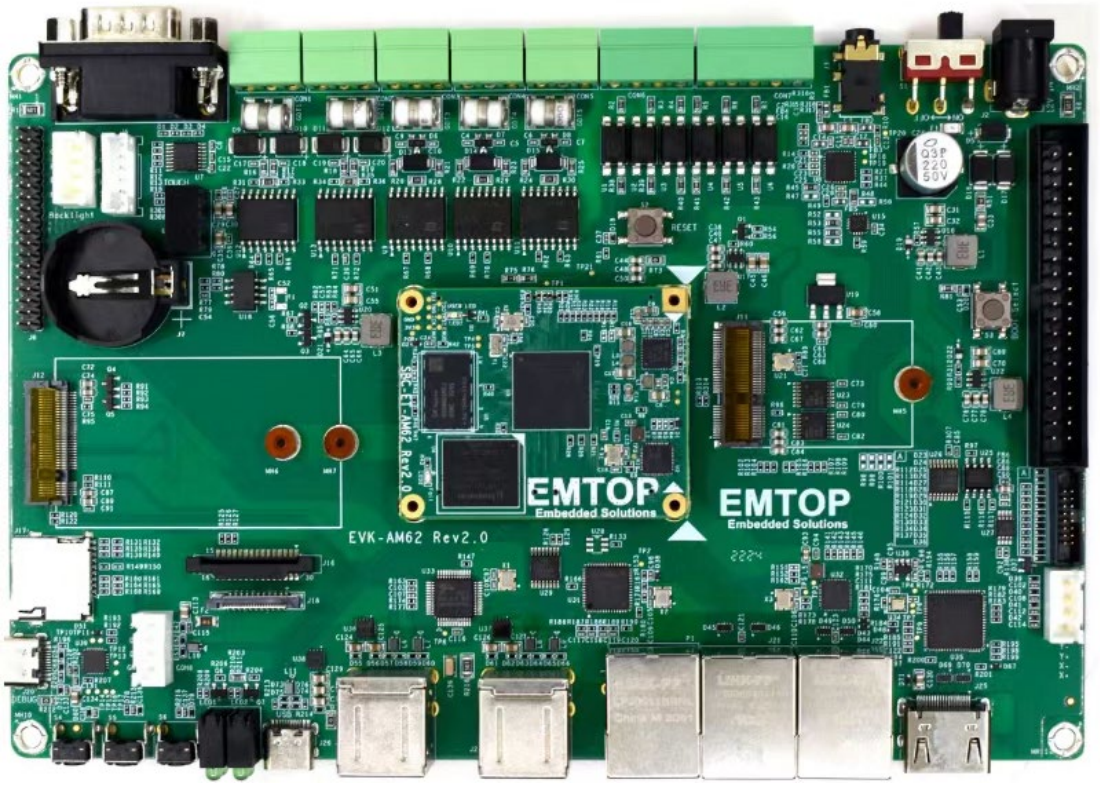
# EVK-ET-AM62




SOM-ET-AM62 + EVK-ET-AM62

# User Manual

Version: 0.6

2025-01-09

# Revision History：

| Version | Date | Description |
|---|---|---|
| 0.1 | 2024-05-13 | Trial Release |
| 0.2 | 2024-08-21 | Support M2 WiFi/BT, GSM |
| 0.3 | 2024-09-19 | Update document |
| 0.4 | 2024-10-05 | Fix DI/DO operation |
| 0.5 | 2024-11-13 | Rename to EVK-ET-AM62 |
| 0.6 | 2025-01-09 | Update u-boot to choose DDR on BOOTMODE |

# Table of Contents

**1. PRODUCT OVERVIEW ..... 6**

1.1 INTRODUCTION ..... 6

1.2 RESOURCE DOWNLOAD ..... 6

1.3 HARDWARE FEATURES ..... 6

1.4 MECHANICAL DIMENSION ..... 6

**2. LINUX OPERATION SYSTEM ..... 7**

2.1 SOFTWARE RESOURCES ..... 7

*2.1.1 Location of Resources ..... 7*

*2.1.2 BSP ..... 8*

2.2 STRUCTURE OF EMBEDDED LINUX SYSTEM ..... 9

2.3 BUILDING DEVELOPMENT ENVIRONMENT ..... 9

*2.3.1 Installing Cross Compilation Tools ..... 10*

*2.3.2 Set Cross Compile Environment ..... 10*

2.4 PREPARING THE SOURCE CODE ..... 11

2.5 COMPILATION ..... 11

2.6 YOCTO SDK ..... 14

*2.6.1 Build Yocto SDK ..... 15*

*2.6.2 Install Yocto SDK ..... 17*

*2.6.3 Compile User Application Program Project ..... 17*

2.7 LINUX SYSTEM CUSTOMIZATION ..... 18

*2.7.1 Replace U-BOOT LOGO ..... 18*

*2.7.2 Setting Configuration Menu ..... 18*

*2.7.3 Menu Options ..... 19*

*2.7.4 Compile Kernel ..... 20*

2.8 INTRODUCTION TO DRIVERS ..... 20

*2.8.1 SD/MMC ..... 22*

*2.8.2 Audio In/Out ..... 23*
2.9 DRIVER DEVELOPMENT ..... 24
*2.9.1 GPIO_LEDs Driver ..... 24*
*2.9.2 PINMUX Configuration Guide ..... 27*
2.10 SYSTEM UPDATE ..... 30
*2.10.1 Update TF Card System Image ..... 30*
*2.10.2 Update eMMC with TFCard ..... 31*
2.11 TEST AND DEMONSTRATION ..... 34
*2.11.1 SSH LOGIN ..... 34*
*2.11.2 RTC ..... 34*
*2.11.3 TIMEZONE ..... 35*
*2.11.4 USB HOST ..... 35*
*2.11.5 NETWORK ..... 37*
*2.11.6 TFT-LCD ..... 38*
*2.11.7 LVDS ..... 38*
*2.11.8 LVDS BACKLIGHT ..... 38*
*2.11.9 TOUCH PANEL ..... 38*
*2.11.10 NAU88C22 AUDIO ..... 41*
*2.11.11 HDMI ..... 41*
*2.11.12 HDMI AUDIO ..... 42*
*2.11.13 UART ..... 42*
*2.11.14 RS485 ..... 44*
*2.11.15 EEPROM ..... 45*
*2.11.16 CAN BUS ..... 45*
*2.11.17 BUTTON ..... 46*
*2.11.18 LED ..... 47*
*2.11.19 DI/DO ..... 48*
*2.11.20 SPI ADC ..... 49*

2.11.21 PWM ..... 51
2.11.22 eMMC ..... 51
2.11.23 SPIFLASH ..... 51
2.11.24 GPMC ..... 52
2.11.25 M.2/KEY B [WIFI and BLUETOOTH] ..... 53
2.11.26 M.2 WIFI ..... 53
2.11.27 M.2 BLUETOOTH ..... 56
2.11.28 M.2 4G/5G MODULE ..... 57
2.11.29 MIPI-CSI CAMERA ..... 59
2.11.30 WAYLAND GPU ..... 61

# 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-ti-2023.04 |
| | CD\Source\linux-ti -6.1.33 |
| | CD\Source\App |
| **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-6.1.33 | 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 | NAU88C22YG 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 |
| | WIFI/BT | NXP 88W8987 driver | Source Code |
| ROOTFS | YOCTO | Wayland with Qt 5.15 | Image |

## 2.2 Structure of Embedded Linux System

EVK-ET-AM62 is shipped with Linux-6.1.33 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 | FAT | EXT4 |
| Image | Bootloader, DTB, Kernel | Yocto Rootfs |

1) Bootloader is a program generated by u-boot compiling; they are **tiboot3.bin**, **tispl.bin** and **u-boot.img**.
2) The kernel used in this document is Linux-6.1.33 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 **Ubuntu22.04** 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: **arm-gnu-toolchain-11.3.rel1-x86_64-aarch64-none-linux-gnu.tar.xz**, **arm-gnu-toolchain-11.3.rel1-x86_64-arm-none-linux-gnueabihf.tar.xz** and **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 mkdir -p /opt/bin/arm**
- **sudo tar -xvf <YOUR_PATH>/arm-gnu-toolchain-11.3.rel1-x86_64-aarch64-none-linux-gnu.tar.xz -C /opt/bin/arm**
- **sudo tar -xvf <YOUR_PATH>/arm-gnu-toolchain-11.3.rel1-x86_64-arm-none-linux-gnueabihf.tar.xz -C /opt/bin/arm**
- **sudo tar -xvf <YOUR_PATH>/gcc-linaro-7.5.0-2019.12-x86_64_aarch64-linux-gnu.tar.xz -C /opt/bin/arm**

It will extract and install under **/opt/bin/arm** 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/bin/arm/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, or get source code under **Source** directory.

- **tar -xvf u-boot-ti-2023.04-git-xxxxxx.tar.xz**
- **tar -xvf linux-ti-6.1.33-git-xxxxxx.tar.xz**

Then we can get the source code directory **u-boot-ti-2020.04** and **linux-ti-6.1.33**.

