MYD-Y6ULX Linux Software Evaluation Guide

File Status:	FILE ID:	MYIR-MYD-Y6ULX-SW-EG-EN-L5.4.3
[√] Draft	VERSION:	V2.0.1
[ ] Release	AUTHOR:	Alex
	CREATED:	2020-07-08
	UPDATED:	2020-01-15

Revision History

VERSION	AUTHOR	PARTICIPANT	DATE	DESCRIPTION
V2.0.1	Alex		20210115	Initial Version: u-boot2019.04, Linux Kernel 5.4.3, Yocto 3.0.

CONTENT

MYD-Y6ULX Linux Software Evaluation Guide····· - 1 -

Revision History····· - 2 -

CONTENT····· - 3 -

1. Overview····· - 8 -

1.1. Hardware Resources····· - 8 -

1.2. Software Resources····· - 8 -

1.3. Documents····· - 9 -

1.4. Preparation····· - 9 -

2. Core Components····· - 10 -

2.1. CPU····· - 10 -

1) View the CPU Information····· - 10 -

2) View CPU Utilization····· - 11 -

3) Gets CPU Temperature····· - 11 -

4) CPU Stress Test····· - 12 -

2.2. Memory····· - 14 -

1) Check the Memory Information····· - 14 -

2) Get memory usage····· - 14 -

3) Memory Stress Test····· - 15 -

2.3. Flash····· - 17 -

1) Emmc····· - 17 -

2) Nand flash····· - 18 -

2.4. RTC····· - 20 -

2.5. Watchdog····· - 22 -

1) Stop the Watchdog····· - 22 -

2) Using the application to test the watchdog····· - 22 -

2.6. Power Manager····· - 24 -

1) View the Mode Supported by the Current Development Board···· - 24 -

2) Enter mem Power Management Mode····· - 24 -

3) Wake up through KEY BUTTON····· - 24 -

3. Peripheral Interface ····· - 26 -
3.1. GPIO ····· - 26 -
1) Export GPIO to User Space ····· - 26 -
2) Set/View the GPIO Direction ····· - 26 -
3.2. LED Lamp ····· - 28 -
1) View the LEDs ····· - 28 -
2) Test a LED ····· - 28 -
3.3. RS232 ····· - 30 -
3.4. RS485 ····· - 31 -
3.5. CAN ····· - 32 -
1) Initializes the CAN network interface ····· - 32 -
2) Send and Receive Data ····· - 32 -
3) Statistics of can0 information ····· - 33 -
3.6. Key ····· - 33 -
1) Device Tree Node ····· - 33 -
2) Test Keys ····· - 34 -
3.7. USB ····· - 36 -
1) Check the Kernel Message of USB ····· - 36 -
2) Mount and Read/Write the USB Flash Disk ····· - 37 -
3.8. Backlight ····· - 38 -
3.9. Touch Panel ····· - 39 -
1) Calibration ····· - 39 -
2) Touch test with evtest command ····· - 39 -
3) Print test information ····· - 41 -
3.10. Display ····· - 43 -
1) TEST LCD ····· - 43 -
2) Modify to support 4.3-inch LCD ····· - 44 -
4. Network Interface ····· - 46 -
4.1. Ethernet ····· - 46 -
1) Configure Ethernet IP addresses Manually and Temporarily ····· - 46 -
4.2. Wi-Fi ····· - 51 -
1) Manually connect to WiFi hotspot for STA mode ····· - 51 -

2) Connect to Wi-Fi Hotspots Automatically····· - 53 -
5. Network Applications····· - 56 -
5.1. PING····· - 56 -
1) Network Connection····· - 56 -
2) PING Public Network····· - 56 -
5.2. SSH····· - 57 -
5.3. SCP····· - 60 -
1) Copy Files from Remote to Local····· - 60 -
2) Copy Files form Local to Remote····· - 60 -
5.4. FTP····· - 61 -
1) Login from Host PC to Target Device with FTP····· - 61 -
2) Create Files on Target Device for Test····· - 62 -
3) View Files on Target Device with FTP at Host PC····· - 62 -
4) Download Files from the Target Device with FTP····· - 62 -
5) Upload Files to the Target Device with FTP····· - 63 -
5.5. TFTP····· - 65 -
1) Install TFTP Server Application····· - 65 -
2) Configure TFTP Server····· - 66 -
3) Restart TFTP Service····· - 66 -
5.6. UDHCPC····· - 67 -
5.7. IPTables····· - 69 -
1) Configure iptables for Target Device····· - 69 -
2) Ping the Target Device····· - 69 -
3) Delete Rules for iptables····· - 70 -
4) Ping the Target Device Again····· - 70 -
5.8. Ethtool····· - 71 -
5.9. iPerf3····· - 73 -
1) Test Performance under TCP Mode····· - 73 -
2) Test Performance under UDP Mode····· - 75 -
6. Linux Graphics System····· - 79 -
6.1. QT····· - 80 -
1) Introduction of QT Running Environment····· - 80 -

2) Start Qt Applications····· - 82 -
7. Multimedia····· - 84 -
7.1. Camera····· - 84 -
1) View the Device Topology····· - 84 -
2) Start Display Server····· - 84 -
7.2. Audio····· - 86 -
1) Play Music on MEasy HMI 2.0····· - 86 -
2) Play Music by Command····· - 86 -
7.3. Video····· - 87 -
8. System Tools····· - 88 -
8.1. Compress and Decompress Tools····· - 88 -
1) tar Tool····· - 88 -
2) gzip Tool····· - 90 -
8.2. File System Tools····· - 91 -
1) mount tool····· - 91 -
2) mkfs tool····· - 91 -
3) fsck tool····· - 93 -
4) dumpe2fs tool····· - 93 -
8.3. Disk Management Utils····· - 95 -
1) fdisk Disk Partitioning Tool····· - 95 -
2) dd tool····· - 95 -
3) du tool····· - 96 -
4) df tool····· - 97 -
8.4. Process Management Utils····· - 98 -
1) ps tool····· - 98 -
2) top tool····· - 100 -
3) vmstat: Virtual memory statistics tool····· - 100 -
4) kill tool····· - 103 -
9. Application Development····· - 105 -
9.1. Development Language····· - 105 -
1) SHELL····· - 105 -

2) C/C++ - 106 -
3) Python - 107 -
9.2. Database - 110 -
9.3. Application Localization for Qt - 112 -
1) Multiple Language - 112 -
2) Fonts - 114 -
3) Virtual Keyboard from Qt - 115 -
10. Reference - 117 -
Appendix A - 118 -
Warranty & Technical Support Services - 118 -

1. Overview

The Linux software evaluation guide describes the testing steps and evaluation methods for core and peripheral resources running open source Linux systems on MYIR Electronics' development boards. This paper can be used as a preliminary evaluation guide or as a test guide for general system developers.

