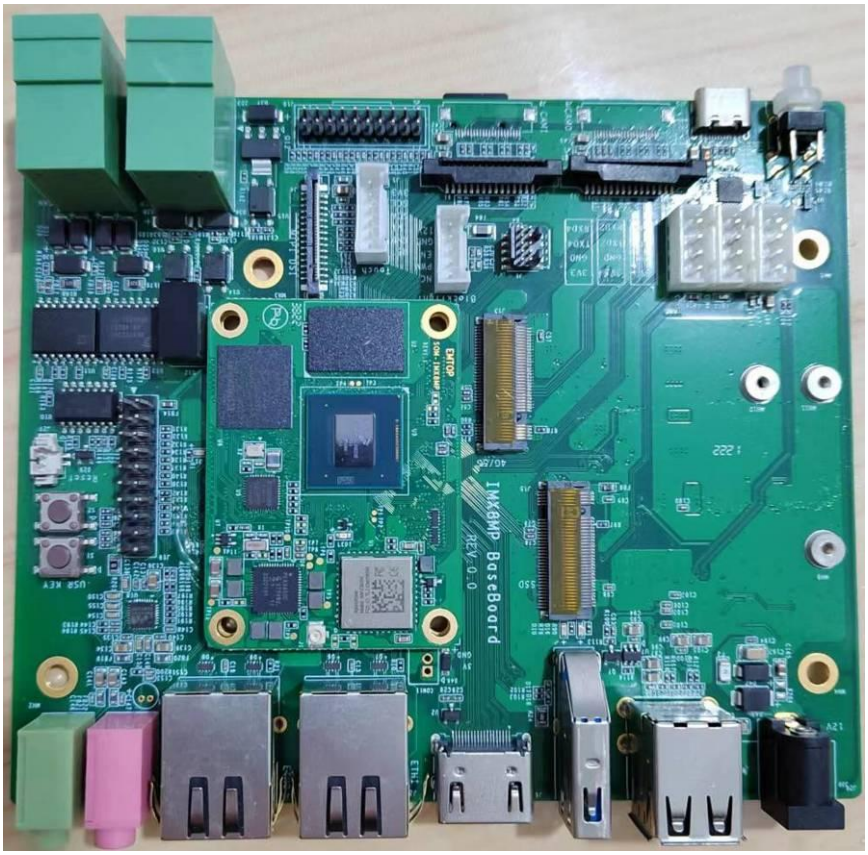
IMX8MP-BASEBOARD



SOM-IMX8MP + IMX8MP-BASEBOARD

User Manual

Version: 0.2

2023-03-17

Revision History：

Version	Date	Description
0.1	2023-02-22	Initial Release
0.2	2023-03-17	Hardware Rev1.0 update

Table of Contents

1. Product Overview

1.1 Introduction

1.2 Resource Download

1.3 Hardware Features

1.4 Mechanical Dimension

2. Linux Operation System

This chapter will give you a general map of the Linux software resources contained in the DVD-ROM provided along with the product, as well as detailed introduction to the process of Linux system development, drivers development, system update, functionality tests and application development examples.

Note：

🕮 It is recommended to learn Ubuntu Linux installation and embedded Linux development technology in advance.

2.1 Software Resources

The DVR-ROM provided along with the board contains demos, application examples, Linux source code and tools, helping you to develop Linux applications and systems easily and quickly.

2.1.1 Location of Resources

You can find software resources such as programs and codes contained in the DVD-ROM according to the information showed in the table below;

Categories	Location
Applications	
Source Code	CD\Source\u-boot-imx-2022.04
	CD\Source\linux-imx-5.15.32
Tools	CD\Tools\
Precompiled Images	CD\Image

2.1.2 BSP

The following table lists types and formats of the files contained in BSP;

Names		Note	Formats
BOOTLOADER	U-BOOT	MMC/SD	Source Code
		FAT	Source Code
		NET	Source Code
KERNEL	LINUX-5.15.32	Support JFFS2/EXT4/FAT/NFS various of file system	Source Code
DEVICE DRIVER	PMIC	PCA9450CHN driver	Source Code
	SERIAL	Serials driver	Source Code
	RTC	Hardware RTC driver	Source Code
	NET	10/100M/1Gbps Ethernet driver	Source Code
	CAN	CAN bus driver	Source Code
	SPI	SPI driver	Source Code
	MIPI-DSI	MIPI-DSI driver	Source Code
	HDMI	HDMI driver	Source Code
	I2C	I2C driver	Source Code
	LVDS	LCD driver	Source Code
	TOUCH SCREEN	I2C and TSC touch panel driver	Source Code
	MMC/SD	MMC/SD controller driver	Source Code
	USB HOST	USB HOST driver	Source Code
	AUDIO	WM8904 Audio driver(supports recording & playback)	Source Code
	BUTTON	GPIO button driver	Source Code
	LED	LED driver	Source Code
	BUZZER	Buzzer driver	Source Code
	CAMERA	CSI Camera driver	Source Code
	PCIe	PCIe interface driver	Source Code
ROOTFS	YOCTO	Wayland with Qt 6.3.1	Image

2.2 Structure of Embedded Linux System

IMX8MP-BASEBOARD is shipped with Linux-5.15.32 system in eMMC by default. This system consists of bootloader, kernel and rootfs. The following table shows the structure of embedded Linux system.

eMMC/SD			
Partition	MBR	FAT	EXT4
Image	Bootloader	DTB, Kernel	Yocto Rootfs

1) Bootloader is a program generated by u-boot compiling; its file name is **flash.bin**.
2) The kernel used in this document is Linux-5.15.32 and has been customized according to the hardware design.
3) Rootfs stores open-source system Yocto with EXT4 format.

2.3 Building Development Environment

Before developing software, user has to establish a Linux cross development environment on PC. This section will take **Ubuntu20.04** operating system as an example to describe how to establish a cross development environment.

It is strongly recommended to install necessary software packages for a newly installed Ubuntu through the following commands.

- **sudo apt-get update; sudo apt-get install -y build-essential git xz-utils ncurses-dev autoconf libtool automake texinfo bison flex libc6:i386 libncurses5:i386 libstdc++6:i386**

Note:

- Each instruction has been put a bullets "•" before it to prevent confusion caused by the long instructions that occupy more than one line in the context.
- Please note the SPACES within each instruction; Missing of any SPACE will cause failure when executing instructions.

2.3.1 Installing Cross Compilation Tools

We provide the cross-compiler under **Tools** directory: **gcc-linaro-7.5.0-2019.12-**

x86_64_aarch64-linux-gnu.tar.xz.

The compiler is mainly used to compile u-boot and kernel.

- **sudo tar -xvf <YOUR_PATH>/gcc-linaro-7.5.0-2019.12-x86_64_aarch64-linux-gnu.tar.xz -C /opt**

It will extract and install under **/opt** directory, keep the default settings.

2.3.2 Set Cross Compile Environment

Run the following commands to set the source code building environment:

- **export PATH=/opt/gcc-linaro-7.5.0-2019.12-x86_64_aarch64-linux-gnu/bin:$PATH**
- **export ARCH=arm64**
- **export CROSS_COMPILE=arm-linux-**

Note：

- The instructions can be added in the .bashrc file located at the user directory, so that the addition of environment variables will be loaded automatically when the system is booting up;
- If you want to check the path, please use the instruction **printenv PATH**

2.4 Preparing the Source Code

Please refer to chapter <**1.2 Resource Download**> to get the development materials, You can get source code under **Source** directory.

- **tar -xvf u-boot-imx-2022.04-git-xxxxxx.tar.xz**
- **tar -xvf linux-imx-5.15.32-git-xxxxxx.tar.xz**

Then we can get the source code directory **u-boot-imx-2022.04** and **linux-imx-5.15.32**.

2.5 Compilation

1） Compiling Bootloader

Run the following commands to compile bootloader:

- **cd u-boot-imx-2022.04**
- **vi make.sh**

```
export PATH=/opt/gcc-linaro-7.5.0-2019.12-x86_64_aarch64-linux-gnu/bin:$PATH
export ARCH=arm64
export CROSS_COMPILE=arm-linux-

DESTDIR="/dev/shm/"
```

PATH: Replace the compiler path according to your local environment if it is installed under other directory.

DESTDIR: point to a directory to store the target image.

Change **DESTDIR** value to make it point to your target directory according to your local environment.

- **./make.sh**

After all the instructions are executed, you can find the booting images named **flash.bin** under **DESTDIR** directory.

2） Compiling Kernel

Execute the following instructions to compile kernel:

- **cd linux-imx-5.15.32**
- **git checkout .**
- **vi make.sh**

```
export PATH=/opt/gcc-linaro-7.5.0-2019.12-x86_64_aarch64-linux-gnu/bin:$PATH
export ARCH=arm64
export CROSS_COMPILE=arm-linux-

DESTDIR="/dev/shm"
```

PATH: Replace the compiler path according to your local environment if it is installed under other directory.

DESTDIR: point to a directory to store the target image.

Please modify **DESTDIR** according to your local environment.

- **make ARCH=arm64 distclean**
- **./make.sh modules**

If it's successfully built, you can find kernel images named **.dtb** files, **Image** and **lib/modules/5.15.32** under **DESTDIR** directory.

Note:

📖 The command **./make.sh**, without parameter, only build dtbs and Image; but **./make.sh modules** will build dtbs, zImage and driver modules.

2.6 Linux System Customization

In order to satisfy different requirements of customers, designers commonly need to make some custom modification based on the default configuration of Linux kernel. This chapter will introduce the process of system customization with some examples.

2.6.1 Replace U-BOOT LOGO

[To be continued]

Note:

📖

2.6.2 Replace Kernel LOGO

- Prepare a picture suitable for your display screen size, named **my_logo.png** for example.
- Install some necessary programs under Ubuntu.
  - **sudo apt-get install netpbm gimp**
- Run command under Ubuntu desktop terminal:
  - **pngtopnm my_logo.png > linuxlogo.pnm**
  - **pnmquant 224 linuxlogo.pnm > linuxlogo224.pnm**
  - **pnmtoplainpnm linuxlogo224.pnm > logo_linux_clut224.ppm**

- Update Linux source code.
  - **cp -f logo_linux_clut224.ppm <YOUR_PATH>/linux-imx-5.15.32/drivers/video/logo/logo_linux_clut224.ppm**
- Re-build the kernel.
  - **make ARCH=arm64 distclean**
  - **./make.sh**

Update the target file **Image** to the board, reboot and check the boot logo on the display screen.