## 2.5 Compilation

### 1） Compiling Bootloader

Run the following commands to compile bootloader:

- **cd u-boot-ti-2023.04**
- **vi make.sh**

```
export PATH=/opt/bin/arm/arm-gnu-toolchain-11.3.rel1-x86_64-aarch64-none-linux-gnu/bin:$PATH
export PATH=/opt/bin/arm/arm-gnu-toolchain-11.3.rel1-x86_64-arm-none-linux-gnueabihf/bin:$PATH

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 setup**

This command will install several components that need to be called during the compilation process.

- **./make.sh**

After all the command is successfully completed, you can find the booting images under **DESTDIR** directory:

```
DDR1G
├── tiboot3.bin
├── tispl.bin
└── u-boot.img
DDR2G
├── tiboot3.bin
├── tispl.bin
└── u-boot.img
DDR4G
├── tiboot3.bin
├── tispl.bin
└── u-boot.img
```

If you only need to compile bootloader for DDR1G, DDR2G or DDR4G, please run the below command to compile one of them:

- **./make.sh 1g**
- **./make.sh 2g**
- **./make.sh 4g**

The bootloader can choose the correct image according to BOOTMODE settings:

| BOOTMODE[15] | BOOTMODE[14] | DDR TYPE |
|---|---|---|

| | | |
|---|---|---|
| 0 | 1 | 1GB |
| 0 | 0 | 2GB |
| 1 | 0 | 4GB |

Copy the target image to SD card:

```
SD-FAT-PARTITION/
├── DDR1G
│   ├── tiboot3.bin
│   ├── tispl.bin
│   └── u-boot.img
├── DDR2G
│   ├── tiboot3.bin
│   ├── tispl.bin
│   └── u-boot.img
├── DDR4G
│   ├── tiboot3.bin
│   ├── tispl.bin
│   └── u-boot.img
├── tiboot3.bin
├── tispl.bin
└── u-boot.img
```

Note: **Copy the bin files under any directory to the root directory.**

### 2） Compiling Kernel

Execute the following instructions to compile kernel:

- **cd linux-ti-6.1.33**
- **git checkout .**
- **vi make.sh**

```
export PATH=/opt/bin/arm/gcc-linaro-7.5.0-2019.12-x86_64_aarch64-linux-gnu/bin:$PATH
export ARCH=arm64
export CROSS_COMPILE=aarch64-linux-gnu-

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/6.1.33** under **DESTDIR** directory.

**Note:**

The command **./make.sh**:

```
./make.sh                  # build dtbs and Image
./make.sh modules          # build dtbs, Image and driver modules
```

## 2.6 Yocto SDK

There are 2 ways to compile user application program [Linux C/C++ project]:

① Compile on Board;

② Compile with Yocto SDK under PC Ubuntu system.

If the application code is simple, and doesn't include too much third party libraries, you can try to compile it on board. For example, let us try to compile the UART program [**Source/App/com.tar.xz**]:

- root@arm:~# **vi Makefile**

```
# CROSS_COMPILE ?= arm-linux-

all:
	$(CROSS_COMPILE)gcc -s -o com com_example.c

clean:
	rm -fr com *~
```

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

```
gcc -s -o com com_example.c
```

Now, we already get the target ELF file **com**. Try to run and check:

- root@arm:~# **./com -h**

```
Usage: ./com option [ dev... ]
        -h  --help      Display this usage information.
        -d  --device    The device ttyS[0-3] or ttyEXT[0-3]
        -s  --string    Write the device data
        -f  --flow      Flow control switch
        -b  --speed     Set speed bit/s
        -m  --mode      Set mode, rs232 or rs485
        -n  --count     Repeat times, default non-stop
```

If the application code is complex, and we can't compile it on board successfully, please try to compile it under Ubuntu with Yocto SDK.

### 2.6.1 Build Yocto SDK

**Note：**

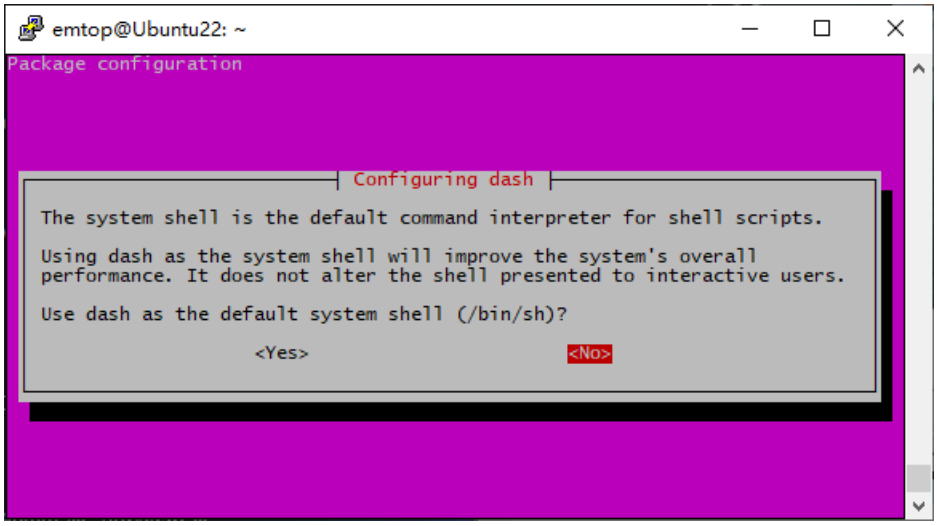
📖 This is an **OPTIONAL** operation step. Only useful when you want to build your own SDK.

Please refer to TI document: **https://software-dl.ti.com/processor-sdk-linux-rt/esd/AM62X/09_00_00_03/exports/docs/linux/Overview_Building_the_SDK.html**

[HOST] Ubuntu 22.04

- **sudo apt-get update**
- **sudo apt-get -f -y install git build-essential diffstat texinfo gawk chrpath socat doxygen dos2unix python3 bison flex libssl-dev u-boot-tools mono-devel mono-complete curl python3-distutils repo pseudo python3-sphinx g++-multilib libc6-dev-i386 jq git-lfs pigz zstd liblz4-tool cpio file zstd lz4**
- **sudo dpkg-reconfigure dash**

Be sure to select “No” when you are asked to use dash as the default system shell:




- **git clone https://git.ti.com/git/arago-project/oe-layersetup.git tisdk**
- **cd tisdk**
- **./oe-layertool-setup.sh -f configs/processor-sdk/processor-sdk-09.00.00-config.txt**
- **cd build**
- **. conf/setenv**
- **export TOOLCHAIN_PATH_ARMV7=/opt/bin/arm/arm-gnu-toolchain-11.3.rel1-x86_64-arm-none-linux-gnueabihf**
- **export TOOLCHAIN_PATH_ARMV8=/opt/bin/arm/arm-gnu-toolchain-11.3.rel1-x86_64-aarch64-none-linux-gnu**
- **vi ../sources/meta-processor-sdk/recipes-graphics/powervr-graphics/powervr-graphics_5.10.bb**

```
DESCRIPTION = "Imagination PowerVR SDK binaries/examples"
LICENSE = "MIT"

SRC_URI = " \
    gitsm://github.com/powervr-graphics/Native_SDK.git;protocol=https;branch=${BRANCH} \
    file://0001-PATCH-use-library-so-names-for-linking.patch \
"
```

- **vi /etc/hosts**

```
ff02::1 ip6-allnodes
```

```
ff02::2 ip6-allrouters

185.199.108.133 raw.githubusercontent.com
```

Build system image:

- **MACHINE=am62xx-evm ARAGO_RT_ENABLE=1 bitbake tisdk-default-image**

It will generate **arago-tmp-default-glibc/deploy/images/am62xx-evm/tisdk-default-image-am62xx-evm.wic.xz**.

**Note:**

🕮 The wic.xz is a **soft link**, please check to find the real file it points to.

### 2.6.2 Install Yocto SDK

Install SDK **Tools/arago-2023.04.tar.xz** with the below command:

- **sudo tar -xvf <YOUR_PATH>/arago-2023.04.tar.xz -C /opt**

### 2.6.3 Compile User Application Program Project

Let us try to compile a Qt example **easing**, which is from **qt-everywhere-src-5.15.0/qtbase/examples/widgets/animation/easing**. Get it from **Source/easing.tar.xz**.

- **source /opt/arago-2023.04/environment-setup**
- **cd easing**
- **qmake**

```
Info: creating stash file easing/.qmake.stash
```

- **make**

```
.....
aarch64-oe-linux-g++ --sysroot=/opt/arago-2023.04/sysroots/aarch64-oe-linux --sysroot=/opt/arago-2023.04/sysroots/aarch64-oe-linux --sysroot=/opt/arago-2023.04/sysroots/aarch
```

```
64-oe-linux -Wl,-O1 -o easing main.o window.o qrc_easing.o moc_window.o /opt/arago-2023.04/sysroots/aarch64-oe-linux/usr/lib/libQt5Widgets.so /opt/arago-2023.04/sysroots/aarch64-oe-linux/usr/lib/libQt5Gui.so /opt/arago-2023.04/sysroots/aarch64-oe-linux/usr/lib/libQt5Core.so -lGLESv2 -lpthread
```

- **file easing**

```
easing: ELF 64-bit LSB shared object, ARM aarch64, version 1 (GNU/Linux), ...
```

Copy the target file **easing** to ARM board, run it and you can see the animation widget.

## 2.7 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.

### 2.7.1 Replace U-BOOT LOGO

- Prepare a 24/32 bpp RGB888 bmp figure, rename it to **emtop-logo.bmp**;
- zip it under Ubuntu system:
  - **gzip emtop-logo.bmp**

Generate **emtop-logo.bmp.gz**.

- Replace the file under eMMC/SD FAT partition.