1.1. Hardware Resources

This document applies to MYD-Y6ULX series boards of MYIR Electronics, which is a set of development platform based on i.MX6UL series. This development platform is composed of two parts, the core board MYC-Y6ULX and the bottom board MYB-Y6ULX. For detailed configuration parameters for the hardware part, please refer to the "MYD-Y6ULX Product Manual". The user will use some of the accessories during the evaluation test, see the list below.

Table 1-1. Optional Modules

Accessories	Interface	Description
Camera	USB	MY-CAM002U(2MegaPixels) http://www.myir-tech.com/product/my_cam002u.htm
LCD	RGB	MY-TFT070CV2 (7Inches LCD with Capacitive Touch Panel) http://www.myir-tech.com/product/my-tft070cv2.htm
4G Module (optional)	USB on Mini PCIe	EC20 https://www.quectel.com/cn/product/ec20r21minipcle.htm

1.2. Software Resources

The BSP of MYD-Y6ULX series development board is based on the transplantation and modification of The Linux BSP of NXP official open source community edition, and the system image is built by the Yocto project. All software resources of Bootloader, Kernel and file system are open in the form of source code, please check the "MYD-Y6ULX SDK2.0.0 Release Notes" for details.

The development board has been programmed with “myir-image-full” image when it leaves the factory, so you only need to power it up and use it.

1.3. Documents

Depending on the stages of using the development board, the SDK contains different categories of documentation and manuals in addition to release notes, evaluation guides, development guides, application notes, and frequently asked questions. Please refer to Table 2-4 of “MYD-Y6ULX SDK2.0.0 Release Notes” for the detailed list of documents.

1.4. Preparation

The following sections focus on how to evaluate and test the system's hardware resources and interfaces as well as software functions. Mainly with the help of some commonly used tools and commands under Linux, as well as our own examples of the application for testing. The software evaluation guide is divided into several parts, including: core components, peripheral interfaces, network applications, multimedia applications, development support applications, system tools and so on. The following chapters will make a comprehensive explanation for each part and describe in detail the specific evaluation methods and steps of each part of resources.

2. Core Components

In Linux, the PROC virtual file system is provided to query the parameters of various core resources and some common tools are provided to evaluate the performance of resources. The following will be specific to the CPU, memory, eMMC. The parameters of RTC and other core resources are also read and tested.

2.1. CPU

The i.MX 6UltraLite is a high performance, ultra efficient processor family featuring NXP’ s advanced implementation of the single ARM Cortex®-A7 core, which operates at speeds up to 528 MHz. The i.MX 6UltraLite includes an integrated power management module that reduces the complexity of the external power supply and simplifies the power sequencing. Each processor in this family provides various memory interfaces, including LPDDR2, DDR3, DDR3L, Raw and Managed NAND flash, NOR flash, eMMC, Quad SPI, and a wide range of other interfaces for connecting peripherals, such as WLAN, Bluetooth ™,GPS,displays, and camera sensors.

1) View the CPU Information

Read the CPU provider and parameter information in the system, which can be obtained through the /proc/cpuinfo file.

```
root@myd-y6ull14x14:~# cat /proc/cpuinfo
processor : 0
model name : ARMv7 Processor rev 5 (v7l)
BogoMIPS : 24.00
Features : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd
32 lpae
CPU implementer : 0x41
CPU architecture: 7
CPU variant : 0x0
```

```
CPU part : 0xc07
CPU revision : 5

Hardware : Freescale i.MX6 Ultralite (Device Tree)
Revision : 0000
Serial : 0323b1d75986e1d8
```

- processor : The number of logical processing cores in a system can be either a physical core for multi-core processors or a virtual logical core using hyperthreading technology
- model name : The name and number of the CPU belongs to
- bogomips : A rough measure of the number of millions of instructions the CPU runs per second at kernel startup (Million Instructions Per Second)

2) View CPU Utilization

```
root@myd-y6ull14x14:~# top
top - 11:02:51 up 2:17, 1 user, load average: 0.70, 0.65, 0.64
Tasks: 71 total, 2 running, 69 sleeping, 0 stopped, 0 zombie
%Cpu(s): 1.0 us, 0.3 sy, 0.0 ni, 98.6 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 490.0 total, 318.7 free, 106.1 used, 65.1 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 364.5 avail Mem
```

- %usr: Represents the CPU utilization of the user space program (not scheduled by NICE)
- %sys: Represents the CPU utilization of system space, mainly kernel programs
- %nic: Represents the CPU usage of a program that has been scheduled through NICE in user space
- %idle: idle CPU

3) Gets CPU Temperature

A temperature sensor is built into the CPU to collect the internal temperature of the CPU, which can be easily obtained through the hardware monitoring module in Linux system.

```
root@myd-y6ull14x14:~# cat /sys/class/thermal/thermal_zone0/temp
52730
```

The number shown above is in the unit of one thousandth of degree centigrade, the result divided by 1000 is the current temperature.

4) CPU Stress Test

There are many ways to test the CPU pressure. For example, "bc" command can be used to calculate the PI, and we can test the stability of the CPU in the process of operation by this means.

```
root@myd-y6ull14x14:~# echo "scale=5000; 4*a(1)" | bc -l -q &
[1] 507
```

The above command will calculate the PI in the background and accurate to 5000 decimal places. The calculation process needs a period of time. At this time, we can check the variation of CPU utilization by the top command, as follows:

```
root@myd-y6ull14x14:~# top
top - 03:47:29 up 33 min,  1 user,  load average: 0.57, 0.16, 0.05
Tasks: 126 total,   2 running, 124 sleeping,   0 stopped,   0 zombie
%Cpu(s): 25.1 us,  0.0 sy,  0.0 ni, 74.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   1943.6 total,   1623.3 free,    234.6 used,     85.7 buff/cache
MiB Swap:      0.0 total,      0.0 free,      0.0 used.   1617.3 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME
+ COMMAND

    784 root      20   0    2728   1644   1392 R  99.7   0.1   0:42.56
```

After about 3 minutes, the result of PI was calculated. The CPU usage was very high and there were no exceptions, indicating that the CPU stress test had passed.

By increasing the accuracy requirement, the test pressure can be further improved.

```
root@myd-y6ull14x14:~# 3.14159265358979323846264338327950288419716939937510582097494459230781640628620899862803482534211706798214808651328230664709384460955058\
223172535940812848111745028410270193852110555964462294895493038196444\
288109756659334461284756482337867831652712019091456485669234603486104\
454326648213393607260249141273724587006606315588174881520920962829254\
409171536436789259036001133053054882046652138414695194151160943305724\
703657595919530921861173819326117931051185480744623799627495673518855\
**********
```

2.2. Memory

The i.MX6UL device provides an external SDRAM interface, supports external memory capacity up to 8 Gigabit (1GB), 16 or 32 bit LPDDR2 or DDR3, and operates at 400 MHz.