2.6.3 Setting Configuration Menu

A default configuration file is provided under kernel source codes:

linux-imx-5.15.32/arch/arm64/configs/emtop_imx8mp_baseboard_defconfig

Please execute the following commands to enter the configuration menu:

- **cd linux-imx-5.15.32**
- **make ARCH=arm64 emtop_imx8mp_baseboard_defconfig**
- **make ARCH=arm64 menuconfig**

Note：

If an error occurs when command 'make ARCH=arm64 menuconfig' is executed, you might need to install 'ncurse' in the Ubuntu system, ‘ncurses’ is a character graphic library required to generate configuration menu. Please enter the following instruction to install the library:
sudo apt-get install libncurses5-dev

2.6.4 Menu Options

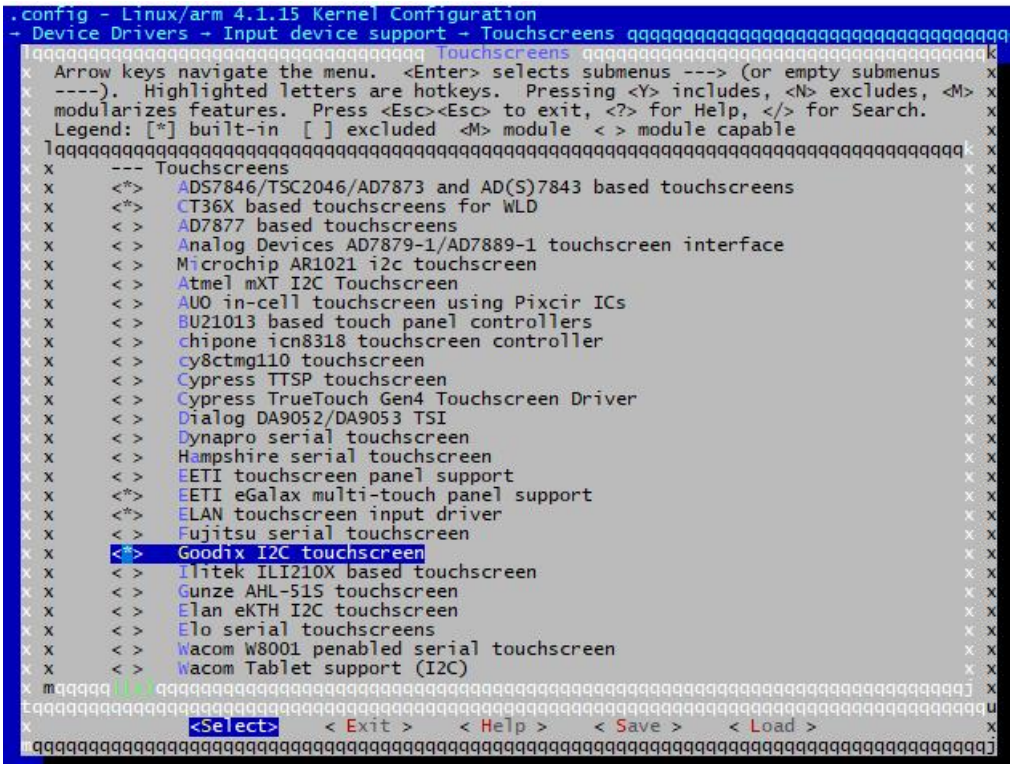
Configure options according to customization requirements after entering configu ration menu, for example, access **Device Drivers** > **Input device support** > **Touchscreens** > **Goodix I2C touchscreen** as shown below:

-> Device Drivers

-> Input device support

-> Touchscreens

-> Goodix I2C touchscreen



Set Goodix I2C touchscreen to **<*>**, exit and save changes.

2.6.5 Compile Kernel

Please execute the following instructions to recompile kernel:

- **./make.sh**

The script will **NOT** overwrite the configuration modified by menuconfig. It mea ns that the current setting you modified is effective in your target kernel image.

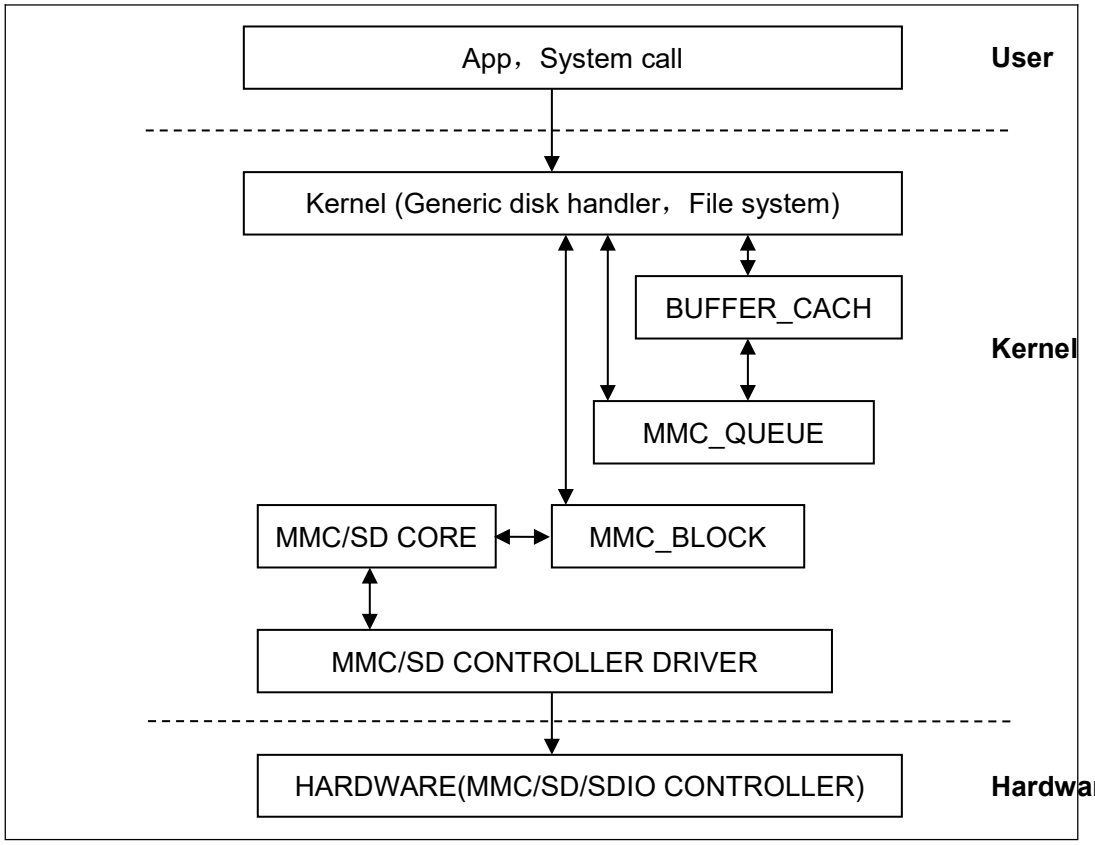
If you want to restore to the default configuration, please delete the file **.config** and run **./make.sh**.

2.7 Introduction to Drivers

The table below shows the access path to find all the drivers:

Category	Name	Description	Location
Bootloader	U-BOOT	MMC/SD	drivers/mmc/fsl_esdhc_imx.c
		FAT	fs/
		NET	drivers/net/fec_mxc.c
Kernel	Linux-5.15.32	Support JFFS2/EXT4/FAT/NFS etc.	fs/
Devices	SERIAL	Serial driver	drivers/tty/serial/imx.c
	RTC	Hardware RTC driver	drivers/rtc/rtc-ds1307.c
	NET	10/100M/1000M Ethernet driver	drivers/net/ethernet/freescale/fec_main.c
	CAN	CAN bus driver	drivers/net/can/flexcan.c
	SPI	SPI driver	drivers/spi/spi-imx.c
	MIPI-DSI	iMX MIPI-DSI driver	drivers/gpu/drm/imx/sec_mipi_dsim-imx.c
	HDMI	HDMI driver	drivers/gpu/drm/imx/dw_hdmi-imx.c
	TOUCH SCREEN	I2C touch panel driver	drivers/input/touchscreen/goodix.c
	MMC/SD	MMC/SD controller driver	drivers/mmc/host/sdhci-esdhc-imx.c
	USB	USB controller driver	drivers/usb/dwc3
	AUDIO	WM8904 Audio driver(supports recording & playback)	sound/soc/codecs/wm8904.c
	BUTTON	GPIO button driver	drivers/input/keyboard/gpio_keys.c
	LED	LED driver	drivers/leds/leds-gpio.c
	BUZZER	Buzzer driver	drivers/leds/leds-gpio.c
	CAMERA	CSI Camera driver	drivers/staging/media/imx/imx8-mipi-csi2-sam.c
	4G/5G	USB GSM modules driver	drivers/usb/serial/option.c
	PCIE	PCIe Interface driver	drivers/phy/freescale/phy-fsl-imx8-pcie.c

2.7.1 SD/MMC



SD/MMC drivers in Linux are mainly consisted of SD/MMC core, mmc_block, mmc_queue and SD/MMC driver：

1） SD/MMC core realizes the codes unrelated to structure in the SD/MMC card operation;

2） mmc_block realizes driver structure when SD/MMC card is used as a block device;

3） mmc_queue realizes management of request queue;

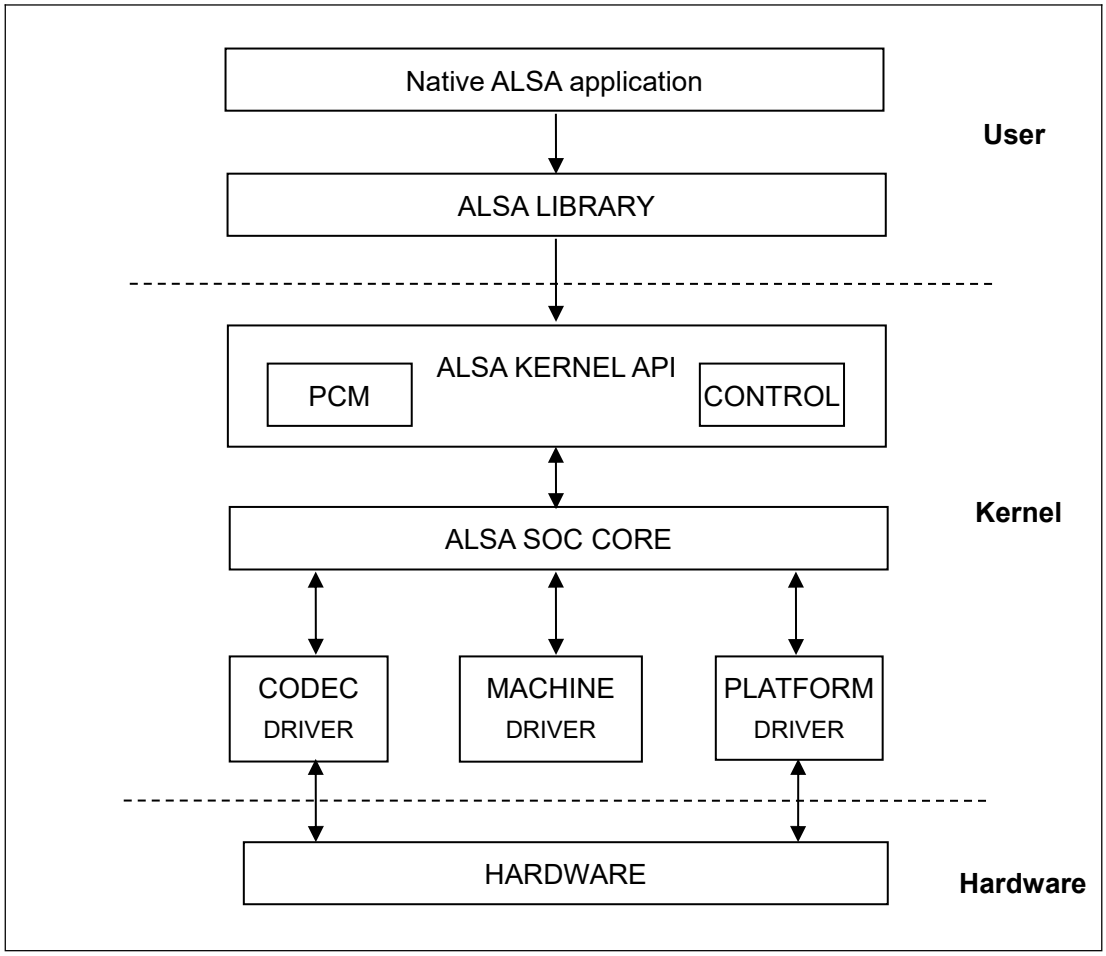
4） SD/MMC driver realizes specific controller driver.