### 2.7.2 Setting Configuration Menu

A default configuration file is provided under kernel source codes:

**linux-ti-6.1.33/kernel/configs/emtop-sbc-et-am62.config**

Please execute the following commands to enter the configuration menu:

- **cd linux-ti-6.1.33**
- **export PATH=/opt/bin/arm/gcc-linaro-7.5.0-2019.12-x86_64_aarch64-linux-gnu/bin:$PATH**
- **export ARCH=arm64**
- **export CROSS_COMPILE=aarch64-linux-gnu-**
- **make defconfig ti_arm64_prune.config ti_rt.config emtop-sbc-et-am62.config**
- **make 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**

### 2.7.3 Menu Options

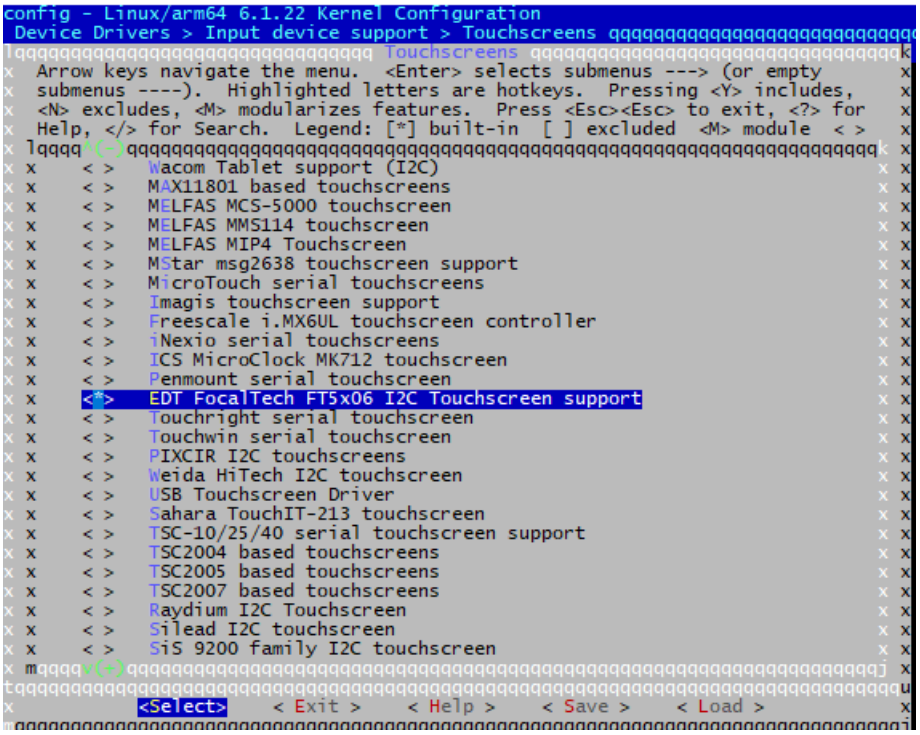
Configure options according to customization requirements after entering configu ration menu, for example, access **Device Drivers** > **Input device support** > **Touc hscreens** > **EDT FocalTech FT5x06 I2C Touchscreen support** as shown below:

-> Device Drivers

-> Input device support

-> Touchscreens

-> EDT FocalTech FT5x06 I2C Touchscreen support




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

### 2.7.4 Compile Kernel

Please execute the following instructions to recompile kernel:

- **./make.sh**

The script will **NOT** overwrite the configuration modified by menuconfig. It means 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**.

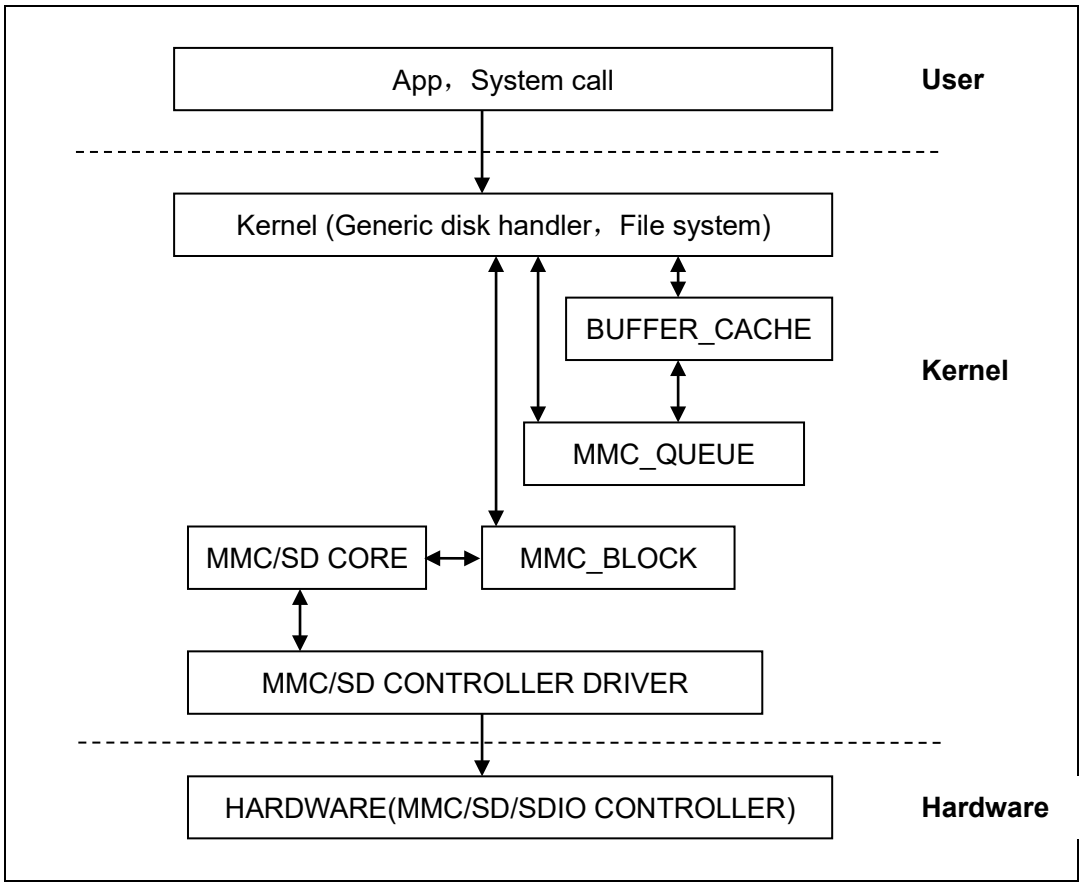
## 2.8 Introduction to Drivers

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

| Category | Name | Description | Location |
|---|---|---|---|

| | | | |
|---|---|---|---|
| **Bootloader** | U-BOOT | MMC/SD | drivers/mmc/am654_sdhci.c |
| | | FAT | fs/ |
| | | NET | drivers/net/ti/am65-cpsw-nuss.c |
| **Kernel** | Linux-6.1.33 | Support JFFS2/EXT4/FAT/NFS etc. | fs/ |
| **Devices** | SERIAL | Serial driver | drivers/tty/serial/8250/8250_omap.c |
| | RTC | Hardware RTC driver | drivers/rtc/rtc-rx8010.c |
| | NET | 10/100M/1000M Ethernet driver | drivers/net/ethernet/ti/am65-cpsw-nuss.c |
| | CAN | CAN bus driver | drivers/net/can/m_can/m_can_platform.c |
| | SPI | SPI driver | drivers/spi/spi-omap2-mcspi.c |
| | DSS | DSS driver | drivers/gpu/drm/tidss/tidss_drv.c |
| | HDMI | HDMI driver | drivers/gpu/drm/bridge/sii902x.c |
| | MIPI-DSI | MIPI-DSI driver | drivers/gpu/drm/panel/panel-simple.c |
| | TOUCH SCREEN | I2C touch panel driver | drivers/input/touchscreen/goodix.c |
| | MMC/SD | MMC/SD controller driver | drivers/mmc/host/sdhci_am654.c |
| | USB | USB controller driver | drivers/usb/dwc3/dwc3-am62.c |
| | AUDIO | NAU88C22 Audio driver(supports recording & playback) | sound/soc/codecs/nau8822.c |
| | BUTTON | GPIO button driver | drivers/input/keyboard/gpio_keys.c |
| | LED | LED driver | drivers/leds/leds-gpio.c |
| | WIFI/BT | NXP 88W8987 driver | 3rdparty/mwifiex-lf-6.1.22_2.0.0 |
| | CAMERA | CSI Camera driver | drivers/media/i2c/ov5640.c |

