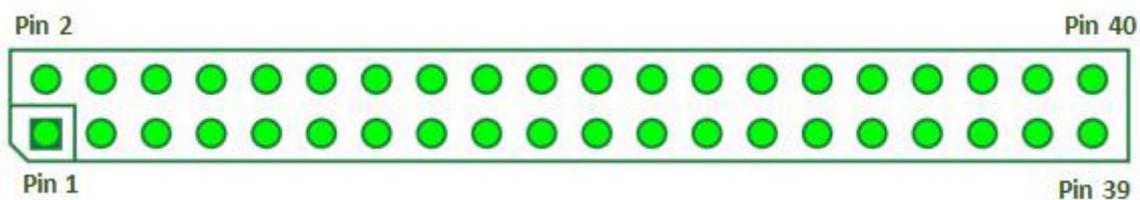
• 11. Hardware Information

11.1 Interface definition

11.1.1 Frontal perspective

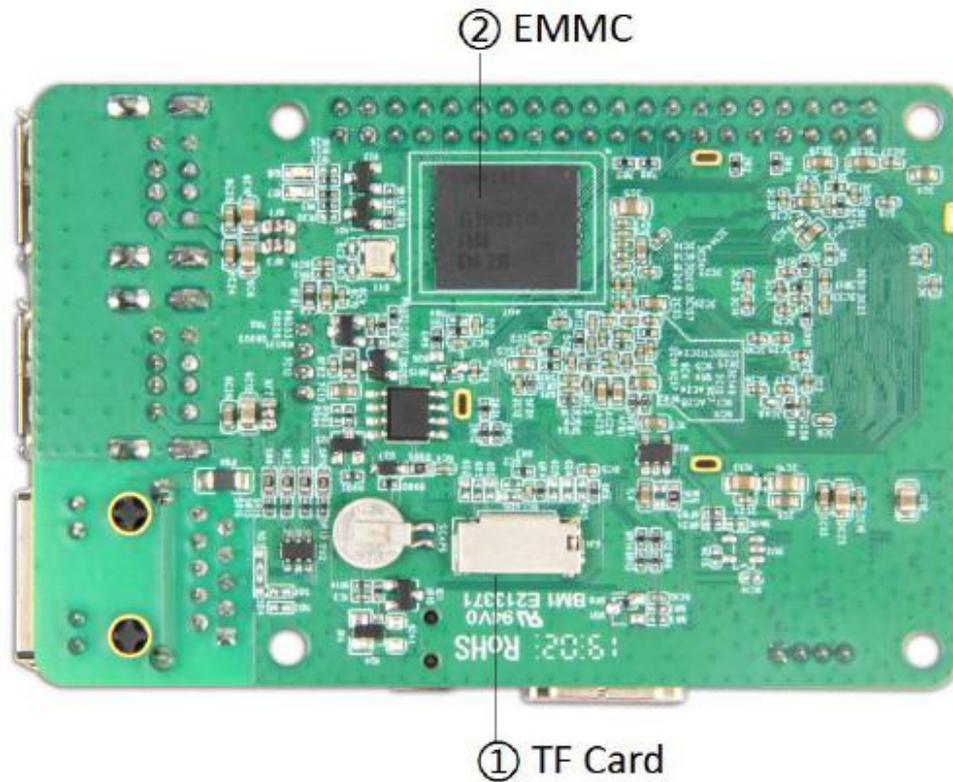| No.	Name	Description
1	DC IN	5V/2A
2	LAN	Ethernet 10/100M
3	USB HOST	USB HOST 2.0*4(support USB upgrade)
4	WIFI Jack	WIFI Jack*1
5	Extension GPIO	Extension GPIO
6	UART	UART*1
7	HDMI OUT	HDMI out*1

Extension GPIO			
PIN	Default function	PIN	Default function
1	VCC_IO_3V3	2	VCC_5V
3	I2C_SDA_A	4	VCC_5V
5	I2C_SCK_A	6	GND
7	GPIO3_C1	8	UART_TX_B
9	GND	10	UART_RX_B
11	UART_RX_A	12	PWM0
13	GPIO0_D6	14	GND
15	SPDIF_OUT	16	I2S_SDI
17	VCC_IO_3V3	18	I2S_SDO
19	PCM_OUT	20	GND
21	PCM_IN	22	I2S_IN
23	PCM_FS	24	PCM_CLK
25	GND	26	UART_TX_A
27	I2C_SDA_B	28	I2C_SCK_B
29	GPIO3_C7	30	GND
31	GPIO0_B4	32	IR
33	PWM1	34	GND
35	SYS_LED	36	I2S_MCLK
37	PWM2	38	I2S_SCLK
39	GND	40	I2S_LRCLK

11.1.2 back v i e w

No.	Name	Description
1	TF Card Slot	TF card *1(max. 64GB)
2	EMMC	8G EMMC