1) Check the Memory Information

The parameter information of memory in the system can be obtained by reading the "*/proc/meminfo*" file.

```
root@myd-y6ull14x14:~# cat /proc/meminfo
MemTotal:         501788 kB
MemFree:          326380 kB
MemAvailable:     373268 kB
Buffers:            4208 kB
Cached:            56212 kB
SwapCached:            0 kB
Active:            35820 kB
Inactive:          47996 kB
Active(anon):      23860 kB
Inactive(anon):     9268 kB
******
```

- MemTotal : All available RAM sizes, physical memory minus reserved bits and kernel usage
- MemFree : LowFree + HighFree
- Buffers : The size used to cache the block device
- Cached : The buffer size of the file
- SwapCached : Memory that has been swapped out. Associated with I/O
- Active : Frequently (recently) used memory
- Inactive : Memory that has not been used much recently

2) Get memory usage

The "free" command can be used to read memory usage, with the "-m" parameter representing Mega Byte.

```
root@myd-y6ull14x14:~# free -m
              total        used        free      shared  buff/cache   available
Mem:            490         106         318           9          65
 364
Swap:             0           0           0
```

- total : Total Memory
- used : The amount of memory used
- free : The amount of memory available

3) Memory Stress Test

The "memtester" tool under Linux system can be used to test the pressure of the existing memory of the system by giving the size and times of test memory. If the memory test is specified once and the memory under test is 300MByte, the test method is as follows.

```
root@myd-y6ull14x14:~# memtester 300M 1
memtester version 4.3.0 (64-bit)
Copyright (C) 2001-2012 Charles Cazabon.
Licensed under the GNU General Public License version 2 (only).

pagesize is 4096
pagesizemask is 0xfffffffffffff000
want 300MB (314572800 bytes)
got  300MB (314572800 bytes), trying mlock ...locked.
Loop 1/1:
  Stuck Address       : ok
  Random Value        : ok
  Compare XOR         : ok
  Compare SUB         : ok
  Compare MUL         : ok
```

```
  Compare DIV          : ok
  Compare OR           : ok
  Compare AND          : ok
  Sequential Increment: ok
  Solid Bits           : ok
  Block Sequential     : ok
  Checkerboard         : ok
  Bit Spread           : ok
  Bit Flip             : ok
  Walking Ones         : ok
  Walking Zeroes       : ok

Done.
```

2.3. Flash

eMMC is a communication and mass data storage device, including a Multi-Media Card (MMC) interface, an NAND Flash component, and a controller on an advanced signal bus, which is compliant with the MMC system specification. Its cost, small size, Flash technology independence, and high data throughput make it an ideal choice for embedded applications. The i.MX6UL device supports 8-bit SDMMC interface (EMMC v4.51), with a maximum support of 128GB. This section will explain the steps and methods of viewing and operating EMMC under Linux system.

1) Emmc

- View eMMC Capacity

The eMMC partition information and capacity can be queried through the "fdisk -l" command as follows.

```
root@myd-y6ull14x14:~# fdisk -l
Disk /dev/mmcblk1: 3672 MB, 3850371072 bytes, 7520256 sectors
117504 cylinders, 4 heads, 16 sectors/track
Units: sectors of 1 * 512 = 512 bytes

Device          Boot StartCHS    EndCHS        StartLBA     EndLBA    Sectors
  Size Id Type
/dev/mmcblk1p1      320,0,1      895,3,16         20480     122879      10240
0 50.0M  c Win95 FAT32 (LBA)
/dev/mmcblk1p2      896,0,1      767,3,16        122880    7520255     739737
6 3612M 83 Linux
```

  - /dev/mmcblk2p1 : Used for bootloader
  - /dev/mmcblk2p2 : Used to store the root file system

- View the eMMC Partition Information

By "df" command, you can query eMMC partition information, usage, mount directory and other information.

```
root@myd-y6ull14x14:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       487M  328M  131M  72% /
devtmpfs        181M  4.0K  181M   1% /dev
tmpfs           246M     0  246M   0% /dev/shm
tmpfs           246M  8.8M  237M   4% /run
tmpfs           246M     0  246M   0% /sys/fs/cgroup
tmpfs           246M     0  246M   0% /tmp
tmpfs           246M  136K  245M   1% /var/volatile
/dev/mmcblk1p1   25M  9.0M   16M  37% /run/media/mmcblk1p1
/dev/mmcblk1p2  149M   86M   53M  63% /run/media/mmcblk1p2
```

- /dev/root : Root file system, mounted to the root directory
- tmpfs : Memory virtual file system, mounted to a different directory
- devtmpfs : Used to create dev for the system

2) Nand flash

●Check code partition definition

The partition structure of NAND is defined in the uboot, which passes the parameters to the kernel after boot.Are defined as follows:

```
#define MFG_NAND_PARTITION "mtdparts=gpmi-nand:5m(boot),1m(env),10m(kernel),1m(dtb),-(rootfs) "
#else
```

●View the partition situation after system boot

After entering the Linux system, you can view the actual partition situation:

```
root@myd-y6ull14x14:~# cat /proc/mtd
dev:    size   erasesize  name
mtd0: 00500000 00020000 "boot"
mtd1: 00100000 00020000 "env"
```

```
mtd2: 00a00000 00020000 "kernel"
mtd3: 00100000 00020000 "dtb"
mtd4: 0ef00000 00020000 "rootfs"
```

- view partition parameters

```
root@myd-y6ull14x14:~#cat /proc/cmdline
console=ttymxc0,115200 ubi.mtd=4 root=ubi0:rootfs rootfstype=ubifs mtdparts
=gpmi-nand:5m(boot),1m(env),10m(kernel),1m(dtb),-(rootfs)
```

- ubi.mtd : the partition of the file system
- rootfstype : the format of the file system
- root : the format of the file system
- Console : debug serial port parameters

2.4. RTC

RTC (Real-time Clock) is a clock itself, which is used to record the Real time. When the software system is shut down, the system time is retained and the timing is continued. After the system is turned back on, the time is synchronized into the software system.

i.MX6UL chip contains a RTC clock. If the actual product does not have a very high requirement for RTC power consumption, and the power off time should be kept within one month, the RTC inside the CPU can be directly used; otherwise, special external RTC chip will be needed. The test of RTC is usually carried out with the common Linux system "hwclock" and "date" commands. The following test writes the system time to the RTC, reads the RTC time and sets it as the system time, and check if the RTC time keeps running during the development board is power off.

- **View RTC Devices in Linux System**

```
root@myd-y6ull14x14:~# ls -l /dev/rtc*
lrwxrwxrwx 1 root root       4 Sep  5 03:14 /dev/rtc -> rtc0
crw------- 1 root root 251, 0 Sep  5 03:14 /dev/rtc0
crw------- 1 root root 251, 1 Sep  5 03:14 /dev/rtc1
```

"/dev/rtc0" is the default RTC device driver node for applications.