Drivers and relevant documents:

linux-imx-5.15.32/drivers/mmc/

linux-imx-5.15.32/drivers/mmc/host/sdhci-esdhc-imx.c

2.7.2 Audio In/Out



ASoC embedded audio system basically consists of three components:

1) Codec driver: The codec driver is platform independent and contains audio controls, audio interface capabilities, codec dapm definition and codec IO functions.

2) Platform driver: It contains the audio dma engine and audio interface drivers (e.g. I2S, AC97, PCM) of that platform.

3) Machine driver: The machine driver handles any machine specific controls and audio events i.e. turning on an amp at start of playback.

Drivers and relevant documents:

linux-imx-5.15.32/sound/soc/fsl

linux-imx-5.15.32/sound/soc/codecs/wm8904.c

2.8 Driver development

2.8.1 GPIO_LEDs Driver

1）Device Definition

linux-imx-5.15.32/arch/arm64/boot/dts/freescale/emtop-imx8mp-baseboard.dts

Configure GPIO3.16 as system running status indicator, blinking as heartbeat.

```
leds {
    compatible = "gpio-leds";
    pinctrl-names = "default";
    pinctrl-0 = <&pinctrl_gpio_led>;

    sys {
        label = "sys";
        gpios = <&gpio3 16 GPIO_ACTIVE_HIGH>;
        linux,default-trigger = "heartbeat";
    };
```

2）GPIO pinmux Configuration

linux-imx-5.15.32/arch/arm64/boot/dts/freescale/emtop-imx8mp-baseboard.dts

Configure NAND_READY_B as GPIO3_IO16 function:

```
&iomuxc {
    ...
    pinctrl_gpio_led: gpioledgrp {
        fsl,pins = <
            MX8MP_IOMUXC_NAND_READY_B__GPIO3_IO16     0x19
        >;
    };
};
```

3）Driver Design

linux-imx-5.15.32/drivers/leds/leds-gpio.c

a) Call platform_driver_register to register gpio_leds driver

```
static struct platform_driver gpio_led_driver = {
    .probe          = gpio_led_probe,
    .shutdown       = gpio_led_shutdown,
    .driver         = {
        .name       = "leds-gpio",
        .of_match_table = of_gpio_leds_match,
```

```
        },
};

module_platform_driver(gpio_led_driver);

MODULE_AUTHOR("Raphael Assenat <raph@8d.com>, Trent Piepho <tpiepho@freesc
ale.com>");
MODULE_DESCRIPTION("GPIO LED driver");
MODULE_LICENSE("GPL");
MODULE_ALIAS("platform:leds-gpio");
```

b) Apply for gpio and call led_classdev_register to led_classdev drivr.

```
static int gpio_led_probe(struct platform_device *pdev)
{
…
    priv->num_leds = pdata->num_leds;
        for (i = 0; i < priv->num_leds; i++) {
            const struct gpio_led *template = &pdata->leds[i];
            struct gpio_led_data *led_dat = &priv->leds[i];

            if (template->gpiod)
                led_dat->gpiod = template->gpiod;
            else
                led_dat->gpiod =
                    gpio_led_get_gpiod(&pdev->dev,
                                i, template);
            if (IS_ERR(led_dat->gpiod)) {
                dev_info(&pdev->dev, "Skipping unavailable LED gpio %d (%s)\n",
                    template->gpio, template->name);
                continue;
            }

            ret = create_gpio_led(template, led_dat,
                                &pdev->dev, NULL,
                                pdata->gpio_blink_set);
            if (ret < 0)
                return ret;
        }
    } else {
        priv = gpio_leds_create(pdev);
        if (IS_ERR(priv))
            return PTR_ERR(priv);
    }
```

```
	platform_set_drvdata(pdev, priv);

	return 0;
}

static int create_gpio_led(const struct gpio_led *template,
	struct gpio_led_data *led_dat, struct device *parent,
	struct fwnode_handle *fwnode, gpio_blink_set_t blink_set)
{
	struct led_init_data init_data = {};
	int ret, state;

	led_dat->cdev.default_trigger = template->default_trigger;
	led_dat->can_sleep = gpiod_cansleep(led_dat->gpiod);
	if (!led_dat->can_sleep)
		led_dat->cdev.brightness_set = gpio_led_set;
	else
		led_dat->cdev.brightness_set_blocking = gpio_led_set_blocking;
	led_dat->blinking = 0;
	if (blink_set) {
		led_dat->platform_gpio_blink_set = blink_set;
		led_dat->cdev.blink_set = gpio_blink_set;
	}
	if (template->default_state == LEDS_GPIO_DEFSTATE_KEEP) {
		state = gpiod_get_value_cansleep(led_dat->gpiod);
		if (state < 0)
			return state;
	} else {
		state = (template->default_state == LEDS_GPIO_DEFSTATE_ON);
	}
	led_dat->cdev.brightness = state ? LED_FULL : LED_OFF;
	if (!template->retain_state_suspended)
		led_dat->cdev.flags |= LED_CORE_SUSPENDRESUME;
	if (template->panic_indicator)
		led_dat->cdev.flags |= LED_PANIC_INDICATOR;
	if (template->retain_state_shutdown)
		led_dat->cdev.flags |= LED_RETAIN_AT_SHUTDOWN;

	ret = gpiod_direction_output(led_dat->gpiod, state);
	if (ret < 0)
		return ret;
```

```
    if (template->name) {
        led_dat->cdev.name = template->name;
        ret = devm_led_classdev_register(parent, &led_dat->cdev);
    } else {
        init_data.fwnode = fwnode;
        ret = devm_led_classdev_register_ext(parent, &led_dat->cdev,
                                             &init_data);
    }
    return ret;
}
```

c) Users may access the file named brightness under **/sys/class/leds/sys/brightness**, and call gpio_led_set to configure LED status

```
static void gpio_led_set(struct led_classdev *led_cdev,
    enum led_brightness value)
{
...
                gpiod_set_value(led_dat->gpiod, level);
}
```

2.8.2 Pinmux Configuration Guide

Let's take the pad GPIO1_IO01 as an example to explain the pinmux setting steps.

- **vi arch/arm64/boot/dts/freescale/emtop-imx8mp-baseboard.dts**

```
&iomuxc {
    pinctrl_pwm1: pwm1grp {
        fsl,pins = <
            MX8MP_IOMUXC_GPIO1_IO01__PWM1_OUT    0x116
        >;
    };
};
```

The macro MX6UL_PAD_SNVS_TAMPER9__GPIO5_IO09 is defined in **arch/arm64/boot/dts/freescale/imx8mp-pinfunc.h**:

```
#define  MX8MP_IOMUXC_GPIO1_IO01__PWM1_OUT                    0x
018  0x278  0x000  0x1  0x0
```

The value means:

mux_reg	conf_reg	input_reg	mux_mode	input_val

0x018	0x278	0x000	0x1	0x0

Usually we don't need to care about the value it defines, the only thing we need to do is to select the target function from the head file.

```
    #define MX8MP_IOMUXC_GPIO1_IO01__GPIO1_IO01
0x018 0x278 0x000 0x0 0x0
    #define MX8MP_IOMUXC_GPIO1_IO01__PWM1_OUT
  0x018 0x278 0x000 0x1 0x0
    #define MX8MP_IOMUXC_GPIO1_IO01__ISP_SHUTTER_TRIG_0
  0x018 0x278 0x5DC 0x3 0x0
    #define MX8MP_IOMUXC_GPIO1_IO01__ANAMIX_REF_CLK_24M
   0x018 0x278 0x000 0x5 0x0
    #define MX8MP_IOMUXC_GPIO1_IO01__CCM_EXT_CLK2
  0x018 0x278 0x000 0x6 0x0
```

You can refer to the below description in <**IMX8MPRM.pdf**>

IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO01 field descriptions (continued)

Field	Description
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: GPIO1_IO01. 000 **ALT0_GPIO1_IO[1]** — Select mux mode: ALT0 mux port: GPIO1_IO01 of instance: gpio1 001 **ALT1_PWM1_OUT** — Select mux mode: ALT1 mux port: PWM1_OUT of instance: pwm1 011 **ALT3_ISP_SHUTTER_TRIG_0** — Select mux mode: ALT3 mux port: ISP_SHUTTER_TRIG_0 of instance: isp 101 **ALT5_REF_CLK_24M** — Select mux mode: ALT5 mux port: REF_CLK_24M of instance: anamix 110 **ALT6_CCM_EXT_CLK2** — Select mux mode: ALT6 mux port: CCM_EXT_CLK2 of instance: ccm

```
&iomuxc {
        pinctrl_pwm1: pwm1grp {
                fsl,pins = <
                        MX8MP_IOMUXC_GPIO1_IO01__PWM1_OUT        <PADCtrlValue>
                >;
        };
};
```

The **PADCtrlValue** is described in <**IMX8MPRM.pdf**>

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO01 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Select Field Select one out of next values for pad: GPIO1_IO01 0 **PE_0_PULL_DISABLE** — Pull Disable 1 **PE_1_PULL_ENABLE** — Pull Enable
7 HYS	Input Select Field Select one out of next values for pad: GPIO1_IO01 0 **HYS_0_CMOS** — CMOS 1 **HYS_1_SCHMITT** — Schmitt
6 PUE	Pull Up / Down Config. Field Select one out of next values for pad: GPIO1_IO01

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO01 field descriptions (continued)

Field	Description
	0 **PUE_0_WEAK_PULL_DOWN** — Weak pull down 1 **PUE_1_WEAK_PULL_UP** — Weak pull up
5 ODE	Open Drain Field Select one out of next values for pad: GPIO1_IO01 0 **ODE_0_OPEN_DRAIN_DISABLE** — Open Drain Disable 1 **ODE_1_OPEN_DRAIN_ENABLE** — Open Drain Enable
4 FSEL	Slew Rate Field Select one out of next values for pad: GPIO1_IO01 0 **FSEL_0_SLOW_SLEW_RATE** — Slow Slew Rate (SR=1) 1 **FSEL_1_FAST_SLEW_RATE** — Fast Slew Rate (SR=0)
3 -	This field is reserved. Reserved
2–1 DSE	Drive Strength Field Select one out of next values for pad: GPIO1_IO01 00 **DSE_X1** — X1 10 **DSE_X2** — X2 01 **DSE_X4** — X4 11 **DSE_X6** — X6
0 -	This field is reserved. Reserved

Sometimes, the **PADCtrlValue** can be set like 0x800xxxxx or 0x400xxxxx,

0x800xxxxx means no need to set its value, keep it as it was;

0x400xxxxx means to set SION bit, force input path of the pad.