### 2.8.1 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;

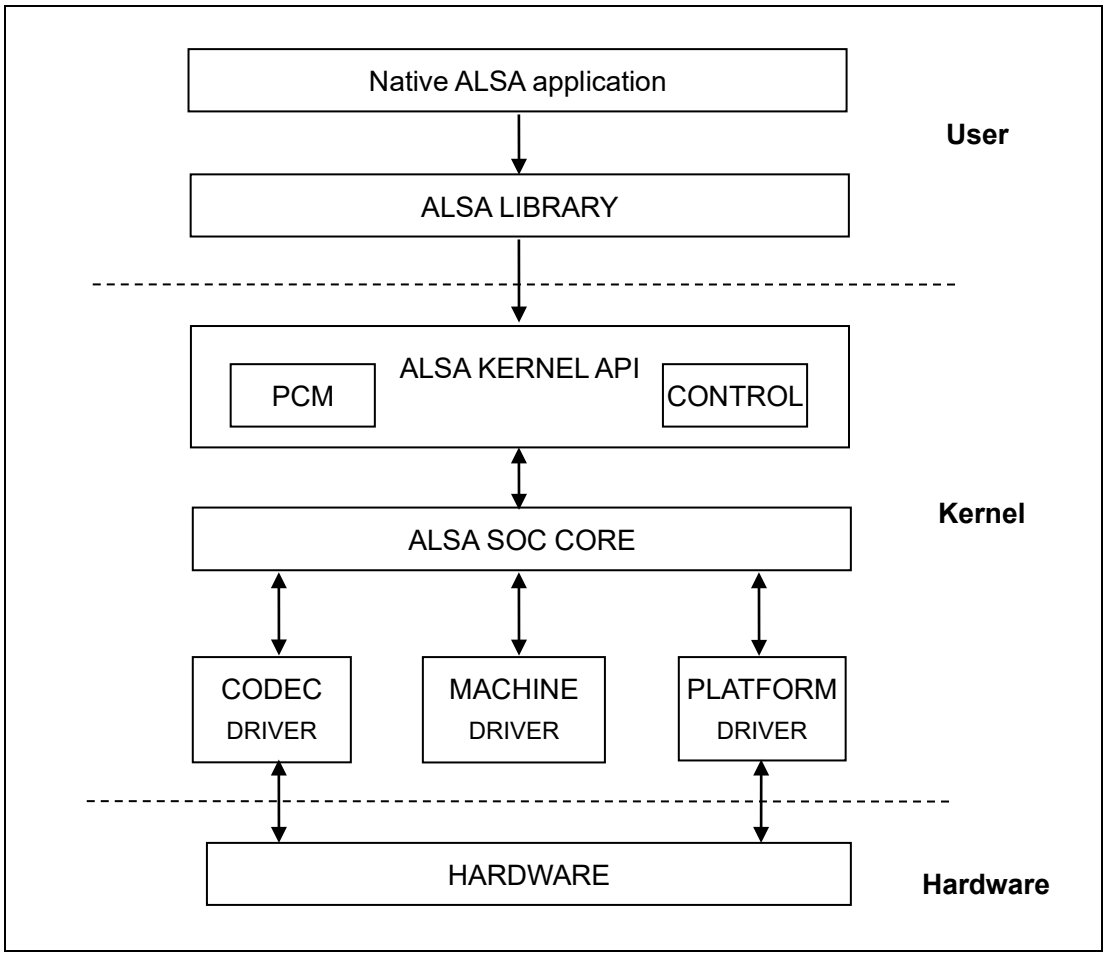
**4**） SD/MMC driver realizes specific controller driver.

**Drivers and relevant documents:**

linux-ti-6.1.33/drivers/mmc/

linux-ti-6.1.33/drivers/mmc/host/sdhci_am654.c

### 2.8.2 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-ti-6.1.33/sound/soc/ti

linux-ti-6.1.33/sound/soc/codecs/nau8822.c

# 2.9 Driver Development

## 2.9.1 GPIO_LEDs Driver

### 1）Device Definition

linux-ti-6.1.33/arch/arm64/boot/dts/ti/evk-et-am62.dts

Configure GPIO0_12 as system running status indicator, blinking as heartbeat.

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

        sys {
                label = "sys";
                gpios = <&main_gpio0 12 GPIO_ACTIVE_HIGH>;
                linux,default-trigger = "heartbeat";
        };
```

### 2）GPIO pinmux Configuration

linux-ti-6.1.33/arch/arm64/boot/dts/ti/evk-et-am62.dts

Configure OSPI0_CSN1 as GPIO0_12 function:

```
&main_pmx0 {
        ...
        usr_led_pins_default: usr-led-pins-default {
                pinctrl-single,pins = <
                        AM62X_IOPAD(0x030, PIN_OUTPUT, 7) /* (G21) OSPI0_CSN1.GPIO0
_12 */
                >;
        };
};
```

### 3）Driver Design

linux-ti-6.1.33/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@freescale.com>");
MODULE_DESCRIPTION("GPIO LED driver");
MODULE_LICENSE("GPL");
MODULE_ALIAS("platform:leds-gpio");
```

**b)** Request gpio and call led_classdev_register to register 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);
}
```

## 2.9.2 PINMUX Configuration Guide

AM625 has two types of GPIO: one controlled by A53, the other controlled by MCU:

A53 GPIOs are named as: GPIO0_12, GPIO1_31

MCU GPIOs are named as: MCU_GPIO0_22.

Configure the A53 GPIO pin attribute:

```
AM62X_IOPAD(0x01b0, PIN_OUTPUT, 7)  /* MCASP0_ACLKR.GPIO1_14 */
```

Configure the MCU GPIO pin attribute:

```
AM62X_MCU_IOPAD(0x0080, PIN_OUTPUT, 7) /* PMIC_LPM_EN0.MCU_GPIO0_22*/
```

Function syntax:

```
AM62X_IOPAD(pa, val, muxmode)      or   AM62X_MCU_IOPAD(pa, val, muxmode)
pa: Physical Address
val: value to write
muxmode: MUXMODE[3:0]
```

They are defined in **linux-ti-6.1.33/arch/arm64/boot/dts/ti/k3-pinctrl.h**:

```
#define AM62X_IOPAD(pa, val, muxmode)          (((pa) & 0x1fff)) ((val) | (muxmode))
#define AM62X_MCU_IOPAD(pa, val, muxmode)     (((pa) & 0x1fff)) ((val) | (muxmod
e))
```

Now, let's explain how to calculate the parameter 'pa' of PMIC_LPM_EN0.

Open document <AM62x Sitara™ Processors>, find out the physical address of pin PMIC_LPM_EN0 is 0x04084080:

| | | | |
|---|---|---|---|
| B7 | C7 | PMIC_LPM_EN0<br>PADCONFIG: MCU_PADCONFIG32 0x04084080 | PMIC_LPM_EN0<br>MCU_GPIO0_22 |

- **vi linux-ti-6.1.33/arch/arm64/boot/dts/ti/k3-am62-mcu.dtsi**

```
mcu_pmx0: pinctrl@4084000 {
    compatible = "pinctrl-single";
    reg = <0x00 0x04084000 0x00 0x88>;
    #pinctrl-cells = <1>;
    pinctrl-single,register-width = <32>;
    pinctrl-single,function-mask = <0xffffffff>;
};
```

It says the base physical address of mcu_pmx0 is **0x04084000**, and then we o

nly need to pass the offset address of pin PMIC_LPM_EN0 through macro AM62X _MCU_IOPAD:

$$offset = 0x04084080 - 0x04084000 = 0x80$$

About the parameter 'val', you can choose from below items:

- **vi linux-ti-6.1.33/arch/arm64/boot/dts/ti/k3-pinctrl.h**

```
/* Only these macros are expected be used directly in device tree files */
#define PIN_OUTPUT          (INPUT_DISABLE | PULL_DISABLE)
#define PIN_OUTPUT_PULLUP   (INPUT_DISABLE | PULL_UP)
#define PIN_OUTPUT_PULLDOWN (INPUT_DISABLE | PULL_DOWN)
#define PIN_INPUT           (INPUT_EN | PULL_DISABLE)
#define PIN_INPUT_PULLUP    (INPUT_EN | PULL_UP)
#define PIN_INPUT_PULLDOWN  (INPUT_EN | PULL_DOWN)
```

About the parameter 'muxmode', find 'Pin Attributes' table of document <AM62x Sitara™ Processors>:

| ALW BALL NUMBER [1] | AMC BALL NUMBER [1] | BALL NAME [2] PADCONFIG Register [15] PADCONFIG Address [16] | SIGNAL NAME [3] | MUX MODE [4] | TYPE [5] |
|---|---|---|---|---|---|
| J22 | J21 | OSPI0_D7 PADCONFIG: PADCONFIG10 0x000F4028 | OSPI0_D7 | 0 | IO |
| | | | SPI1_D1 | 1 | IO |
| | | | MCASP1_AFSX | 2 | IO |
| | | | UART6_CTSn | 3 | I |
| | | | GPIO0_10 | 7 | IO |
| B7 | C7 | PMIC_LPM_EN0 PADCONFIG: MCU_PADCONFIG32 0x04084080 | PMIC_LPM_EN0 | 0 | O |
| | | | MCU_GPIO0_22 | 7 | IO |

You can see the MUXMODE of MCU_GPIO0_22 is **7**.

If your target pin is controlled by A53, please append it under **main_pmx0** node in dts file; otherwise put it under **mcu_pmx0** node:

```
&mcu_pmx0 {
    usr_led_pins_default: usr-led-pins-default {
        pinctrl-single,pins = <
            AM62X_MCU_IOPAD(0x0080, PIN_OUTPUT, 7) /* (B7) PMIC_LPM_EN0.
MCU_GPIO0_22 */
        >;
    };
    ......
}
```

## 2.10 System Update

EVK-ET-AM62 can boot up from TF card and eMMC, it's decided by the **BOOT Select** button:

Press Down [Not Release]: Boot from TF card

Otherwise: Boot from eMMC.

### 2.10.1 Update TF Card System Image

**1） Make A Bootable TF Card**

- Get the system image from **Image** directory, named as **EVK-ET-AM62-SD-REVXX.img.xz**, unxz it and get the raw image **EVK-ET-AM62-SD-REVXX.img**.
- If you work under Windows system, please run **Tools/win32diskimager** to write the **EVK-ET-AM62-SD-REVXX.img** into TF Card. If you work under Linux system, please use **dd** command to write it into TF Card.

**2） Update U-Boot**

If you've made some changes to the u-boot source code, and want to update it into TF card, please copy the target image into the root directory of the TF card FAT partition:

```
├── tiboot3.bin
├── tispl.bin
└── u-boot.img
```

**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. If **lsmod** command shows nothing loaded after updating, Partition 2 [EXT4] **/lib/modules** also needs to be updated.

**4） Update Rootfs**

Because EXT4 isn't accessible Under Windows, please mount the Partition 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 0 && mmc erase 0 20000**

## 2.10.2 Update eMMC with TFCard

**Option 1: Write Complete Image into eMMC**

- Make a bootable TF card and boot up the system;
- Choose the target image [under directory **Image/**] and copy it into the USB disk [Formatted as NTFS or exFAT]. If it is **.xz** file, please unxz it to generate **.img** file;
- Install the USB disk on the ARM board, for example, the USB disk is recognized as **sda**;

- root@arm:~# **mount /dev/sda /mnt**

- Run command to start writing eMMC:
  - root@arm:~# **umount /dev/mmcblk0***

```
umount: /dev/mmcblk0: not mounted.
umount: /dev/mmcblk0boot0: not mounted.
umount: /dev/mmcblk0boot1: not mounted.
umount: /dev/mmcblk0p1: not mounted.
umount: /dev/mmcblk0p2: not mounted.
umount: /dev/mmcblk0rpmb: not mounted.
```

  - root@arm:~# **dd if=/mnt/EVK-ET-AM62-SD-REVXX.img of=/dev/mmcblk0 status=progress bs=4M**

- Run command to write bootloader [MUST]:
  - root@arm:~# **bootloader-update.sh 1g**

| | |
|---|---|
| **bootloader-update.sh 1g** | For 1GB DDR Device |
| **bootloader-update.sh 2g** | For 2GB DDR Device |
| **bootloader-update.sh 4g** | For 4GB DDR Device |

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

**Option 2: Synchronize eMMC with TF card**

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

```
running system update...
==============eMMC UPDATE=================
Warning: disk /dev/mmcblk0 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 (mmcblk0p2): 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]  mmcblk0: 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 (mmcblk0p2): mounted filesystem with ordered data mode. O
pts: (null). Quota mode: none.
```

- Power down the board and remove the TF card.

## 2.11 Test and Demonstration

This section will run some tests on the peripheral devices.

POWER: **12V DC**

Debug Port: **UART0**, **115200 1N8**, **USB TypeC slot [J20]**

### 2.11.1 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.

### 2.11.2 RTC

There is a RTC chip RX8010SJ on board, and the integrated RTC is also enabled by default. So there are two RTC devices accessible under system.

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

```
rtc-rx8010 0-0032
```

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

```
rtc-ti-k3 2b1f0000.rtc
```

That means the **rtc0** is RX8010SJ, and **rtc1** is the 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 2024-02-05 10:12:

- root@arm:~# **date -s "2024-02-05 10:12"; hwclock -f /dev/rtc0 -w**

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

- root@arm:~# **hwclock -f /dev/rtc0**

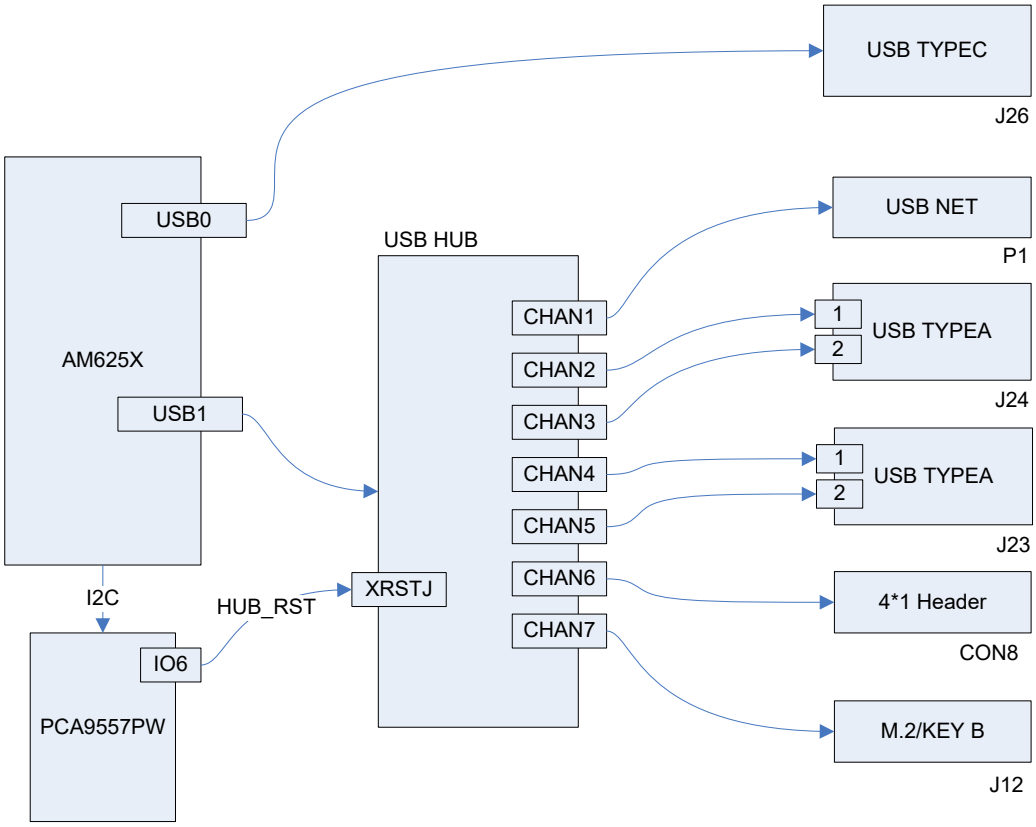
```
2024-02-05 10:12:07.365014+00:00
```

### 2.11.3 TIMEZONE

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**

### 2.11.4 USB HOST




We can test slots J26, J24 and J23 with USB disk. 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 **/media/sda1** by udev.

### 2.11.5 NETWORK

There are two 1Gbps network chips RTL8211F on board, and an USB net LAN9500A.

- 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 by 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
```

- 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
```

### 2.11.6 TFT-LCD

Devices already tested:

| MODEL | DESCRIPTION | DTB |
|---|---|---|
| LCD8000-70T | 800 * 480, with touch panel | emtop-evk-et-am62-lcd8000-800x480.dtb |

### 2.11.7 LVDS

Devices already tested:

| MODEL | DESCRIPTION | DTB |
|---|---|---|
| VISLCD-101HYS145ACT02 | 1920 * 1200, with touch panel | emtop-evk-et-am62.dtb |

**Figure 2-1** VISLCD-101HYS145ACT02

### 2.11.8 LVDS BACKLIGHT

- root@arm:~# **echo 5 > /sys/class/backlight/backlight/brightness**

**Note:**

The value of backlight level should be: **0** ~ **8**.

### 2.11.9 TOUCH PANEL

| MODEL | TYPE | I2C BUS |
|---|---|---|
| GT9271 | I2C CTP | I2C3 |

- 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:          generic gt9271
/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 330 (BTN_TOUCH)
  Event type 3 (EV_ABS)
    Event code 0 (ABS_X)
      Value      799
      Min          0
      Max        799
    Event code 1 (ABS_Y)
      Value      479
      Min          0
      Max        479
    Event code 47 (ABS_MT_SLOT)
      Value        0
      Min          0
      Max          9
    Event code 53 (ABS_MT_POSITION_X)
      Value        0
      Min          0
      Max        799
    Event code 54 (ABS_MT_POSITION_Y)
      Value        0
      Min          0
      Max        479
    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 1707131270.474572, type 3 (EV_ABS), code 53 (ABS_MT_POSITION_X), value 93
Event: time 1707131270.474572, type 3 (EV_ABS), code 54 (ABS_MT_POSITION_Y), value 93
Event: time 1707131270.474572, type 3 (EV_ABS), code 0 (ABS_X), value 93
Event: time 1707131270.474572, type 3 (EV_ABS), code 1 (ABS_Y), value 93
Event: time 1707131270.474572, ---------- SYN_REPORT ------------
Event: time 1707131270.641322, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID), value -1
Event: time 1707131270.641322, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 0
Event: time 1707131270.641322, ---------- SYN_REPORT ------------
Event: time 1707131271.588488, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID), value 5
Event: time 1707131271.588488, type 3 (EV_ABS), code 53 (ABS_MT_POSITION_X), value 156
Event: time 1707131271.588488, type 3 (EV_ABS), code 54 (ABS_MT_POSITION_Y), value 114
Event: time 1707131271.588488, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 1
Event: time 1707131271.588488, type 3 (EV_ABS), code 0 (ABS_X), value 156
Event: time 1707131271.588488, type 3 (EV_ABS), code 1 (ABS_Y), value 114
Event: time 1707131271.588488, ---------- SYN_REPORT ------------
Event: time 1707131271.791086, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID), value -1
Event: time 1707131271.791086, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 0
Event: time 1707131271.791086, ---------- SYN_REPORT ------------
Event: time 1707131272.186580, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID), value 6
Event: time 1707131272.186580, type 3 (EV_ABS), code 53 (ABS_MT_POSITION_X), value 107
Event: time 1707131272.186580, type 3 (EV_ABS), code 54 (ABS_MT_POSITION_Y), value 84
Event: time 1707131272.186580, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 1
Event: time 1707131272.186580, type 3 (EV_ABS), code 0 (ABS_X), value 107
Event: time 1707131272.186580, type 3 (EV_ABS), code 1 (ABS_Y), value 84
Event: time 1707131272.186580, ---------- SYN_REPORT ------------
Event: time 1707131272.361357, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID),
```

```
value -1
Event: time 1707131272.361357, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 0
Event: time 1707131272.361357, ---------- SYN_REPORT ------------
```