- **Set System Time**

Set the system time to Tue Sep 8 09:07:30 UTC 2020 as follows.

```
root@myd-y6ull14x14:~# date 090809072020.30
Tue Sep  8 09:07:30 UTC 2020
```

- **Write System Time into RTC**

Writes the system time set by the previous date command to the RTC device.

```
root@myd-y6ull14x14:~# hwclock -w
```

- **Read Data from RTC and Set to System Time**

```
root@myd-y6ull14x14:~# hwclock -r
2020-09-08 09:08:02.588845+00:00
```

- **Keep RTC Time When Power is Off**

Turn off the power and restart the device after about 2 minutes. Check the RTC time and system time again.

```
root@myd-y6ull14x14:~# hwclock -r
2020-09-08 09:10:50.206047+00:00
```

The above information indicates that the RTC time and system time viewed after rebooting have increased by about 2 minutes compared with the previous setting, indicating that the RTC is working normally. If the precision of the RTC needs to be tested in detail, the outage time can be extended by such as 24 hours to test the difference between the RTC time and the standard time.

- **Make the System Time and the RTC Time Synchronization**

```
root@myd-y6ull14x14:~# hwclock -s
root@myd-y6ull14x14:~# date
Tue Sep  8 09:11:07 UTC 2020
```

If you add the "hwclock -s" command to the startup script of Linux System, you can ensure that system time and RTC time are synchronized with each startup.

2.5. Watchdog

The Linux kernel includes the Watchdog subsystem. In the hardware design process, the watchdog timer inside the chip or an external watchdog chip can generally be used to implement the Watchdog function to monitor the operation of the system. When the system fails to feed the dog in an abnormal situation, the system can automatically reset There is a watchdog inside the MYD-Y6ULX chip. This section demonstrates the use of the watchdog, testing the opening and closing of the watchdog, and setting the watchdog timeout time.

1) Stop the Watchdog

The watchdog can be closed by writing the "V" character to the watchdog device by command.

```
root@myd-y6ull14x14:~# echo V > /dev/watchdog0
```

From the above information, the system cannot turn off the watchdog due to the nowayout attribute.

2) Using the application to test the watchdog

- **Set the Timeout Settings of the Watchdog**

To implement the timeout through IOCTL command:WDIOC_SETTIMEOUT, and you also need a parameter, that is:timeout. Examples of use:

```
ioctl(fd, WDIOC_SETTIMEOUT, &timeout);
```

The above is the reference code for setting the current timeout of the watchdog. “fd” is the file handle of the watchdog device.

- **Demo Application of the Watchdog**

Compile the production execution file(watchdog) and copy it to the development board. Execute the following command:

```
root@myir:~# watchdog_test
Usage: wdt_driver_test <timeout> <sleep> <test>
    timeout: value in seconds to cause wdt timeout/reset
```

```
    sleep: value in seconds to service the wdt
    test: 0 - Service wdt with ioctl(), 1 - with write()
```

Run the watchdog app for a timeout of 4s and feed the dog every 1s:

```
root@myd-y6ull14x14:~# watchdog_test  4 1 0
Starting wdt_driver (timeout: 4, sleep: 1, test: ioctl)
Trying to set timeout value=4 seconds
The actual timeout was set to 4 seconds
Now reading back -- The timeout is 4 seconds
```

If the above 1s is changed to greater than 4s, the development board will restart after the required 4s feeding time.For more information on ST watchdog peripherals, please refer to the following links:

2.6. Power Manager

This section demonstrates the Suspend functionality of Linux power management, allowing development boards to sleep and wake up via external events.

The Linux kernel generally provides three types of suspend: Freeze, Standby, and STR(Suspend to RAM), which can be triggered by writing "freeze", "standby", and "mem" to the "*/sys/power/state*" files in user space.

1) View the Mode Supported by the Current Development Board

```
root@myd-y6ull14x14:~# cat /sys/power/state
freeze standby mem
```

2) Enter mem Power Management Mode

Write a string to */sys/power/state* in user space to enter the corresponding power management mode. The sleep mode is as follows:

```
root@myd-y6ull14x14:~# echo "mem" > /sys/power/state
[  278.716505] PM: suspend entry (deep)
[  278.720724] Filesystems sync: 0.000 seconds
[  278.741713] Freezing user space processes ... (elapsed 0.001 seconds) done.
[  278.750794] OOM killer disabled.
[  278.754120] Freezing remaining freezable tasks ... (elapsed 0.001 seconds) done.
[  278.762886] printk: Suspending console(s) (use no_console_suspend to debug)
```

3) Wake up through KEY BUTTON

Take “mem” mode as an example to test. the debugging port cannot input again, and the heartbeat lamp stops flashing.At this time, Press button K3 on the development board to wake up the system, as follows:

```
[  278.872728] fec 20b4000.ethernet eth0: Link is Down
[  278.874269] PM: suspend devices took 0.100 seconds
```

```
[  278.878646] Disabling non-boot CPUs ...
[  278.878658] Turn off M/F mix!
[  278.882566] imx-sdma 20ec000.sdma: loaded firmware 3.5
[  279.205157] usb 1-1: reset high-speed USB device number 2 using ci_hdrc
[  279.526926] PM: resume devices took 0.650 seconds
[  279.560349] OOM killer enabled.
[  279.563503] Restarting tasks ... done.
[  279.630415] PM: suspend exit
root@myd-y6ull14x14:~#
```

At this point, the debugging port can input again, and the heartbeat lamp starts flashing.

3. Peripheral Interface

3.1. GPIO

The GPIO pins of MYD-Y6ULX are defined in the form of GPIOX_Y. For the mapping relationship between the pin label name and the form of GPIOX_Y, please refer to the "MYC-Y6ULX Pin list_V13.xlsx" manual. The formula for converting GPIOX_Y into pin number is:

```
(X-1) *32+Y
```

1) Export GPIO to User Space

```
root@myd-y6ull14x14:~#/sys# echo 24 > /sys/class/gpio/export
```

After the successful exporting to user space, the directory gpio23 will be generated under the directory */sys/class/gpio/*.

2) Set/View the GPIO Direction

- **Set as Input**

```
root@myd-y6ull14x14:~# echo "in" > /sys/class/gpio/gpio24/direction
```

- **Set as Output**

```
root@myd-y6ull14x14:~# echo "out" > /sys/class/gpio/gpio24/direction
```

- **View the Direction of GPIO**

```
root@myd-y6ull14x14:~# cat /sys/class/gpio/gpio24/direction
out
```

Return “in” for input and “out” for output.

3) Set/View the GPIO Value

- **Set Output Low:**

```
root@myd-y6ull14x14:~# echo "0" > /sys/class/gpio/gpio24/value
```

- **Set Output High:**

```
root@myd-y6ull14x14:~# echo "1" > /sys/class/gpio/gpio24/value
```

- **View the GPIO Value:**

```
root@myir:~# cat /sys/class/gpio/gpio24/value
1
```

3.2. LED Lamp

The Linux system provides a separate subsystem to facilitate the operation of LED devices from user space. The subsystem provides an interface for LED devices in the form of files. These interfaces are located in the */sys/class/leds* directory. In the list of hardware resources, we have listed all the LEDs on the device. Let's test a LED by reading and writing sysfs from the command. The following commands are generic commands and are general methods of controlling LEDs .