2.9 System Update

SOM-IMX8MP core board can boot up from TF card and eMMC.

Boot Order: eMMC -> TFCard

2.9.1 Update TF Card System Image

1） Make A Bootable TF Card

a) Get the system image from **Image** directory, named as **IMX8MP-BASEBOARD-Yocto-SD-REVXX.img.xz**, unxz it and get the raw image **IMX8MP-BASEBOARD-Yocto-SD-REVXX.img**.

b) If you work under Windows system, please run **Tools/win32diskimager** to write the **IMX8MP-BASEBOARD-Yocto-SD-REVXX.img** into TF Card. If you work under Linux system, please use **dd** command to write it into TF Card.

Image Name	Display Supported
IMX8MP-BASEBOARD-Yocto-SD-REVXX.img	HDMI

Note:

2） Update U-Boot

If you've made some changes to the u-boot source code, and want to update it into TFCard, please run the below command:

- **dd if=<YOUR_PATH>/flash.bin of=/dev/sdx bs=1K seek=32 conv=notrunc**

Note:

🕮 **/dev/sdx** is the TFCard device node recognized under Ubuntu system.

3） Update Kernel

If you have modified the kernel source code, please update the dtb and Image under Partition 1 [FAT32] of the TF Card. That partition can be recognized by Windows or Linux.

4） Update Rootfs

Because EXT4 isn't accessible Under Windows, please mount the Partiton 2 of TF Card under Ubuntu, change the target file and umount the card.

Note:

🕮 If eMMC is already written with system image, please erase eMMC and then reboot the board, because the board will first try to boot from eMMC by default.

🕮 Enter u-boot command and erase eMMC:

u-boot=> **mmc dev 2 && mmc erase 0 20000**

2.9.2 Update eMMC with TFCard

Option 1: Write Complete Image into eMMC

- Make a bootable TFCard and boot up the system;
- Choose the target image [under directory **Image/**] and copy it into the USB disk. If it is **.xz** file, please unxz it to generate **.img** file.
- Install the USB disk on the ARM board, it will be automatically mounted under directory **/run/media/**, for example, the USB disk is recognized as **sda1**;
- Run command to start writing eMMC:
  - root@arm:~# **umount /dev/mmcblk2***
  - root@arm:~# **dd if=/run/media/sda1/IMX8MP-BASEBOARD-Yocto-SD-REVXX.img of=/dev/mmcblk2**

After it's done, power off the board, remove the TFCard, then reboot the board, it should boot from eMMC and enter into Linux prompt.

Option 2: Write Contents in TFCard into eMMC

- Make a bootable TFCard and boot up the system;
- Run command to start writing eMMC:
  - root@arm:~# **system-update.sh**

```
running system update...
==============eMMC UPDATE=================
Warning: disk /dev/mmcblk2 will be formatted !
3000+0 records in
3000+0 records out
1536000 bytes (1.5 MB, 1.5 MiB) copied, 0.189324 s, 8.1 MB/s

Welcome to fdisk (util-linux 2.37.4).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
......
Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

[    82.174125] EXT4-fs (mmcblk2p2): mounted filesystem with ordered data mode. O
pts: (null). Quota mode: none.
sending incremental file list
./
bin/
bin/arping
bin/ash -> /bin/busybox.nosuid
bin/base64 -> /usr/bin/base64.coreutils
bin/bash -> /bin/bash.bash
bin/bash.bash
bin/busybox -> busybox.nosuid
......
sent 13,977,149 bytes  received 141 bytes  2,541,325.45 bytes/sec
total size is 31,423,849  speedup is 2.25
rsync error: some files/attrs were not transferred (see previous errors) (code 23) at
main.c(1336) [sender=3.2.7]
[   825.639924]  mmcblk2: p1 p2
5120+0 records in
5120+0 records out
5242880 bytes (5.2 MB, 5.0 MiB) copied, 0.203386 s, 25.8 MB/s
UPDATE  : COMPLETED
```

```
Catch a signal
[  826.153152] EXT4-fs (mmcblk2p2): mounted filesystem with ordered data mode. O
pts: (null). Quota mode: none.
```

- Power down the board and remove the TF card.

2.10 Test and Demonstration

This section will run some tests on the peripheral devices.

POWER: **12V DC**

Debug Port: **UART2**, **115200 1N8**.

Figure 2-1 Debug Port

2.10.1 SSH LOGIN

The SSH server is already enabled by default. Please get the local IP of the wired-network or wireless-network on ARM board and then login from PC side with SSH client such as PuTTY, **root** account with empty password.

Note:

The SSH server is dropbear, not openssh-server.

2.10.2 RTC

There is a RTC chip RX-8025T on the base board, but the integrated RTC is still enabled by default. So there are 2 RTC devices accessible under system.

- root@arm:~# **cat /sys/class/rtc/rtc0/name**

```
rtc-ds1307 2-0032
```

- root@arm:~# **cat /sys/class/rtc/rtc1/name**

```
snvs_rtc 30370000.snvs:snvs-rtc-lp
```

That means the **rtc0** is rtc-ds1307 [RX-8025T], and **rtc1** is snvs_rtc [Integrated RTC]. The command **hwclock** accesses /dev/rtc0 as default. If you want to access /dev/rtc1, please append parameter: **-f /dev/rtc1**.

Let's set the current time to 2023-02-05 10:12,

- root@arm:~# **date -s "2023-02-05 10:12"; hwclock -w**

Reboot the board, and check the hardware RTC time with below command:

- root@arm:~# **hwclock**

```
2023-02-05 10:13:03.435901+00:00
```

2.10.3 TIMEZONE SETTING

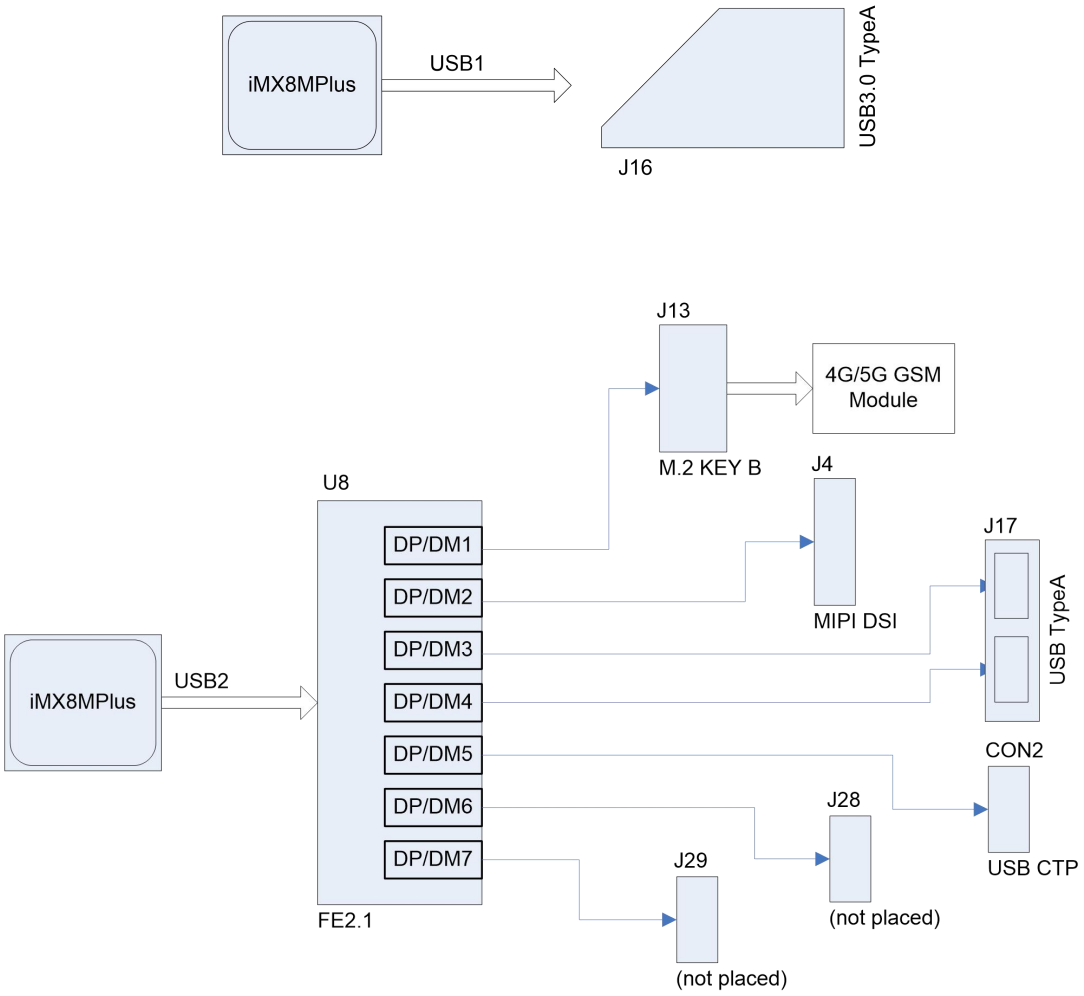
Set Beijing Time for example:

- root@arm:~# **echo "Asia/Shanghai" > /etc/timezone**
- root@arm:~# **ln -sf /usr/share/zoneinfo/Asia/Shanghai /etc/localtime**
- root@arm:~# **sync**

Note:

📖 If NXP Yocto image doesn't contain zoneinfo, copy **/usr/share/zoneinfo** under Ubuntu system to the board, and retry the above commands.

2.10.4 USB HOST



There are 3 USB host channels [USB typeA slot] extended on the base board. Install an USB disk on these slots, check message below:

```
[   272.082860] usb-storage 2-1.1:1.0: USB Mass Storage device detected
[   272.098248] scsi host0: usb-storage 2-1.1:1.0
[   273.104255] scsi 0:0:0:0: Direct-Access     SanDisk  Flash Memory     0.1  PQ: 0
ANSI: 2
[   273.130158] sd 0:0:0:0: [sda] 2001888 512-byte logical blocks: (1.02 GB/977 MiB)
[   273.143825] sd 0:0:0:0: [sda] Write Protect is off
[   273.147410] sd 0:0:0:0: [sda] Mode Sense: 03 00 00 00
[   273.148611] sd 0:0:0:0: [sda] No Caching mode page found
[   273.155755] sd 0:0:0:0: [sda] Assuming drive cache: write through
[   273.176207]  sda: sda1
[   273.199625] sd 0:0:0:0: [sda] Attached SCSI removable disk
[   273.783449] FAT-fs (sda1): Volume was not properly unmounted. Some data may be
corrupt. Please run fsck.
```

- root@arm:~# **mount**

```
.....
/dev/sda1 on /run/media/sda1 type vfat (rw,relatime,gid=6,fmask=0007,dmask=0007,all
ow_utime=0020,codepage=437,iocharset=iso8859-1,shortname=mixed,errors=remount-ro)
```

The USB disk is automatically mounted under **/run/media/sda1** by udev.