## 2.11.10 NAU88C22 AUDIO

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

```
**** List of PLAYBACK Hardware Devices ****
card 0: AM62xNAU8822 [AM62x-NAU8822], device 0: davinci-mcasp.0-nau8822-hifi nau8822-hifi-0 [davinci-mcasp.0-nau8822-hifi nau8822-hifi-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

**Playback**:

- root@arm:~# **for wav in `ls /usr/share/sounds/alsa/*.wav`; do aplay $wav; done**

**Record:**

- root@arm:~# **arecord -d 20 -c 1 -t wav -f cd test.wav**

## 2.11.11 HDMI

| MODEL | DESCRIPTION | DTB |
|---|---|---|
| HDMI Display | | emtop-evk-et-am62.dtb<br>emtop-evk-et-am62-hdmi.dtb |

**emtop-evk-et-am62.dtb**: Support LVDS and HDMI dual displays;

**emtop-evk-et-am62-hdmi.dtb**: Support only HDMI display.

## 2.11.12 HDMI AUDIO

| MODEL | DESCRIPTION | DTB |
|---|---|---|
| HDMI Display | Support audio | emtop-evk-et-am62-hdmi.dtb |

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

```
**** List of PLAYBACK Hardware Devices ****
card 0: AM62xSil9022HDM [AM62x-Sil9022-HDMI], device 0: davinci-mcasp.0-i2s-hifi i2s-hifi-0 [davinci-mcasp.0-i2s-hifi i2s-hifi-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

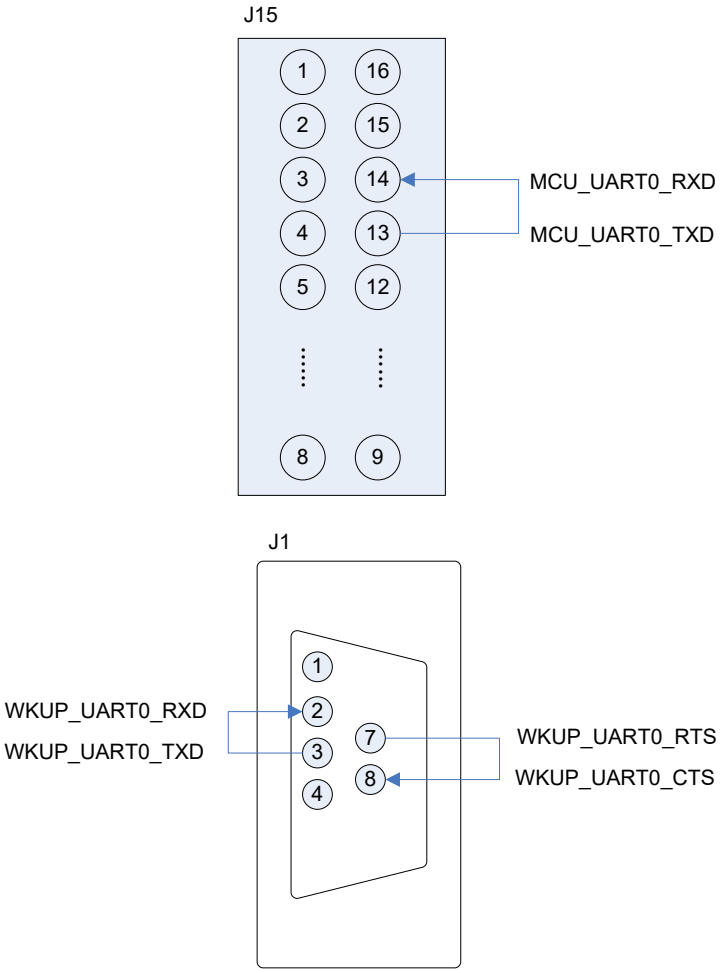
**Playback**:

- root@arm:~# **for wav in `ls /usr/share/sounds/alsa/*.wav`; do aplay $wav; done**

## 2.11.13 UART

| DEVICE NODE | HARDWARE | USAGE |
|---|---|---|
| /dev/ttyS2 | UART0 | DEBUG PORT |
| /dev/ttyS3 | UART1 | RS485 |
| /dev/ttyS7 | UART5 | RS485 |
| /dev/ttyS8 | UART6 | BLUETOOTH |
| /dev/ttyS9 | MCU_UART0 | HEADER J15 |
| /dev/ttyS10 | WKUP_UART0 | DB9 J1 |

Don't test UART0, UART1, UART5 and UART6 here. Let's test the others [MCU_UART0 and WKUP_UART0]. Connect their RXD and TXD pin:




And run below command:

- root@arm:~# **/test/app/com -d /dev/ttyS9**

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

The default baud rate is **115200**. If you want to assign another specific baud rate:

- root@arm:~# **/test/app/com -d /dev/ttyS9 -b 9600**

Please refer to the source code **com.tar.xz** for all supported baud rates.

- root@arm:~# **/test/app/com -d /dev/ttyS10 -f**

```
SEND: 1234567890
```

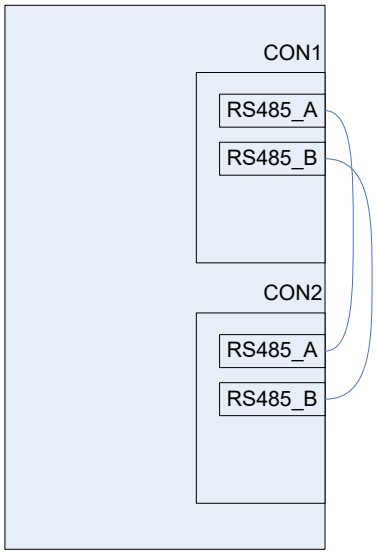
```
RECV: 1234567890
SEND: 1234567890
RECV: 1234567890
```

The parameter **-f** is to enable hardware flow control [RTS/CTS] feature.

### 2.11.14 RS485

| DEVICE NODE | HARDWARE | USAGE | REMARK |
|---|---|---|---|
| /dev/ttyS3 | UART1 | RS485 | |
| /dev/ttyS7 | UART5 | RS485 | with RTS |

There are 2 RS485 bus on board, let us connect them together.




- root@arm:~# **/test/app/com -d /dev/ttyS3 -s "Hello world" &**
- root@arm:~# **/test/app/com -d /dev/ttyS7 -m rs485**

```
SEND: 1234567890
SEND: Hello world
RECV: 1234567890
RECV: Hello world
SEND: 1234567890
SEND: Hello world
```

```
RECV: 1234567890
RECV: Hello world
```

## 2.11.15 EEPROM

There is an AT24LC32A on core board with write protection enabled.

- root@arm:~# **hexdump -Cv /sys/bus/i2c/devices/0-0050/eeprom**

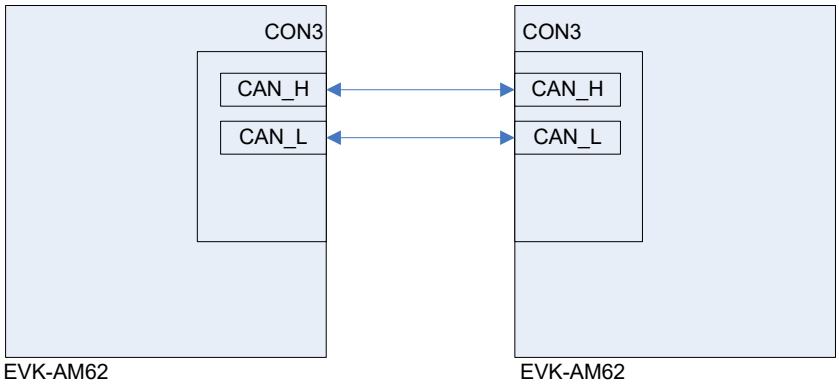
```
00000000  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
00000010  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
00000020  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
00000030  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
00000040  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
00000050  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
*
```

## 2.11.16 CAN BUS

There is 3 CAN bus on board.

| ITEM | HW INTERFACE | REGISTER BASEADDR | LINUX INTERFACE | SLOT |
|---|---|---|---|---|
| 1 | MCAN0 | 0x20701000 | can2 | CON3 |
| 2 | MCU_MCAN0 | 0x4e00000 | can0 | CON4 |
| 3 | MCU_MCAN1 | 0x4e10000 | can1 | CON5 |

Connect 2 CAN bus with 2 EVK-ET-AM62:




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

```
can0: flags=128<NOARP>  mtu 16
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 10 (U
NSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

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.

## 2.11.17 BUTTON

**PMIC_PBn** [**S4**] button: The default behavior is POWER-OFF.

User Button [**S5, S6**]:

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

```
No device specified, trying to scan all of /dev/input/event*
Available devices:
/dev/input/event0:          keys
/dev/input/event1:          tps65219-pwrbutton
/dev/input/event2:          ADS7846 Touchscreen
Select the device event number [0-2]: 0
```