1) View the LEDs

The directory where leds are operated is */sys/class/leds*. The contents of the directory are as follows:

```
root@myd-y6ull14x14:~# ls  /sys/class/leds/
cpu  mmc0::  mmc1::
```

2) Test a LED

Take red lights for example.

- **Get the Status of the LED**

Where 0 means the LED is on and 1 means the LED is off.

```
root@myd-y6ull14x14:~# cat  /sys/class/leds/cpu/brightness
255
```

The above results show that the current red light LED is on.

- **Turn off the LED**

```
root@myd-y6ull14x14:~# echo 0 > /sys/class/leds/cpu/brightness
```

- **Turn on the LED**

```
root@myd-y6ull14x14:~# echo 1 > /sys/class/leds/cpu/brightness
```

- **Trigger the LED**

After starting the "timer" trigger mode, the LED flashes at a default cycle of 1Hz with a duty ratio of 50%.

```
root@myd-y6ull14x14:~# echo "heartbeat" > /sys/class/leds/cpu/trigger
```

3.3. RS232

This section uses the Linux API to configure the transceiver of RS232 interface on the development board. The Linux serial device driver node are generally named as */dev/*ttymxcn(n=0,1,2,3...) . N represents the serial port number in the system, "ttymxc" is the serial port device name defined by the kernel. In this section, J10 interface (namely UART2) on MYD-Y6ULX base board is taken as an example for testing. The numbered device node of UART2 is ttymxc1. Please follow the table below to prepare before testing:

Table3-1. RS232 Interface Configuration

	MYD-Y6ULX	
/dev/ttymxc0	Uart1	
/dev/ttymxc1	Uart2	
/dev/ttymxc2	Uart3	

Connect the USB-RS232 converter to the USB Host port of the PC and connect the RXD and TXD of J10 to the TXD and RXD of the USB-RS232 converter, respectively.

- **Send Data from Windows PC with SSCOM**

  Use the debug serial port tool to send data “1234567890” under windows and check the receiving area.

- **Development board test receiving and sending data**

```
root@myd-y6ull14x14:~# uart_test -d /dev/ttymxc1 -b 115200
/dev/ttymxc1 RECV 10 total
/dev/ttymxc1 RECV: 1234567890
/dev/ttymxc1 RECV 10 total
/dev/ttymxc1 RECV: 1234567890
/dev/ttymxc1 RECV 10 total
/dev/ttymxc1 RECV: 1234567890
/dev/ttymxc1 RECV 10 total
/dev/ttymxc1 RECV: 1234567890
```

3.4. RS485

This routine demonstrates how to use the Linux API to test the development board's data sending and receiving functions with RS485 device alias Serial3,so the device node is ttymxc3. Take RS485 of J10 interface as an example for testing.

Prepare two development boards, connect the 485 interface, connect 485A and 485B of one board to 485A and 485B of the other board.

- **A development board is used as the sending end, and the running program sends data:**

```
# rs485_write -d /dev/ttymxc3 -b 4800 -e 1
SEND[20]: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0x0c 0
x0d 0x0e 0x0f 0x10 0x11 0x12 0x13 0x14
SEND[20]: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0x0c 0
x0d 0x0e 0x0f 0x10 0x11 0x12 0x13 0x14
SEND[20]: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0x0c 0
x0d 0x0e 0x0f 0x10 0x11 0x12 0x13 0x14
SEND[20]: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0x0c 0
x0d 0x0e 0x0f 0x10 0x11 0x12 0x13 0x14
```

- **The other board is the receiving end, and run the program to receive data:**

```
# rs485_read -d /dev/ttymxc3 -b 4800 -e 1
RECV[20]: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0x0c 0x
0d 0x0e 0x0f 0x10 0x11 0x12 0x13 0x14
RECV[20]: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0x0c 0x
0d 0x0e 0x0f 0x10 0x11 0x12 0x13 0x14
RECV[20]: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0x0c 0x
0d 0x0e 0x0f 0x10 0x11 0x12 0x13 0x14
RECV[20]: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0x0c 0x
0d 0x0e 0x0f 0x10 0x11 0x12 0x13 0x14
```

3.5. CAN

This section uses “cansend” and “candump” commands in Linux system for SocketCAN communication test. The test here uses two development boards for docking tests.

The CANH and CANL pins of the J10 are connected to CANH and CANL of the same type.

1) Initializes the CAN network interface

- **Set the CAN Baud Rate**

Using CAN requires setting the baud rate and opening the CAN network interface. Refer to the following command to set the arbitration baud rate of two boards to 1Mbps and the data Baud rate device to 4Mbps respectively, and turn on the CAN function.

```
root@myd-y6ull14x14:~# ip link set can0 type can bitrate 50000 triple-sampli
ng on
root@myd-y6ull14x14:~# ifconfig can0 up
```

At this point, you can use the CAN function.

2) Send and Receive Data

- **Send Data**

Set one of the boards to send and use “cansend” to send a 8-byte fixed format string to test:

```
root@myd-y6ull14x14:~# cansend can0 100#01.02.03.04.05.06.07.08
```

- **Receive Data**

Set another board to receive. Candump CAN be used to view CAN's receiving data:

```
root@myd-y6ull14x14:~#  candump can0
root@myd-y6ull14x14:~#  can0 100 [8] 01 02 03 04 05 06 07 08
```

3) Statistics of can0 information

After receiving and sending CAN data, it displays the details and statistics of CAN device. The value of "clock" stands for CAN clock, "drop" stands for lost packets, and "overrun" stands for bus overrun, "error" stands for bus error.

```
root@myd-y6ull14x14:~#  ip -details -statistics link show can0
2: can0: <NOARP,ECHO> mtu 16 qdisc noop state DOWN mode DEFAULT gr
oup default qlen 10
    link/can  promiscuity 0 minmtu 0 maxmtu 0
    can state STOPPED (berr-counter tx 0 rx 0) restart-ms 0
          flexcan: tseg1 4..16 tseg2 2..8 sjw 1..4 brp 1..256 brp-inc 1
          clock 30000000
          re-started bus-errors arbit-lost error-warn error-pass bus-off
          0          0          0          0          0          0          num
txqueues 1 numrxqueues 1 gso_max_size 65536 gso_max_segs 65535
    RX: bytes  packets  errors  dropped overrun mcast
    0          0        0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    0          0        0       0       0       0
```

3.6. Key

The "*/dev/input/eventxx*" devices in Linux system can be used to easily debug input devices such as mice, keyboards, touchpads, etc. This section focuses on testing the key. We can use the "hexdump" command and the "dmesg" command to see if the keys work normally. MYD-Y6ULX has three buttons, and S3 is the system reset button. K3 is the wake-up button of wkup, which is not configured in A7 device tree. S1 is the user key that has been configured in the device tree.