Reset USB1

- root@arm:~# **echo 0 > /sys/class/leds/usb1_pwren/brightness; sleep 1; echo 1 > /sys/class/leds/usb1_pwren/brightness**

Reset USB2 HUB

- root@arm:~# **echo 0 > /sys/class/leds/usb2hub_pwren/brightness; sleep 1; echo 1 > /sys/class/leds/usb2hub_pwren/brightness**

2.10.5 NETWORK

There are two 1Gbps network chips AR8035 on board.

HARDWARE	LINUX SYSTEM	INTERFACE	PHY	PHY ADDR
Baseboard J2 [ETH1]	eth0	FEC	BaseBoard AR8035	6
Baseboard J3 [ETH0]	eth1	EQOS	CoreBoard AR8035	4

- root@arm:~# **ifconfig eth0**

```
eth0      Link encap:Ethernet  HWaddr 3a:f7:82:bc:fa:0a
          inet addr:192.168.1.81  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::38f7:82ff:febc:fa0a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:11 errors:0 dropped:4 overruns:0 frame:0
          TX packets:42 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1555 (1.5 KiB)  TX bytes:7192 (7.0 KiB)
```

DHCP feature is enabled as default; the board can request a valid IP address from DHCP server in local network. Also, you can try the below command to force to request IP address:

- root@arm:~# **udhcpc -i eth0**

```
udhcpc: started, v1.35.0
udhcpc: broadcasting discover
udhcpc: broadcasting select for 192.168.1.81, server 192.168.1.1
udhcpc: lease of 192.168.1.81 obtained from 192.168.1.1, lease time 86400
/etc/udhcpc.d/50default: Adding DNS 192.168.1.1
```

Because there are several network interfaces: eth1, ppp, wlan, we need to configure the default gateway:

- root@arm:~# **route del default; route add default eth0**

- root@arm:~# **ping -I eth0 www.baidu.com**

```
PING www.a.shifen.com (14.215.177.38) from 192.168.1.81 eth0: 56(84) bytes of data.
64 bytes from www.baidu.com (183.232.231.174): icmp_seq=1 ttl=56 time=12.1 ms
64 bytes from www.baidu.com (183.232.231.174): icmp_seq=2 ttl=56 time=12.2 ms
64 bytes from www.baidu.com (183.232.231.174): icmp_seq=3 ttl=56 time=12.1 ms
64 bytes from www.baidu.com (183.232.231.174): icmp_seq=4 ttl=56 time=12.5 ms
^C
--- www.a.shifen.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 7.058/7.447/7.771/0.319 ms
```

Do the same operations to eth1.

Perhaps the eth devices order is not guaranteed to be the same every time the board boots up. We have a way to know each of them points to which device:

- root@arm:~# **cat /sys/class/net/eth0/device/uevent**

```
DRIVER=fec
OF_NAME=ethernet
OF_FULLNAME=/soc@0/bus@30800000/ethernet@30be0000
OF_COMPATIBLE_0=fsl,imx8mp-fec
OF_COMPATIBLE_1=fsl,imx8mq-fec
OF_COMPATIBLE_2=fsl,imx6sx-fec
OF_COMPATIBLE_N=3
OF_ALIAS_0=ethernet0
MODALIAS=of:NethernetT(null)Cfsl,imx8mp-fecCfsl,imx8mq-fecCfsl,imx6sx-fec
```

- root@arm:~# **cat /sys/class/net/eth1/device/uevent**

```
DRIVER=imx-dwmac
OF_NAME=ethernet
OF_FULLNAME=/soc@0/bus@30800000/ethernet@30bf0000
OF_COMPATIBLE_0=nxp,imx8mp-dwmac-eqos
OF_COMPATIBLE_1=snps,dwmac-5.10a
OF_COMPATIBLE_N=2
OF_ALIAS_0=ethernet1
MODALIAS=of:NethernetT(null)Cnxp,imx8mp-dwmac-eqosCsnps,dwmac-5.10a
```

2.10.6 HDMI

MODEL	DTB
HDMI Displayer	emtop-imx8mp-baseboard-hdmi.dtb

Edit **uEnv.txt**: let **fdtfile** point to the DTB in the above table.

```
fdtfile=emtop-imx8mp-baseboard-hdmi.dtb
```

Connect HDMI displayer, power up the ARM board. It can display Linux boot logo and Wayland desktop.

2.10.7 MIPI-DSI

Devices already tested:

MODEL	DESCRIPTION	DTB
MIPI-70T	1024 * 600, with touch panel GT911	emtop-imx8mp-baseboard-mipi-dsi.dtb

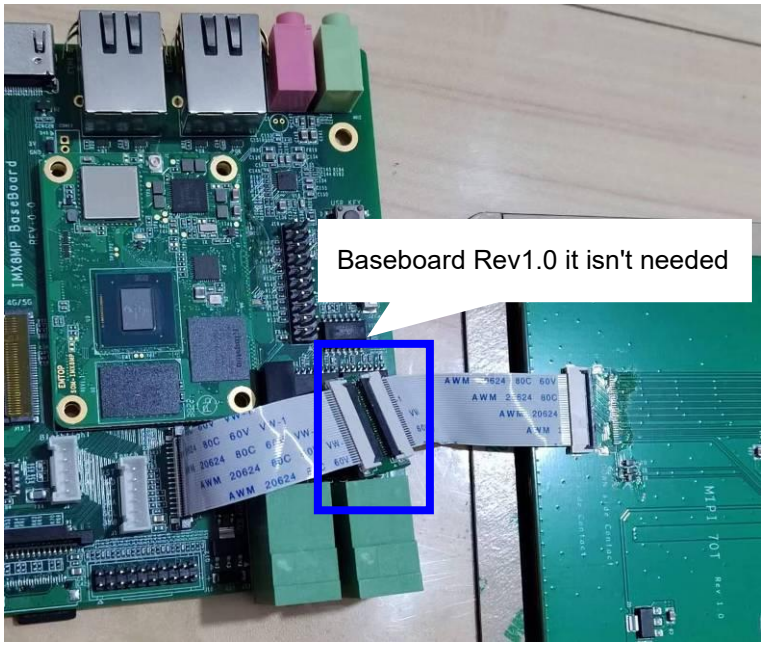


Figure 2-2 MIPI-70T Connection

Edit **uEnv.txt**: let **fdtfile** point to the DTB in the above table.

2.10.8 MIPI-DSI BACKLIGHT

- root@arm:~# **echo 50 > /sys/class/backlight/dsi_backlight/brightness**

Note:

The value of backlight level should be: **0** ~ **255**.

2.10.9 LVDS

Devices already tested:

MODEL	DESCRIPTION	DTB
BA104S01-100	800 * 600	emtop-imx8mp-baseboard-lvds.dtb

Edit **uEnv.txt**: let **fdtfile** point to the DTB in the above table.

2.10.10 LVDS BACKLIGHT

- root@arm:~# **echo 50 > /sys/class/backlight/lvds_backlight/brightness**

Note:

🕮 The value of backlight level should be: **0** ~ **255**.

2.10.11 TOUCH PANEL

MODEL	TYPE	I2C BUS
GT911	I2C CTP	I2C2

- root@arm:~# **evtest**

```
No device specified, trying to scan all of /dev/input/event*
Available devices:
/dev/input/event0:          30370000.snvs:snvs-powerkey
/dev/input/event1:          Goodix Capacitive TouchScreen
/dev/input/event2:          gpio-keys
Select the device event number [0-2]: 1
Input driver version is 1.0.1
Input device ID: bus 0x18 vendor 0x416 product 0x38f version 0x1060
Input device name: "Goodix Capacitive TouchScreen"
Supported events:
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 59 (KEY_F1)
    Event code 60 (KEY_F2)
    Event code 61 (KEY_F3)
    Event code 62 (KEY_F4)
    Event code 63 (KEY_F5)
    Event code 64 (KEY_F6)
    Event code 125 (KEY_LEFTMETA)
    Event code 330 (BTN_TOUCH)
  Event type 3 (EV_ABS)
    Event code 0 (ABS_X)
      Value      0
      Min        0
      Max     1023
    Event code 1 (ABS_Y)
      Value      0
      Min        0
      Max      599
    Event code 47 (ABS_MT_SLOT)
      Value      0
      Min        0
```

```
      Max          4
    Event code 48 (ABS_MT_TOUCH_MAJOR)
      Value        0
      Min          0
      Max        255
    Event code 50 (ABS_MT_WIDTH_MAJOR)
      Value        0
      Min          0
      Max        255
    Event code 53 (ABS_MT_POSITION_X)
      Value        0
      Min          0
      Max       1023
    Event code 54 (ABS_MT_POSITION_Y)
      Value        0
      Min          0
      Max        599
    Event code 57 (ABS_MT_TRACKING_ID)
      Value        0
      Min          0
      Max      65535
Properties:
  Property type 1 (INPUT_PROP_DIRECT)
Testing ... (interrupt to exit)
[Touch the panel ...]
Event: time 1647024852.722824, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID), value 0
Event: time 1647024852.722824, type 3 (EV_ABS), code 53 (ABS_MT_POSITION_X), value 878
Event: time 1647024852.722824, type 3 (EV_ABS), code 54 (ABS_MT_POSITION_Y), value 255
Event: time 1647024852.722824, type 3 (EV_ABS), code 48 (ABS_MT_TOUCH_MAJOR), value 10
Event: time 1647024852.722824, type 3 (EV_ABS), code 50 (ABS_MT_WIDTH_MAJOR), value 10
Event: time 1647024852.722824, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 1
Event: time 1647024852.722824, type 3 (EV_ABS), code 0 (ABS_X), value 878
Event: time 1647024852.722824, type 3 (EV_ABS), code 1 (ABS_Y), value 255
Event: time 1647024852.722824, ---------- SYN_REPORT ------------
Event: time 1647024852.756503, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID), value -1
Event: time 1647024852.756503, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 0
Event: time 1647024852.756503, ---------- SYN_REPORT ------------
```