```
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 102 (KEY_HOME)
    Event code 105 (KEY_LEFT)
Properties:
Testing ... (interrupt to exit)
Event: time 1707132374.777133, type 1 (EV_KEY), code 102 (KEY_HOME), value 1
Event: time 1707132374.777133, ---------- SYN_REPORT ------------
Event: time 1707132374.904456, type 1 (EV_KEY), code 102 (KEY_HOME), value 0
Event: time 1707132374.904456, ---------- SYN_REPORT ------------
Event: time 1707132375.518704, type 1 (EV_KEY), code 105 (KEY_LEFT), value 1
Event: time 1707132375.518704, ---------- SYN_REPORT ------------
Event: time 1707132375.615938, type 1 (EV_KEY), code 105 (KEY_LEFT), value 0
Event: time 1707132375.615938, ---------- SYN_REPORT ------------
```

### 2.11.18 LED

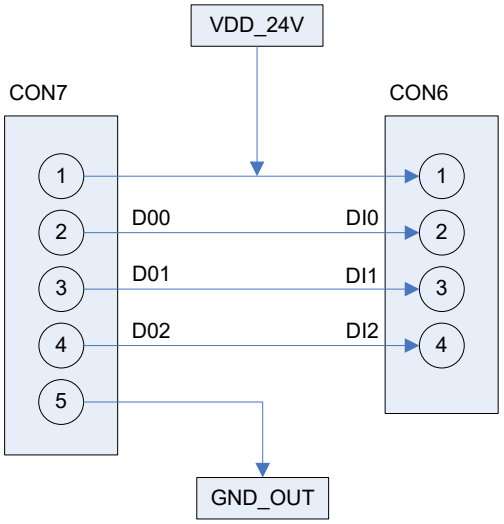
| LED | GPIO | LINUX DEVICE |
|---|---|---|
| SYS_nLED | GPIO0_12 | /sys/class/leds/sys |

There is only one LED controllable on board, the **SYS_nLED** keeps blinking to indicate system running status. But we can make it work as an ordinary LED:

- 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 **SYS_nLED** blinking with 2Hz frequency.

## 2.11.19 DI/DO




| DI/DO | GPIO | BASEADDR[HEX] | IO OFFSET |
|---|---|---|---|
| DO0 | GPIO0_0 | 600000 | 0 |
| DO1 | GPIO0_3 | 600000 | 3 |
| DO2 | GPIO0_4 | 600000 | 4 |
| DI0 | GPIO0_5 | 600000 | 5 |
| DI1 | GPIO0_2 | 600000 | 2 |
| DI2 | GPIO0_6 | 600000 | 6 |

We can get the GPIOCHIP number with the below command:

- root@arm:~# **gpiochip=`gpiodetect | awk '/600000/ {print $1}'`**

**Note:**

The **600000** is the **BASEADDR** of the corresponding GPIO.

Set DO0 output low:

- root@arm:~# **gpioset $gpiochip 0=0**

Set DO0 output heigh:

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

Set DO1 output low:

- root@arm:~# **gpioset $gpiochip 3=0**

**Note:**

🕮 **gpioset <GPIOCHIP> <IO OFFSET>=<output value>**

Read DI0 input:

- root@arm:~# **gpioget $gpiochip 5**

```
1 or 0
```

**Note:**

🕮 **gpioget <GPIOCHIP> <IO OFFSET>**

Or monitor the IO status changing event:

- root@arm:~# **gpiomon $gpiochip 5**

```
event:  RISING EDGE offset: 5 timestamp: [      1151.814356387]
event:  FALLING EDGE offset: 5 timestamp: [     1151.815449803]
event:  RISING EDGE offset: 5 timestamp: [      1152.091556803]
```

**Note:**

🕮 **libgpiod version is 1.6.3.**

## 2.11.20 SPI ADC

There is a SPI ADC chip ADC102S051 on baseboard.

- root@arm:~# **cat /sys/bus/iio/devices/iio:device0/name**

```
adc102s051
```

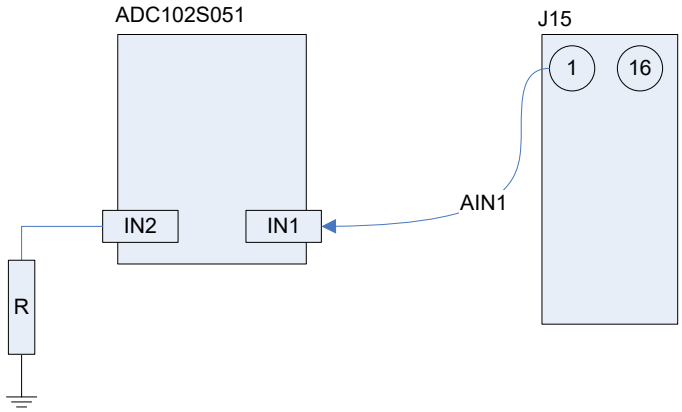



**Figure 2-2** SPI ADC Schematic

Read channel **IN1** input:

- root@arm:~# **cat /sys/bus/iio/devices/iio:device0/in_voltage0_raw**

```
1884
```

- root@arm:~# **cat /sys/bus/iio/devices/iio:device0/in_voltage0_scale**

```
0.805664062
```

Formula:

Voltage = **raw** * **scale** (mV)

Read channel **IN2** input:

- root@arm:~# **cat /sys/bus/iio/devices/iio:device0/in_voltage1_raw**

```
3
```

The channel IN2 is connected to GND, so it always read 0 approximately.

### 2.11.21 PWM

The MCU_GPIO0_8 on baseboard header [**J15**] is configured as PWM output as default.

- root@arm:~# **/test/app/mcu_timer1 1000**

The command above can set it output 1KHz with 50% duty circle.

- root@arm:~# **/test/app/mcu_timer1 1000 80**

The command above can set it output 1KHz with 80% duty circle.

- root@arm:~# **/test/app/mcu_timer1 -h**

```
Usage:
    /test/app/mcu_timer1 frequency [duty_circle] - Configure MCU_TIMER1 output under PWM mode

    frequency                   unit: Hz, maximum 17000000Hz[17MHz]
    duty_circle                 percentage, valid value: 0,10,20,...,90,100, 50 as default

Example:
    /test/app/mcu_timer1 1000           output 1KHz with 50% duty circle
    /test/app/mcu_timer1 1000 80        output 1KHz with 80% duty circle
    -h                                  display help info
```

### 2.11.22 eMMC

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

### 2.11.23 SPIFLASH

A SPIFlash XT25F64BSOIGT is equipped on baseboard.

- root@arm:~# **dmesg |grep -i spi-nor**

```
[    8.020943] spi-nor spi0.1: xt25f64 (8192 Kbytes)
```

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

```
dev:    size   erasesize  name
mtd0: 00800000 00010000 "spi0.1"
```

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**

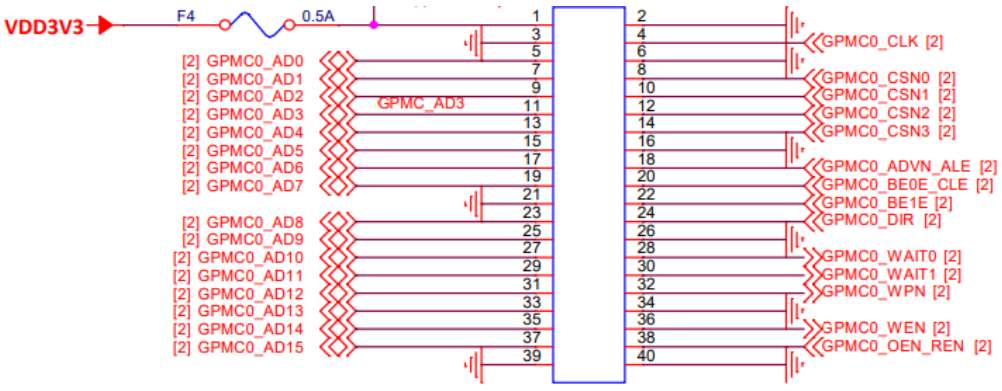
Write and read under directory **/mnt**, the content will keep in the QSPIFlash memory.

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

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

## 2.11.24 GPMC

There is a 16bit width GPMC bus on the baseboard.




The GPMC base address is: 0x51000000. We can access the bus with **devmem** tool:

- root@arm:~# **devmem 0x51000000**

## 2.11.25 M.2/KEY B [WIFI and BLUETOOTH]

Devices already tested:

| MODEL | CHIPSET | RESOLUTION |
|---|---|---|
| 1ZM M.2 Module | NXP 88W8987 | Support Wi-Fi and Bluetooth 5.1 |




**Figure 2-3** 1ZM M.2 Module

## 2.11.26 M.2 WIFI

- 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**

```
[    27.300912] mlan: loading out-of-tree module taints kernel.
[    27.391413] wlan: Loading MWLAN driver
[    27.392160] wlan: Register to Bus Driver...
[    27.478548] vendor=0x02DF device=0x9149 class=0 function=1
```

```
[    27.478688] Attach moal handle ops, card interface type: 0x105
[    27.478716] rps set to 0 from module param
[    27.478721] No module param cfg file specified
[    27.478736] SDIO: max_segs=128 max_seg_size=65536
[    27.478750] rx_work=1 cpu_num=4
[    27.478806] Attach mlan adapter operations.card_type is 0x105.
[    27.479449] wlan: Enable TX SG mode
[    27.479455] wlan: Enable RX SG mode
[    27.486150] Request firmware: sdiouart8987_combo_v0.bin
[    27.791167] Wlan: FW download over, firmwarelen=612304 downloaded 612304
[    28.662012] WLAN FW is active
[    28.662030] on_time is 28660259810
[    28.721744] fw_cap_info=0x181d7f03, dev_cap_mask=0xffffffff
[    28.721921] max_p2p_conn = 8, max_sta_conn = 8
[    28.722567] SDIO rx aggr: 1 block_size=512
[    28.722648] wlan: Enable RX SG mode
[    28.722651] mpa_rx_buf_size=65280
[    28.750785] Register NXP 802.11 Adapter wlan0
[    28.762135] Register NXP 802.11 Adapter wlan1
[    28.774227] Register NXP 802.11 Adapter p2p0
[    28.774334] wlan: version = SD8987----16.92.21.p76.5-MM5X16391.p3-GPL-(FP92)
[    28.775489] wlan: Register to Bus Driver Done
[    28.775511] wlan: Driver loaded successfully
```

- root@arm:~# **ifconfig wlan0 up**
- root@arm:~# **iw wlan0 scan | grep SSID**