1) Device Tree Node

Open the supporting device tree file myb-imx6ul-14x14.dtsi, and you can see the corresponding GPIO-Keys node:

```
        gpio-keys {
```

```
            compatible = "gpio-keys";
            pinctrl-names = "default";
            pinctrl-0 = <&pinctrl_gpio_key>;
            user {
                    label = "User Button";
                    gpios = <&gpio5 0 GPIO_ACTIVE_HIGH>;
                    gpio-key,wakeup;
                    linux,code = <KEY_1>;
            };
    };
```

2) Test Keys

- **View Information of the Input Device Event**

```
root@myd-y6ull14x14:~# evtest
No device specified, trying to scan all of /dev/input/event*
Available devices:
/dev/input/event0:    20cc000.snvs:snvs-powerkey
/dev/input/event1:    generic ft5x06 (00)
/dev/input/event2:    iMX6UL Touchscreen Controller
/dev/input/event3:    gpio-keys
```

The above results show that the corresponding device event for GPIO-Keys is “event2”.

- **Dump Information of event0**

Execute the following command and press K3. The serial port terminal will print out the following information:

```
Select the device event number [0-3]: 3
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 2 (KEY_1)
Properties:
Testing ... (interrupt to exit)
Event: time 1599641292.878878, type 1 (EV_KEY), code 2 (KEY_1), value 0
Event: time 1599641292.878878, ---------- SYN_REPORT ------------
Event: time 1599641293.049011, type 1 (EV_KEY), code 2 (KEY_1), value 1
Event: time 1599641293.049011, ---------- SYN_REPORT ------------
Event: time 1599641293.718856, type 1 (EV_KEY), code 2 (KEY_1), value 0
Event: time 1599641293.718856, ---------- SYN_REPORT ------------
Event: time 1599641293.858903, type 1 (EV_KEY), code 2 (KEY_1), value 1
Event: time 1599641293.858903, ---------- SYN_REPORT ------------
Event: time 1599641294.198953, type 1 (EV_KEY), code 2 (KEY_1), value 0
Event: time 1599641294.198953, ---------- SYN_REPORT ------------
Event: time 1599641294.308852, type 1 (EV_KEY), code 2 (KEY_1), value 1
Event: time 1599641294.308852, ---------- SYN_REPORT ------------
Event: time 1599641294.558876, type 1 (EV_KEY), code 2 (KEY_1), value 0
Event: time 1599641294.558876, ---------- SYN_REPORT ------------
Event: time 1599641294.678870, type 1 (EV_KEY), code 2 (KEY_1), value 1
Event: time 1599641294.678870, ---------- SYN_REPORT ------------
Event: time 1599641294.898855, type 1 (EV_KEY), code 2 (KEY_1), value 0
Event: time 1599641294.898855, ---------- SYN_REPORT ------------
Event: time 1599641295.038987, type 1 (EV_KEY), code 2 (KEY_1), value 1
Event: time 1599641295.038987, ---------- SYN_REPORT ------------
```

Each time K3 is pressed, the current terminal will print out the current event code value, that is, the key works normally.

3.7. USB

MYD-Y6ULX has 2 USB2.0 ports, one is used for OTG to download the mirror image, and the other USB2.0 port becomes 2 USB ports through the expansion chip. This section verifies the feasibility of the USB Host driver through related commands or hot plug and USB HUB, and realizes the function of reading and writing U disk and the function of USB enumeration.

1) Check the Kernel Message of USB

- **View the USB Device Information**

Connect the U disk to the USB Host interface of the development board, and the kernel log is as follows:

```
root@myd-y6ull14x14:~# [  755.254972] usb 1-1.1: new high-speed USB devic
e number 3 using ci_hdrc
[  755.413421] usb 1-1.1: New USB device found, idVendor=058f, idProduct=6
387, bcdDevice= 1.06
[  755.422370] usb 1-1.1: New USB device strings: Mfr=1, Product=2, SerialNu
mber=3
[  755.437531] usb 1-1.1: Product: Mass Storage
[  755.441955] usb 1-1.1: Manufacturer: Generic
[  755.451900] usb 1-1.1: SerialNumber: 03F004F7
[  755.468433] usb-storage 1-1.1:1.0: USB Mass Storage device detected
[  755.493065] scsi host0: usb-storage 1-1.1:1.0
[  756.567834] scsi 0:0:0:0: Direct-Access     Generic  Flash Disk       8.07 P
Q: 0 ANSI: 4
[  756.594293] sd 0:0:0:0: [sda] 31129600 512-byte logical blocks: (15.9 GB/14.
8 GiB)
[  756.616305] sd 0:0:0:0: [sda] Write Protect is off
[  756.634066] sd 0:0:0:0: [sda] Write cache: disabled, read cache: enabled, do
esn't support DPO or FUA
[  756.694383]  sda:
[  756.711553] sd 0:0:0:0: [sda] Attached SCSI removable disk
```

```
[  757.721016] FAT-fs (sda): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.
```

From the above information it can be concluded that the device to be assigned is sda.

2) Mount and Read/Write the USB Flash Disk

- **Mount**

The file system has been added with the automatic mounting function, no manual mounting is required.

```
root@myd-y6ull14x14:~# ls /run/media/
```

- **Write USB Flash Disk**

```
root@myd-y6ull14x14:~# echo "myir udisk test" > /run/media/sda/test_file.txt
```

- **Read USB Flash Disk**

You need to create a test.txt file on your USB drive in advance.

```
root@myd-y6ull14x14:~# cat /run/media/sda/test_file.txt
myir udisk test
```

After the file is written, the sync command needs to be executed to ensure that the data is fully written to the USB disk before the device can be unmounted.

3.8. Backlight

This routine is mainly to test the brightness of the backlight. It can query and adjust the brightness of the backlight by operating the corresponding file of the backlight driver in "*/sys/class/backlight*" directory.

- **The SYSFS Directory of Backlight Driver**

```
root@myd-y6ull14x14:~# cd /sys/class/backlight/backlight-display/
root@myd-y6ull14x14:/sys/class/backlight/backlight-display# ls
actual_brightness brightness device power subsystem type
bl_power consumers max_brightness scale suppliers uevent
```

- **View Brightness Value of Backlight**

```
root@myd-y6ull14x14:/sys/class/backlight/backlight-display# cat brightness
6
```

- **Check the Maximum Level of Brightness**

```
root@myd-y6ull14x14:/sys/class/backlight/backlight-display# cat max_brightnes
s
7
```

- **Change brightness**

If you need to adjust the backlight level, use echo to write the corresponding brightness level to brightness, but it cannot exceed the maximum level.

```
root@myd-y6ull14x14:/sys/class/backlight/backlight-display# echo 10 > brightn
ess
sh: write error: Invalid argument
```

3.9. Touch Panel

The MYD-Y6ULX series development board supports capacitive touch and resistive touch. The MYiR capacitive touch IC is FT5x16, and the resistive touch uses the internal TSC module of the i.MX6UL chip to read ADC data. Users can purchase accessories by themselves according to actual needs. The following is a simple test through the touch function of the evtest command screen.