2.10.12 WM8904 AUDIO

- root@arm:~# **aplay -l**

```
**** List of PLAYBACK Hardware Devices ****
card 0: imx8mpwm8904 [imx8mp-wm8904], device 0: 30c30000.sai-wm8904-hifi
wm8904-hifi-0 [30c30000.sai-wm8904-hifi wm8904-hifi-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

Playback:

- root@arm:~# **aplay /usr/share/sounds/alsa/*.wav**

Record:

- root@arm:~# **arecord -r 44100 -f S16_LE -c 2 -d 10 record.wav**

Wait several seconds, press Ctrl+C to terminate arecord program. Now, let's play it to check:

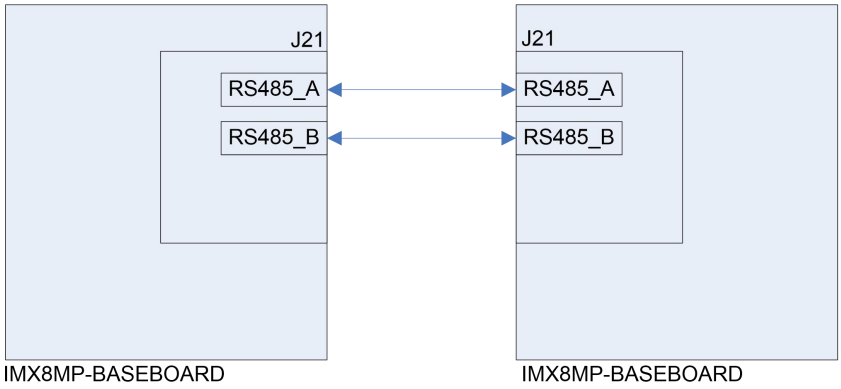
- root@arm:~# **aplay record.wav**

2.10.13 UART

DEVICE NODE	HARDWARE	USAGE
/dev/ttymxc0	UART1	BLUETOOTH
/dev/ttymxc1	UART2	DEBUG PORT
/dev/ttymxc2	UART3	RS485

2.10.14 RS485

Connect a RS485 device, or connect 2 boards directly:



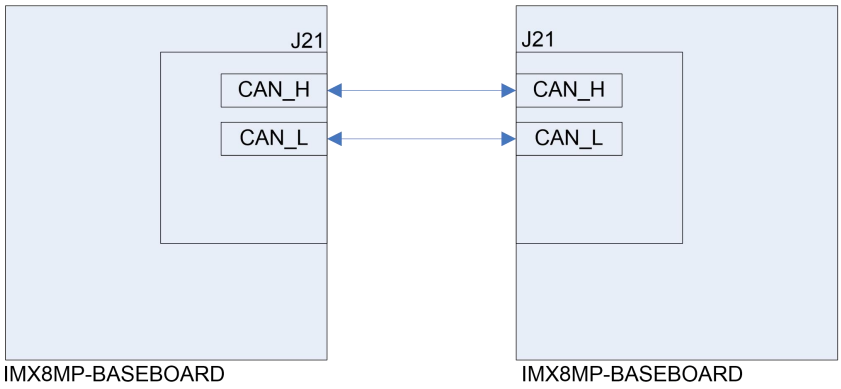
Run below command on both of the boards:

- root@arm:~# **/test/app/com -d /dev/ttymxc2 -m rs485**

```
SEND: 1234567890
RECV: 1234567890
SEND: 1234567890
RECV: 1234567890
```

2.10.15 CAN BUS

Connect 2 boards directly:



- root@arm:~# **ifconfig can0**

```
can0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          NOARP  MTU:16  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:10
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Configure parameters [both side]:

- root@arm:~# **ifconfig can0 down**
- root@arm:~# **ip link set can0 type can bitrate 125000**
- root@arm:~# **ip link set can0 type can restart-ms 100**
- root@arm:~# **ifconfig can0 up**

Start to listen on one board:

- root@arm:~# **candump can0 &**

Send package on the other board:

- root@arm:~# **cansend can0 "5A1#1122334455667788"**

For more information, please refer to project can-utils.

2.10.16 BUTTON

ON/OFF Button:

- root@arm:~# **evtest /dev/input/event0**

```
Input driver version is 1.0.1
Input device ID: bus 0x19 vendor 0x0 product 0x0 version 0x0
Input device name: "30370000.snvs:snvs-powerkey"
Supported events:
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 116 (KEY_POWER)
Properties:
Testing ... (interrupt to exit)
Event: time 1675609543.617915, type 1 (EV_KEY), code 116 (KEY_POWER), value 1
Event: time 1675609543.617915, ---------- SYN_REPORT ------------
Event: time 1675609545.154207, type 1 (EV_KEY), code 116 (KEY_POWER), value 0
Event: time 1675609545.154207, ---------- SYN_REPORT ------------
```

User Button [**S1**]:

- root@arm:~# **evtest /dev/input/event1**

```
Input driver version is 1.0.1
Input device ID: bus 0x19 vendor 0x1 product 0x1 version 0x100
Input device name: "gpio-keys"
Supported events:
```

```
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 256 (BTN_0)
Properties:
Testing ... (interrupt to exit)
Event: time 1675609912.982177, type 1 (EV_KEY), code 256 (BTN_0), value 1
Event: time 1675609912.982177, ---------- SYN_REPORT ------------
Event: time 1675609913.107597, type 1 (EV_KEY), code 256 (BTN_0), value 0
Event: time 1675609913.107597, ---------- SYN_REPORT ------------
```

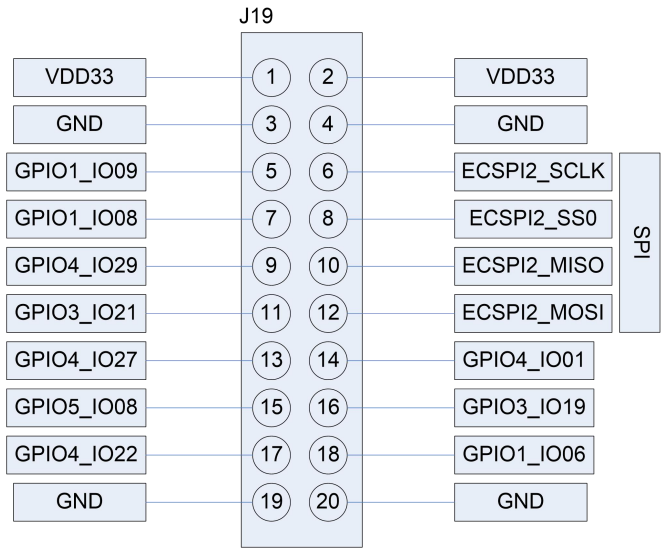
2.10.17 LED

There are two LED respectively on core board and the base board, they are controlled by GPIO3_IO16. Let's test it:

- root@arm:~# **echo none > /sys/class/leds/sys/trigger**
- root@arm:~# **while test 1; do echo 1 > /sys/class/leds/sys/brightness;sleep 1;echo 0 > /sys/class/leds/sys/brightness;sleep 1;done**

Your can see the corresponding LED blinking with 2Hz frequency.

2.10.18 GPIO



GPIO	GPIOCHIP NUM	OFFSET
GPIO1_IO09	0	9

GPIO1_IO08	0	8
GPIO4_IO29	3	29
GPIO3_IO21	2	21
GPIO4_IO27	3	27
GPIO5_IO08	4	8
GPIO4_IO22	3	22
GPIO4_IO01	3	1
GPIO3_IO19	2	19
GPIO1_IO06	0	6

- root@arm:~# **gpiodetect**

```
gpiochip0 [30200000.gpio] (32 lines)
gpiochip1 [30210000.gpio] (32 lines)
gpiochip2 [30220000.gpio] (32 lines)
gpiochip3 [30230000.gpio] (32 lines)
gpiochip4 [30240000.gpio] (32 lines)
```

- root@arm:~# **gpioinfo 0**

```
gpiochip0 - 32 lines:
        line   0:      unnamed         unused  input  active-high
        line   1:      unnamed         unused  input  active-high
        line   2:      unnamed         unused  input  active-high
        line   3:      unnamed  "interrupt"    input  active-high [used]
        line   4:      unnamed         unused  input  active-high
        line   5:      unnamed         unused  input  active-high
        line   6:      unnamed         unused  input  active-high
        line   7:      unnamed "usb1_pwren"    output  active-high [used]
        line   8:      unnamed         unused  input  active-high
        line   9:      unnamed         unused  input  active-high
        line  10:      unnamed         unused  input  active-high
        line  11:      unnamed  "PHY reset"    output  active-low [used]
        line  12:      unnamed         unused  input  active-high
        line  13:      unnamed        "reset"  output  active-low [used]
        line  14:      unnamed         unused  input  active-high
        line  15:      unnamed "usb2hub_pwren" output active-high [used]
        line  16:      unnamed         unused  input  active-high
        line  17:      unnamed         unused  input  active-high
        line  18:      unnamed         unused  input  active-high
        line  19:      unnamed         unused  input  active-high
        line  20:      unnamed         unused  input  active-high
        line  21:      unnamed         unused  input  active-high
```

```
line  22:      unnamed         unused   input   active-high
line  23:      unnamed         unused   input   active-high
line  24:      unnamed         unused   input   active-high
line  25:      unnamed         unused   input   active-high
line  26:      unnamed         unused   input   active-high
line  27:      unnamed         unused   input   active-high
line  28:      unnamed         unused   input   active-high
line  29:      unnamed         unused   input   active-high
line  30:      unnamed         unused   input   active-high
line  31:      unnamed         unused   input   active-high
```

Let's set GPIO1_IO09 output high:

- root@arm:~# **gpioset 0 9=1**

Set GPIO1_IO09 output low:

- root@arm:~# **gpioset 0 9=0**

Read GPIO1_IO09 input value:

- root@arm:~# **while test 1; do gpioget 0 9; sleep 1; done**

```
0
0
1   [Provide 3.3V to the corresponding pin]
1
1
```

Monitor the pin state:

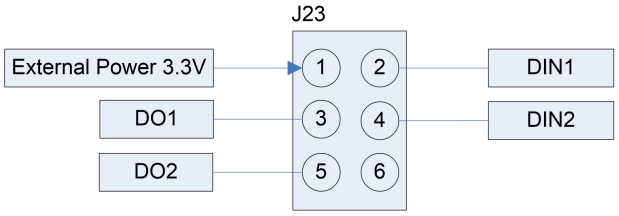
- root@arm:~# **gpiomon 0 9**

```
event:    RISING EDGE offset: 9 timestamp: [     12573.516365625]
event: FALLING EDGE offset: 9 timestamp: [     12573.521460375]
event:    RISING EDGE offset: 9 timestamp: [     12573.620453625]
event: FALLING EDGE offset: 9 timestamp: [     12575.427290500]
```

Note:

📖 The extension **libgpiod** is already installed in the release image, current version is 1.6.3.

2.10.19 DI/DO



SIGNAL	GPIO	GPIOCHIP NUM	OFFSET
DO1	GPIO3_IO24	2	24
DO2	GPIO3_IO25	2	25
DIN1	GPIO4_IO19	3	19
DIN2	GPIO4_IO18	3	18

Provide external power 3.3V to pin1 @ **J23**.

Test DO1 output:

- root@arm:~# **while test 1; do gpioset 2 24=0; sleep 1; gpioset 2 24=1; sleep 1; done**

Test DO2 output:

- root@arm:~# **while test 1; do gpioset 2 25=0; sleep 1; gpioset 2 25=1; sleep 1; done**

Connect DO1 with DIN1, DO2 with DIN2 in **J23**.