```
[   106.018542] wlan: wlan0 START SCAN
[   111.007459] wlan: SCAN COMPLETED: scanned AP count=23
        SSID: EMTOP
......
```

- 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
[  254.521484] wlan: wlan0 START SCAN
[  259.510371] wlan: SCAN COMPLETED: scanned AP count=21
[  259.516932] wlan: HostMlme wlan0 send auth to bssid dc:XX:XX:XX:53:70
[  259.517747] wlan0:
[  259.517755] wlan: HostMlme Auth received from dc:XX:XX:XX:53:70
[  259.522576] CMD_RESP: cmd 0x121 error, result=0x2
[  259.522608] IOCTL failed: 000000005a18a418 id=0x200000, sub_id=0x200024 action=2, status_code=0x3
[  259.528757] wlan: HostMlme wlan0 Connected to bssid dc:XX:XX:XX:53:70 successfully
[  259.531799] wlan0:
[  259.531820] wlan: Send EAPOL pkt to dc:XX:XX:XX:53:70
[  259.539604] wlan0:
[  259.539632] wlan: Send EAPOL pkt to dc:XX:XX:XX:53:70
[  259.548222] IPv6: ADDRCONF(NETDEV_CHANGE): wlan0: link becomes ready
[  259.548858] 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
```

## 2.11.27 M.2 BLUETOOTH

- root@arm:~# **hciattach /dev/ttyS8 any 115200 flow**

```
[  447.897177] Bluetooth: Core ver 2.22
[  447.897924] NET: Registered PF_BLUETOOTH protocol family
[  447.897942] Bluetooth: HCI device and connection manager initialized
[  447.897977] Bluetooth: HCI socket layer initialized
[  447.897988] Bluetooth: L2CAP socket layer initialized
[  447.898038] Bluetooth: SCO socket layer initialized
[  447.920896] Bluetooth: HCI UART driver ver 2.3
[  447.920929] Bluetooth: HCI UART protocol H4 registered
Device setup complete
[  447.921092] Bluetooth: HCI UART protocol LL registered
[  447.921141] Bluetooth: HCI UART protocol Three-wire (H5) registered
[  447.922183] Bluetooth: HCI UART protocol Broadcom registered
[  447.922241] Bluetooth: HCI UART protocol QCA registered
[  447.922286] Bluetooth: HCI UART protocol Marvell registered
[  448.433149] Bluetooth: MGMT ver 1.22
[  448.446176] 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:**

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

## 2.11.28 M.2 4G/5G MODULE

Devices already tested:

| MODEL | DESCRIPTION |
|---|---|
| QUECTEL EM05-CE | 4G module |
| QUECTEL RM500Q-GL | 5G module |

Install GSM module, antenna, SIM card, and then power on the board.

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

```
Bus 002 Device 004: ID 2c7c:0800 Quectel Wireless Solutions Co., Ltd. RM500Q-GL
Bus 002 Device 003: ID 0424:9e00 Microchip Technology, Inc. (formerly SMSC) LAN9500A/LAN9500Ai
Bus 002 Device 002: ID 1a40:0201 Terminus Technology Inc. FE 2.1 7-port Hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

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

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

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

```
......
Serial connection established.
using channel 1
Using interface ppp0
Connect: ppp0 <--> /dev/ttyGSM03
sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x6586363d> <pcomp> <accomp>]
rcvd [LCP ConfReq id=0x0 <asyncmap 0x0> <auth chap MD5> <magic 0xafb6e7ff> <pcomp> <accomp>]
sent [LCP ConfAck id=0x0 <asyncmap 0x0> <auth chap MD5> <magic 0xafb6e7ff> <pcomp> <accomp>]
rcvd [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x6586363d> <pcomp> <accomp>]
rcvd [LCP DiscReq id=0x1 magic=0xafb6e7ff]
rcvd [CHAP Challenge id=0x1 <5d8494ff9ffeb38c9d39b711e1dc3f38>, name = "UMTS_CHAP_SRVR"]
```

```
sent [CHAP Response id=0x1 <b8f94ab38da425eb339f548b11d591c9>, name = "$LTE_USERNAME"]
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::2052:fff9:f87b:4287>]
rcvd [IPCP ConfReq id=0x0]
sent [IPCP ConfNak id=0x0 <addr 0.0.0.0>]
rcvd [IPCP ConfNak id=0x1 <addr 10.23.13.247> <ms-dns1 120.196.165.7> <ms-dns2 221.179.38.7>]
sent [IPCP ConfReq id=0x2 <addr 10.23.13.247> <ms-dns1 120.196.165.7> <ms-dns2 221.179.38.7>]
rcvd [IPCP ConfReq id=0x1]
sent [IPCP ConfAck id=0x1]
rcvd [IPCP ConfAck id=0x2 <addr 10.23.13.247> <ms-dns1 120.196.165.7> <ms-dns2 221.179.38.7>]
Could not determine remote IP address: defaulting to 10.64.64.64
not replacing default route to eth2 [192.168.3.1]
local  IP address 10.23.13.247
remote IP address 10.64.64.64
primary  DNS address 120.196.165.7
secondary DNS address 221.179.38.7
```

**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/lte_on/brightness**

**GSM Enable**

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

**GSM Reset:**

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

## 2.11.29 MIPI-CSI CAMERA

Devices already tested:

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

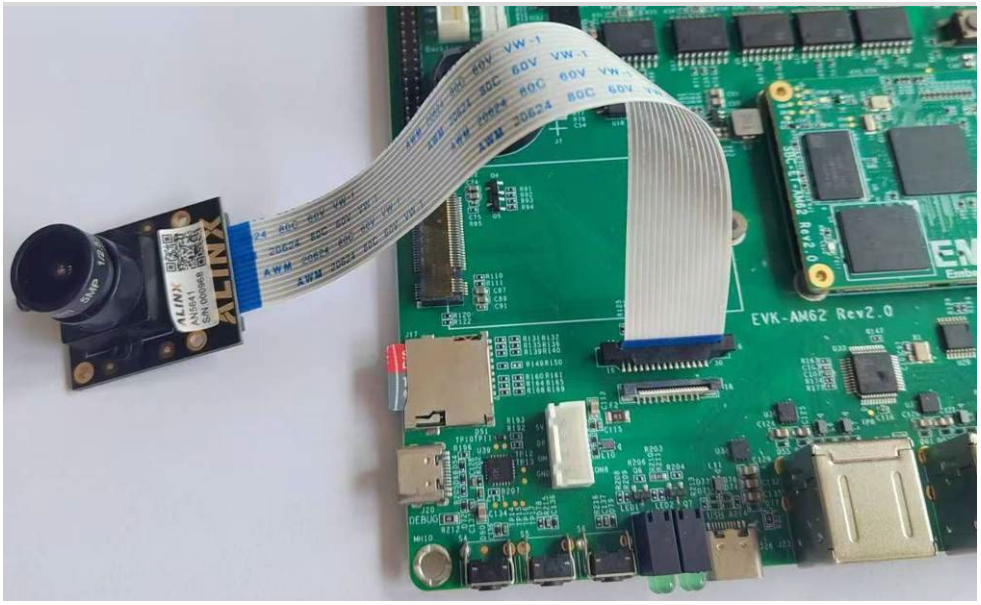



**Figure 2-4** AN5641

- root@arm:~# **media-ctl -p**

```
.....
- entity 13: ov5640 0-003c (1 pad, 1 link, 0 route)
                type V4L2 subdev subtype Sensor flags 0
                device node name /dev/v4l-subdev2
        pad0: Source
                [stream:0 fmt:UYVY8_1X16/640x480@1/30 field:none colorspace:srgb
xfer:srgb ycbcr:601 quantization:full-range
                 crop.bounds:(0,0)/2624x1964
                 crop:(16,14)/2592x1944]
                -> "cdns_csi2rx.30101000.csi-bridge":0 [ENABLED,IMMUTABLE]
.....
```

The camera device node is **/dev/video0**.

Camera Test:

- root@arm:~# **media-ctl -V '"30102000.ticsi2rx":0/0 [fmt:UYVY8_1X16/640x480 field:none]'**
- root@arm:~# **media-ctl -V '"cdns_csi2rx.30101000.csi-bridge":0/0 [fmt:UYVY8_1X16/640x480 field:none]'**
- root@arm:~# **media-ctl --set-v4l2 '"ov5640 0-003c':0[fmt:UYVY8_1X16/640x480 fi**

**eld:none]"**

- root@arm:~# **gst-launch-1.0 v4l2src device="/dev/video0" ! video/x-raw, width=640, height=480, format=UYVY ! autovideosink**

```
Setting pipeline to PAUSED ...
Pipeline is live and does not need PREROLL ...
Pipeline is PREROLLED ...
Setting pipeline to PLAYING ...
New clock: GstSystemClock
Redistribute latency...
0:00:02.4 / 99:99:99.
```

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

## 2.11.30 WAYLAND GPU

- root@arm:~# **glmark2-es2-wayland --run-foreve**

```
libEGL warning: egl: failed to create dri2 screen
=======================================================
    glmark2 2021.12
=======================================================
    OpenGL Information
    GL_VENDOR:      Mesa/X.org
    GL_RENDERER:    softpipe
    GL_VERSION:     OpenGL ES 3.1 Mesa 22.3.5 (git-54fd9d7dea)
=======================================================
[build] use-vbo=false: FPS: 1 FrameTime: 1000.000 ms
[build] use-vbo=true: FPS: 1 FrameTime: 1000.000 ms
......
```



**Figure 2-5** glmark2-es2-wayland