1) Calibration

Capacitive screen does not need calibration, resistive screen needs calibration.

```
root@myd-y6ull14x14:~# ts_calibrate
xres = 800, yres = 480
Took 7 samples...
Top left : X = 3174 Y =  509
Took 5 samples...
Top right : X = 3120 Y = 3715
Took 4 samples...
Bot right : X =  843 Y = 3686
Took 7 samples...
Bot left : X =  855 Y =  483
Took 1 samples...
Center : X = 1992 Y = 2104
-53.327515 -0.002611 0.218416
573.700073 -0.165326 -0.001703
Calibration constants: -3494872 -171 14314 37598008 -10834 -111 65536
root@myd-y6ull14x14:~#
```

2) Touch test with evtest command

The terminal executes "evtest" to enter the test interface. Select the test peripheral as the touch screen, where the default is input interrupt 0. Select "0" in the test interface and press enter to start the test:

```
root@myd-y6ull14x14:~# 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 ft5x06 (00)
/dev/input/event2:      gpio-keys
/dev/input/event3:      bd718xx-pwrkey
Select the device event number [0-3]: 1
Input driver version is 1.0.1
Input device ID: bus 0x18 vendor 0x0 product 0x0 version 0x0
Input device name: "generic ft5x06 (00)"
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)
    --snip--
    Testing ... (interrupt to exit)
    Event: time 1599614278.142778, type 3 (EV_ABS), code 57 (ABS_MT_TRAC
KING_ID), value 7
    Event: time 1599614278.142778, type 3 (EV_ABS), code 53 (ABS_MT_POSITI
ON_X), value 419
    Event: time 1599614278.142778, type 3 (EV_ABS), code 54 (ABS_MT_POSITI
ON_Y), value 367
    Event: time 1599614278.142778, type 1 (EV_KEY), code 330 (BTN_TOUCH),
 value 1
    Event: time 1599614278.142778, type 3 (EV_ABS), code 0 (ABS_X), value 4
19
```

```
    Event: time 1599614278.142778, type 3 (EV_ABS), code 1 (ABS_Y), value 3
67
    Event: time 1599614278.142778, ---------- SYN_REPORT ------------
    Event: time 1599614278.195100, type 3 (EV_ABS), code 53 (ABS_MT_POSITI
ON_X), value 418
    Event: time 1599614278.195100, type 3 (EV_ABS), code 54 (ABS_MT_POSITI
ON_Y), value 365
    Event: time 1599614278.195100, type 3 (EV_ABS), code 0 (ABS_X), value 4
18
    Event: time 1599614278.195100, type 3 (EV_ABS), code 1 (ABS_Y), value 3
65
    Event: time 1599614278.195100, ---------- SYN_REPORT ------------
    Event: time 1599614278.272333, type 3 (EV_ABS), code 57 (ABS_MT_TRAC
KING_ID), value -1
    Event: time 1599614278.272333, type 1 (EV_KEY), code 330 (BTN_TOUCH),
 value 0
    Event: time 1599614278.272333, ---------- SYN_REPORT ------------
. . .
```

It can be seen that after the touch screen is connected, it will be recognized as an input interface, and the event number of the touch screen is 0.

3) Print test information

Click the touch screen, and the terminal will print the corresponding information.For example,they will output the event information reported by the touch driver to console as follows:

```
    Event: time 1599614278.142778, type 3 (EV_ABS), code 1 (ABS_Y), value 3
67
    Event: time 1599614278.142778, ---------- SYN_REPORT ------------
    Event: time 1599614278.195100, type 3 (EV_ABS), code 53 (ABS_MT_POSITI
ON_X), value 418
```

```
    Event: time 1599614278.195100, type 3 (EV_ABS), code 54 (ABS_MT_POSITI
ON_Y), value 365
    Event: time 1599614278.195100, type 3 (EV_ABS), code 0 (ABS_X), value 4
18
    Event: time 1599614278.195100, type 3 (EV_ABS), code 1 (ABS_Y), value 3
65
    Event: time 1599614278.195100, ---------- SYN_REPORT ------------
    Event: time 1599614278.272333, type 3 (EV_ABS), code 57 (ABS_MT_TRAC
KING_ID), value -1
    Event: time 1599614278.272333, type 1 (EV_KEY), code 330 (BTN_TOUCH),
 value 0
Event: time 1599614278.272333, ---------- SYN_REPORT ------------
```

From the above information, it mainly displays coordinate values and key values,the specific information is as follows:

- EV_SYN: Synchronous events
- EV_KEY: Key event, such as BTN_ Touch means touch button
- EV_ABS: Absolute coordinates, such as those reported by touch screen
- BTN_TOUCH: Touch button
- ABS_MT_TRACKING_ID:Represents the beginning of the collection of information. And then another ABS_MT_TRACKING_ID indicates the end of the collection

A single touch message is carried in ABS and sent in certain order, such as:

- ABS_X:X absolute coordinates relative to the screen
- ABS_Y:Y absolute coordinates relative to the screen

But multi-touch information is carried in ABS_MT and sent in a certain, such as:

- ABS_MT_POSITION_X: Represents the center point x coordinate position of the screen contact surface
- ABS_MT_POSITION_Y: coordinate position representing the center point Y screen contact

3.10. Display

This example demonstrates the operation of the Linux FrameBuffer device to realize the LCD output display RGB color and color synthesis test. The routine is developed based on the Linux FrameBuffer API interface. The LCD needs to be connected to the J3 interface before testing.

MYiR Technology provides three LCD modules, and the Linux software default configuration is MY-TFT070CV2.