Let DO1 output a signal sequence and read value from DIN1:

- root@arm:~# **killall gpiomon; gpiomon 3 19 & while test 1; do gpioset 2 24=0; sleep 1; gpioset 2 24=1; sleep 1; done**

```
event: FALLING EDGE offset: 19 timestamp: [      1326.690525750]
event:   RISING EDGE offset: 19 timestamp: [      1327.696323000]
event: FALLING EDGE offset: 19 timestamp: [      1328.703165250]
event:   RISING EDGE offset: 19 timestamp: [      1329.709568625]
```

Let DO2 output a signal sequence and read value from DIN2:

- root@arm:~# **killall gpiomon; gpiomon 3 18 & while test 1; do gpioset 2 25=0; sleep 1; gpioset 2 25=1; sleep 1; done**

```
event: FALLING EDGE offset: 18 timestamp: [      1528.980944750]
event:    RISING EDGE offset: 18 timestamp: [      1529.987558750]
event: FALLING EDGE offset: 18 timestamp: [      1530.994076500]
event:    RISING EDGE offset: 18 timestamp: [      1532.001185375]
```

Stop **gpiomon** process at the end:

- root@arm:~# **killall gpiomon**

2.10.20 M.2/KEY-M PCIe

Devices already tested:

MODEL	TYPE
PM991 NVMe	SSD

- root@arm:~# **fdisk -l**

```
Disk /dev/nvme0n1: 119.24 GiB, 128035676160 bytes, 250069680 sectors
Disk model: SAMSUNG MZALQ128HBHQ-000L1
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 77102AE6-5D1D-4C14-84EE-452828F87C2C
```

If the SSD is not partitioned, you can partition it with command **fdisk**. If the SSD is already partitioned, it will be automatically mounted by system, you should umount it before formatting operation:

- root@arm:~# **umount /dev/nvme0n1***

- root@arm:~# **fdisk /dev/nvme0n1**

```
This disk is currently in use - repartitioning is probably a bad idea.
It's recommended to umount all file systems, and swapoff all swap
partitions on this disk.

Command (m for help): n
Partition number (1-128, default 1):
First sector (34-250069646, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-250069646, default 250069646):
```

```
Created a new partition 1 of type 'Linux filesystem' and of size 119.2 GiB.
Partition #1 contains a ext4 signature.

Do you want to remove the signature? [Y]es/[N]o: Y

The signature will be removed by a write command.

Command (m for help): p
Disk /dev/nvme0n1: 119.24 GiB, 128035676160 bytes, 250069680 sectors
Disk model: SAMSUNG MZALQ128HBHQ-000L1
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 77102AE6-5D1D-4C14-84EE-452828F87C2C

Device             Start          End      Sectors     Size Type
/dev/nvme0n1p1    2048 250069646 250067599 119.2G Linux filesystem

Filesystem/RAID signature on partition 1 will be wiped.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
[ 1509.690418]    nvme0n1: p1
Syncing disks.
[ 1509.696390]    nvme0n1: p1
```

- root@arm:~# **mkfs.ext4 /dev/nvme0n1p1**

```
mke2fs 1.46.5 (30-Dec-2021)
Discarding device blocks: done
Creating filesystem with 31258449 4k blocks and 7815168 inodes
Filesystem UUID: 951adf23-f3f2-4a1d-8fb7-45bf78603fb1
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424, 20480000, 23887872

Allocating group tables: done
Writing inode tables: done
Creating journal (131072 blocks): done
Writing superblocks and filesystem accounting information: done
```

Now, try to mount it, read and write to it like USB disk.

2.10.21 M.2/KEY-B USB

Please refer to chapter **4G & 5G**.

2.10.22 TFCard

When booting from eMMC, the TFCard will be recognized as a removable disk device.

2.10.23 eMMC

eMMC is mainly used for keeping system image, needless to test it manually.

2.10.24 QSPIFLASH

- root@arm:~# **cat /sys/class/mtd/mtd0/device/spi-nor/partname**

```
w25q64dw
```

- root@arm:~# **cat /proc/mtd**

```
dev:    size   erasesize  name
mtd0: 00800000 00010000 "30bb0000.spi"
```

Erase and format:

- root@arm:~# **flash_erase /dev/mtd0 0 0**

```
Erasing 8192 Kibyte @ 0 -- 100 % complete
```

- root@arm:~# **mount -t jffs2 /dev/mtdblock0 /mnt**

Write and read under directory **/mnt**, the content will keep in the QSPIFlash memory.

- root@arm:~# **umount /mnt**

Next boot, mount the flash and you can see the contents written before.

2.10.25 UNIQUE ID

- root@arm:~# **cat /sys/devices/soc0/serial_number**

```
17070800239290FE
```

2.10.26 AW-CM358SM WIFI

- root@arm:~# **modprobe mlan**
- root@arm:~# **modprobe moal sta_name=wlan uap_name=wlan wfd_name=p2p max_vir_bss=1 cfg80211_wext=0xf cal_data_cfg=none fw_name=sdiouart8987_combo_v0.bin**

```
[  504.103927] wlan: Loading MWLAN driver
[  504.108540] wlan: Register to Bus Driver...
[  504.113955] vendor=0x02DF device=0x9149 class=0 function=1
[  504.119792] Attach moal handle ops, card interface type: 0x105
[  504.125675] No module param cfg file specified
[  504.130141] SDIO: max_segs=128 max_seg_size=65535
[  504.134862] rx_work=1 cpu_num=4
[  504.138035] Attach mlan adapter operations.card_type is 0x105.
[  504.144184] wlan: Enable TX SG mode
[  504.147698] wlan: Enable RX SG mode
[  504.157132] Request firmware: sdiouart8987_combo_v0.bin
[  504.163437] WLAN FW already running! Skip FW download
[  504.168584] WLAN FW is active
[  504.171580] on_time is 504170812875
[  504.198969] fw_cap_info=0x181d7f03, dev_cap_mask=0xffffffff
[  504.204573] max_p2p_conn = 8, max_sta_conn = 8
[  504.209315] SDIO rx aggr: 1 block_size=512
[  504.213472] wlan: Enable RX SG mode
[  504.216983] mpa_rx_buf_size=65280
[  504.245620] wlan: version = SD8987----16.92.21.p41.4-MM5X16322.p3-(FP92)
[  504.257468] wlan: Register to Bus Driver Done
[  504.261946] wlan: Driver loaded successfully
```

- root@arm:~# **ifconfig wlan0 up**
- root@arm:~# **iwlist wlan0 scan**

```
.....
          Cell 03 - Address: DC:73:85:76:53:6C
                    ESSID:"EMTOP" [3]
                    Mode:Master
                    Frequency=2.462 GHz (Channel 11)
                    Quality:0/5    Signal level:-93 dBm    Noise level:-96 dBm
                    Encryption key:on
                    Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 6 Mb/s; 9 Mb/s
                              11 Mb/s; 12 Mb/s; 18 Mb/s; 24 Mb/s; 36 Mb/s
                              48 Mb/s; 54 Mb/s
                    Extra:Beacon interval=100
                    IE: IEEE 802.11i/WPA2 Version 1
                        Group Cipher : CCMP
                        Pairwise Ciphers (1) : CCMP
                        Authentication Suites (1) : PSK
                    IE:                                                    Unknown:
DD180050F2020101800003A4000027A4000042435E0062322F00
                    IE: Unknown: DD08AC853D8201000000
                    IE:                                                    Unknown:
DD230050F204104A000110104400010210080002078010 3C0001011049000600372A000120
                    IE: Unknown: DD0F00E0FC400000000100FD04A80000A8
                    Extra:band=bg
```

- root@arm:~# **wpa_passphrase EMTOP 12345678 >> /etc/wpa_supplicant.conf**

File: /etc/wpa_supplicant.conf

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1

network={
        key_mgmt=NONE
}
network={
        ssid="EMTOP"
        #psk="12345678"

psk=c238e09ef54285daf31c8f6833efab9fb8ff55632f7b9a7d94c117711de27822
}
```

- root@arm:~# **wpa_supplicant -B -iwlan0 -c/etc/wpa_supplicant.conf**

```
Successfully initialized wpa_supplicant
```

```
rfkill: Cannot open RFKILL control device
[ 2594.006812] wlan: wlan0 START SCAN
[ 2598.357520] wlan: SCAN COMPLETED: scanned AP count=0
[ 2603.369086] wlan: wlan0 START SCAN
[ 2607.717417] wlan: SCAN COMPLETED: scanned AP count=1
[ 2607.735508] wlan: Connected to bssid 94:XX:XX:XX:0a:bc successfully
[ 2608.381534] wlan0:
[ 2608.381550] wlan: Send EAPOL pkt to 94:XX:XX:XX:0a:bc
[ 2608.398971] wlan0:
[ 2608.398985] wlan: Send EAPOL pkt to 94:XX:XX:XX:0a:bc
[ 2608.400137] woal_cfg80211_set_rekey_data return: gtk_rekey_offload is DISABLE
```

- root@arm:~# **udhcpc -i wlan0**

```
udhcpc: started, v1.35.0
udhcpc: broadcasting discover
udhcpc: broadcasting discover
udhcpc: broadcasting select for 192.168.1.100, server 192.168.1.1
udhcpc: lease of 192.168.1.100 obtained from 192.168.1.1, lease time 7200
RTNETLINK answers: File exists
/etc/udhcpc.d/50default: Adding DNS 192.168.1.1
```

Note:

When the kernel configuration is modified and rebuilt, perhaps the WiFi driver should be rebuilt if the **modprobe** command reports error and fails.

2.10.27 AW-CM358SM BLUETOOTH

- root@arm:~# **hciattach /dev/ttymxc0 any 115200 flow**

```
Setting TTY to N_HCI line discipline
Device setup complete
[   146.160466] NET: Registered PF_ALG protocol family
```

- root@arm:~# **bluetoothctl**

```
Agent registered
[bluetooth]# power on
Changing power on succeeded
[bluetooth]# scan on
Discovery started
[CHG] Controller D0:C5:D3:F9:60:06 Discovering: yes
[NEW] Device 78:C5:28:67:88:03 78-C5-28-67-88-03
```

```
[NEW] Device 7B:A2:1E:1D:15:60 7B-A2-1E-1D-15-60
.. ..
[bluetooth]# scan off
```

Please search **bluetoothctl** usage on web for more information.

Note:

📖 **mlan.ko** and **moal.ko** must be loaded before **hciattach** operation, otherwise it will report error: **Bluetooth: hci0: Frame reassembly failed (-84)**.

2.10.28 4G & 5G

Devices already tested:

MODEL	4G/5G
QUECTEL EM05-CE	4G
QUECTEL RM500Q-GL	5G

Figure 2-3 4G Module **EM05-CE**

Figure 2-4 5G Module **RM500Q-GL**

Install QUECTEL GSM module, SIM card and antenna.

Enable power supply [**4G**]:

- root@arm:~# **echo 0 > /sys/class/leds/gsm_pwrsel/brightness**
- root@arm:~# **echo 1 > /sys/class/leds/gsm_pwren/brightness**

Or enable power supply [**5G**]:

- root@arm:~# **echo 1 > /sys/class/leds/gsm_pwrsel/brightness**
- root@arm:~# **echo 1 > /sys/class/leds/gsm_pwren/brightness**

Wait about 10 seconds.

```
[   696.459095] option 3-1.1:1.0: GSM modem (1-port) converter detected
[   696.465847] usb 3-1.1: GSM modem (1-port) converter now attached to ttyUSB0
[   696.473511] option 3-1.1:1.1: GSM modem (1-port) converter detected
[   696.480292] usb 3-1.1: GSM modem (1-port) converter now attached to ttyUSB1
[   696.487876] option 3-1.1:1.2: GSM modem (1-port) converter detected
[   696.494574] usb 3-1.1: GSM modem (1-port) converter now attached to ttyUSB2
[   696.502194] option 3-1.1:1.3: GSM modem (1-port) converter detected
[   696.508949] usb 3-1.1: GSM modem (1-port) converter now attached to ttyUSB3
```

Terminate **pppd** program which may be running background:

- root@arm:~# **killall -q pppd && sleep 3**

- root@arm:~# **pppd call quectel-ppp &**

```
... ...
Script /usr/local/sbin/chat -E -s -v -f /etc/ppp/peers/quectel-chat-connect finished (pid 891),
status = 0x0
Serial connection established.
using channel 6
Using interface ppp0
Connect: ppp0 <--> /dev/ttyGSM03
sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x99ca38bd> <pcomp> <accomp>]
rcvd [LCP ConfReq id=0xa <asyncmap 0x0> <auth chap MD5> <magic 0x8fb21dd6>
<pcomp> <accomp>]
sent [LCP ConfAck id=0xa <asyncmap 0x0> <auth chap MD5> <magic 0x8fb21dd6>
<pcomp> <accomp>]
rcvd [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x99ca38bd> <pcomp> <accomp>]
sent [LCP EchoReq id=0x0 magic=0x99ca38bd]
rcvd [LCP DiscReq id=0xb magic=0x8fb21dd6]
rcvd [CHAP Challenge id=0x1 <ede1a1633678b8a18ed16d5f1891b8cf>, name =
"UMTS_CHAP_SRVR"]
sent [CHAP Response id=0x1 <68c3d55a12080e299e8b3751431746cf>, name =
"$LTE_USERNAME"]
rcvd [LCP EchoRep id=0x0 magic=0x8fb21dd6 99 ca 38 bd]
rcvd [CHAP Success id=0x1 ""]
CHAP authentication succeeded
CHAP authentication succeeded
sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
sent [IPV6CP ConfReq id=0x1 <addr fe80::a062:33a3:7882:408f>]
rcvd [IPCP ConfReq id=0x8]
sent [IPCP ConfNak id=0x8 <addr 0.0.0.0>]
rcvd [IPCP ConfNak id=0x1 <addr 10.33.200.184> <ms-dns1 202.96.128.86> <ms-dns2
202.96.134.133>]
sent [IPCP ConfReq id=0x2 <addr 10.33.200.184> <ms-dns1 202.96.128.86> <ms-dns2
202.96.134.133>]
rcvd [IPCP ConfReq id=0x9]
sent [IPCP ConfAck id=0x9]
rcvd [IPCP ConfAck id=0x2 <addr 10.33.200.184> <ms-dns1 202.96.128.86> <ms-dns2
202.96.134.133>]
Could not determine remote IP address: defaulting to 10.64.64.64
local  IP address 10.33.200.184
remote IP address 10.64.64.64
primary     DNS address 202.96.128.86
secondary DNS address 202.96.134.133
Script /etc/ppp/ip-up started (pid 900)
```

```
Script /etc/ppp/ip-up finished (pid 900), status = 0x0
```

Note:

🕮 If **pppd** command reports error, please try to run it again.

Configure default gateway:

- root@arm:~# **route del default; route add default ppp0**

Configure **resolv.conf**:

- root@arm:~# **cat /etc/ppp/resolv.conf > /etc/resolv.conf**

Note:

🕮 The **resolv.conf** is very important. If it's not correct, the ping command with URL will report error like this: Temporary failure in name resolution.

Connection test:

- root@arm:~# **ping -I ppp0 www.baidu.com**

```
PING www.a.shifen.com (14.215.177.38) from 10.32.232.200 ppp0: 56(84) bytes of data.
64 bytes from 14.215.177.38: icmp_seq=1 ttl=54 time=37.0 ms
64 bytes from 14.215.177.38: icmp_seq=2 ttl=54 time=43.5 ms
64 bytes from 14.215.177.38: icmp_seq=3 ttl=54 time=51.8 ms
64 bytes from 14.215.177.38: icmp_seq=4 ttl=54 time=41.4 ms
^C64 bytes from 14.215.177.38: icmp_seq=5 ttl=54 time=33.4 ms

--- www.a.shifen.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 20329ms
rtt min/avg/max/mdev = 33.408/41.456/51.856/6.272 ms
```

GSM Disable

It's usually called **'airplane mode'**, disable wireless transmission.

- root@arm:~# **echo 0 > /sys/class/leds/gsm_pwren/brightness**

GSM Enable

- root@arm:~# **echo 1 > /sys/class/leds/gsm_pwren/brightness**

GSM Reset:

- root@arm:~# **echo 1 > /sys/class/leds/gsm_reset/brightness; sleep 3; echo 0 > /sys/class/leds/gsm_reset/brightness**

2.10.29 MIPI-CSI CAMERA

Devices already tested:

MODEL	CORE	RESOLUTION
ALINX AN5641	OV5640	QSXGA (2592x1944), 1080p, 1280x960, VGA (640x480)

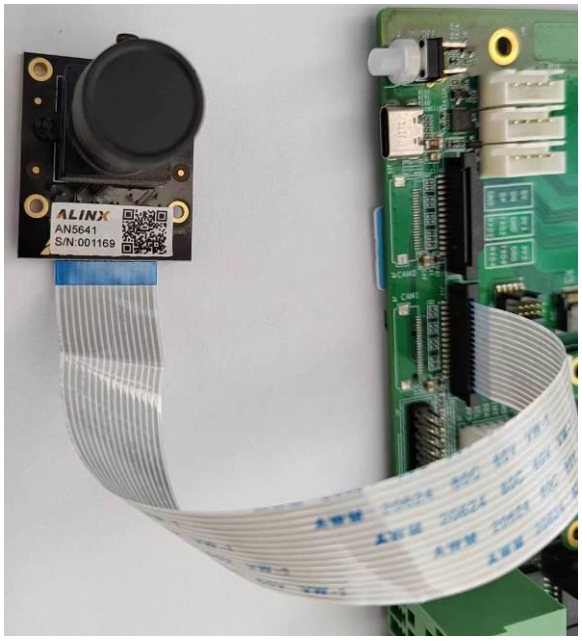


Figure 2-5 AN5641

- root@arm:~# **dmesg | grep img**

```
[    7.237305] mx8-img-md: Registered mxc_isi.0.capture as /dev/video3
[    7.244081] mx8-img-md: Registered mxc_isi.1.capture as /dev/video4
[    7.250703] mx8-img-md: Registered sensor subdevice: ov5640 2-003c (1)
[    7.257573] mx8-img-md: created link [mxc_isi.0] => [mxc_isi.0.capture]
[    7.264387] mx8-img-md: created link [mxc-mipi-csi2.0] => [mxc_isi.0]
[    7.270974] mx8-img-md: created link [mxc_isi.1] => [mxc_isi.1.capture]
[    7.277719] mx8-img-md: created link [mxc-mipi-csi2.1] => [mxc_isi.1]
[    7.284267] mx8-img-md: created link [ov5640 2-003c] => [mxc-mipi-csi2.1]
```

We can find out their relevance: ov5640 -> mxc-mipi-csi2.1 -> mxc_isi.1 ->

/dev/video4. Then we know the current camera device node is **/dev/video4**.

Camera Test:

- root@arm:~# **gst-launch-1.0 v4l2src device=/dev/video4 ! video/x-raw,width=1920,height=1080 ! waylandsink window-width=1280 window-height=720**

```
[  397.031883] mxc-mipi-csi2.0: mipi_csis_imx8mp_phy_reset, No remote pad found!
Setting pipeline to PAUSED ...
Pipeline is live and does not need PREROLL ...
Pipeline is PREROLLED ...
Setting pipeline to PLAYING ...
New clock: GstSystemClock
[  397.565658] bypass csc
[  397.568029] input fmt YUV4
[  397.570763] output fmt YUYV
Redistribute latency...
0:00:03.0 / 99:99:99.
```

Now we can see the real-time image stream captured by the camera is displaying on Wayland desktop.

2.10.30 SUSPEND and RESUME

Suspend to ram:

- root@arm:~# **echo mem > /sys/power/state**

```
[ 1980.810526] PM: suspend entry (deep)
[ 1980.875086] Filesystems sync: 0.060 seconds
[ 1980.880938] Freezing user space processes ... (elapsed 0.001 seconds) done.
[ 1980.889475] OOM killer disabled.
[ 1980.892714] Freezing remaining freezable tasks ... (elapsed 0.001 seconds) done.
[ 1980.901384] printk: Suspending console(s) (use no_console_suspend to debug)
[Click the ON/OFF KEY on the base board...]
[ 1983.825649] usb usb3-port1: device 3-1 not suspended yet
[ 1983.854400] imx-dwmac 30bf0000.ethernet eth1: Link is Down
[ 1983.855317] imx-dwmac 30bf0000.ethernet eth1: FPE workqueue stop
[ 1983.862950] PM: suspend devices took 2.952 seconds
[ 1983.865208] Disabling non-boot CPUs ...
[ 1983.866666] psci: CPU1 killed (polled 0 ms)
[ 1983.868102] psci: CPU2 killed (polled 4 ms)
```

```
[ 1983.869913] psci: CPU3 killed (polled 0 ms)
[ 1983.870360] Enabling non-boot CPUs ...
[ 1983.870738] Detected VIPT I-cache on CPU1
[ 1983.870762] GICv3: CPU1: found redistributor 1 region 0:0x00000000388a0000
[ 1983.870799] CPU1: Booted secondary processor 0x0000000001 [0x410fd034]
[ 1983.871255] CPU1 is up
[ 1983.871575] Detected VIPT I-cache on CPU2
[ 1983.871589] GICv3: CPU2: found redistributor 2 region 0:0x00000000388c0000
[ 1983.871609] CPU2: Booted secondary processor 0x0000000002 [0x410fd034]
[ 1983.871933] CPU2 is up
[ 1983.872278] Detected VIPT I-cache on CPU3
[ 1983.872292] GICv3: CPU3: found redistributor 3 region 0:0x00000000388e0000
[ 1983.872312] CPU3: Booted secondary processor 0x0000000003 [0x410fd034]
[ 1983.872658] CPU3 is up
[ 1983.968441] imx-dwmac 30bf0000.ethernet eth1: configuring for phy/rgmii-id link mode
[ 1983.979122] imx-dwmac 30bf0000.ethernet eth1: No Safety Features support found
[ 1983.979144] imx-dwmac 30bf0000.ethernet eth1: IEEE 1588-2008 Advanced Timestamp
supported
[ 1983.979586] imx-dwmac 30bf0000.ethernet eth1: FPE workqueue start
[ 1984.126218] caam 30900000.crypto: registering rng-caam
[ 1984.126241] xhci-hcd xhci-hcd.1.auto: xHC error in resume, USBSTS 0x401, Reinit
[ 1984.126251] usb usb1: root hub lost power or was reset
[ 1984.126255] usb usb2: root hub lost power or was reset
[ 1984.126270] xhci-hcd xhci-hcd.2.auto: xHC error in resume, USBSTS 0x401, Reinit
[ 1984.126277] usb usb3: root hub lost power or was reset
[ 1984.126281] usb usb4: root hub lost power or was reset
[ 1984.469789] PM: resume devices took 0.596 seconds
[ 1984.645221] OOM killer enabled.
[ 1984.648365] Restarting tasks ...
[ 1984.648645] usb 3-1: USB disconnect, device number 3
[ 1984.656957] done.
[ 1984.659421] PM: suspend exit
[ 1984.792080] usb 3-1: new high-speed USB device number 4 using xhci-hcd
[ 1984.991143] hub 3-1:1.0: USB hub found
[ 1984.995313] hub 3-1:1.0: 7 ports detected
```

Note:

🕮 The user key **S1** on the base board also support wake-up function.