Table 3-2 Optional Modules

model	Description
MY-TFT070CV2	7-inch Capacitive
MY-TFT070RV2	7-inch resistive
MY-TFT043RV2	4.3-inch resistive

1) TEST LCD

```
# framebuffer_test
The framebuffer device was opened successfully.
vinfo.xres=480
vinfo.yres=272
vinfo.bits_per_bits=16
vinfo.xoffset=0
vinfo.yoffset=0
red.offset=11
green.offset=5
blue.offset=0
transp.offset=0
finfo.line_length=960
finfo.type = PACKED_PIXELS
The framebuffer device was mapped to memory successfully.
color: red rgb_val: 0000F800
color: green rgb_val: 000007E0
color: blue rgb_val: 0000001F
```

```
color: r & g rgb_val: 0000FFE0
color: g & b rgb_val: 000007FF
color: r & b rgb_val: 0000F81F
color: white rgb_val: 0000FFFF
color: black rgb_val: 00000000
```

2) Modify to support 4.3-inch LCD

DTS: arch/arm/boot/dts/myb-imx6ul-14x14.dtsi

```
        display0: display@0 {
                bits-per-pixel = <16>;
                bus-width = <16>;
                display-timings {
                        native-mode = <&timing1>;
                        /*4.3-inch*/
                        timing0: timing0 {
                                clock-frequency = <9200000>;
                                hactive = <480>;
                                vactive = <272>;
                                hfront-porch = <8>;
                                hback-porch = <4>;
                                hsync-len = <41>;
                                vback-porch = <2>;
                                vfront-porch = <4>;
                                vsync-len = <10>;
                                hsync-active = <0>;
                                vsync-active = <0>;
                                de-active = <1>;
                                pixelclk-active = <0>;
                        };
                        /*7.0-inch*/
                        timing1: timing1 {
                                clock-frequency = <33000000>;
                                hactive = <800>;
                                vactive = <480>;
```

```
                    hfront-porch = <210>;
                    hback-porch = <46>;
                    hsync-len = <1>;
                    vback-porch = <22>;
                    vfront-porch = <23>;
                    vsync-len = <20>;
                    hsync-active = <0>;
                    vsync-active = <0>;
                    de-active = <1>;
                    pixelclk-active = <0>;
            };
        };
```

For example, the above code native-mode = <&timing1>, is the default parameter, the 7-inch nominated parameter is selected, and it is changed to native-mode = <&timing0>; that is, the 4.3-inch timing parameter is selected, and then restarted Compile dtb and download to the board.

4. Network Interface

The MYD_Y6ULX development board includes a 10/100M Ethernet interface and a WIFI/Bluetooth Combo Module(WM-N-BM-02). The configuration of these two network devices is described below.

4.1. Ethernet

Under Linux, there are many tools for network configuration, such as net-tools, Iproute2, Systemd-Networkd, Network Manager and Connman, etc. All of these can be selected according to actual needs during system customization. Here, several commonly used methods of Ethernet manual temporary configuration and automatic permanent configuration are introduced.

The factory image of the development board is configured with a static IP, t he configuration file is as follows:

```
root@myd-y6ull14x14:~# ls -l /etc/systemd/network/
total 8
-rwxr-xr-x 1 root root 88 Sep 9 09:05 10-static-eth0.network
-rwxr-xr-x 1 root root 88 Sep 9 09:05 11-static-eth1.network
lrwxrwxrwx 1 root root 9 Sep 9 09:11 99-default.link -> /dev/null
```

1) Configure Ethernet IP addresses Manually and Temporarily

- **Use ifconfig to Configure the Network Manually**

Firstly, check the network device information through ifconfig command as follows:

```
root@myd-y6ull14x14:~# ifconfig -a
eth0 Link encap:Ethernet HWaddr fa:2e:32:6f:4b:55
inet addr:192.168.30.202 Bcast:192.168.30.255 Mask:255.255.255.0
inet6 addr: fe80::f82e:32ff:fe6f:4b55/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
```

```
          RX packets:12795 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3896 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1217512 (1.1 MiB)  TX bytes:177644 (173.4 KiB)

eth1      Link encap:Ethernet  HWaddr da:0e:6a:e6:8a:a6
          inet addr:192.168.40.202  Bcast:192.168.40.255  Mask:255.255.255.0
          inet6 addr: fe80::d80e:6aff:fee6:8aa6/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:46 errors:0 dropped:0 overruns:0 frame:0
          TX packets:49 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3637 (3.5 KiB)  TX bytes:8216 (8.0 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:2596 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2596 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:232480 (227.0 KiB)  TX bytes:232480 (227.0 KiB)
```

Eth0 is the actual Ethernet device and defaults to a random hardware MAC a ddress. If the user needs to write the MAC address himself,

Eth0 is given the following command to manually configure the IP address 192.168.0.100:

```
# ifconfig eth0 192.168.0.100 netmask 255.255.255.0 up
```

The command above manually configured eth0 with the IP address of 192.168.0.100, the subnet mask of 255.255.255.0, and the default configured broadcast address of 192.168.0.255, and activated with the up parameter, as shown below:

```
# ifconfig eth0
```

```
eth0      Link encap:Ethernet  HWaddr FC:69:47:33:A5:65
          inet addr:192.168.0.100  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  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:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:61 Base address:0xc000
```

- **Configure the Network Manually by iprout2**

The ifconfig command that sets the IP address manually can also be substituted using "ip addr" and "ip link", for more information about iprout2, please refre to https://wiki.linuxfoundation.org/networking/iproute2 for detail.

```
# ip addr flush dev eth0
# ip addr add 192.168.0.101/24 brd + dev eth0
# ip link set eth0 up
```

If an IP address has been previously configured, the IP address configured using "ip addr add" will become the secondary address, so use "ip addr flush" to clear the previous address before configuring and activating it. Once configured, view the eth0 information from the IP Addr Show command as follows:

```
root@myir:~# ip addr show eth0
3: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state
DOWN group default qlen 1000
    link/ether fc:69:47:33:a5:65 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.101/24 brd 192.168.1.255 scope global eth0
       valid_lft forever preferred_lft forever
```

1) Configure Ethernet Automatically and Permanently

IP addresses configured through the "ifconfig" and "ip" commands are lost when power is lost, and if you need to make IP addresses permanent, you need to modify the corresponding configuration files of the network management tool.

- **Configure IP Addresses by ifupdown Dynamically**

The /etc/network/ Interfaces file for dynamically obtaining the IP address using the ifupdown or busybox configuration is shown below:

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp
        pre-up /etc/network/nfs_check
        wait-delay 15
```

By default, eth0 will request the dynamic IP address from the DHCP server in the LAN and configure the IP address, gateway and DNS each time the machine is booted up.

- **Configure IP Addresses by systemd-networkd Dynamically**

MYD-Y6ULX default factory image adopts systemd-networkd for network configuration, and for details, you can check the configuration file 80-Wired.net Work in /lib/systemd/network/ directory.

```
root@myir:~# /lib/systemd/network# cat 80-wired.network
[Match]
Name=en* eth*
KernelCommandLine=!nfsroot

[Network]
DHCP=yes

[DHCP]
RouteMetric=10
ClientIdentifier=mac
```

The configuration file above will configure the network matching en* and eth*. If the kernel boot parameter does not contain the nfsroot parameter, then the

DHCP will automatically configure the IP address, gateway, DNS and other information for the matched network card. See the following connection for detailed configuration parameters:
https://www.freedesktop.org/software/systemd/man/systemd.network.html。

- **Configure Static IP Addresses by systemd-networkd Automatically and Permanently**

If you want to configure a permanent IP address for eth0, you can write another 10-static-eth0.network file to place in /etc/systemd/network/ directory. The 10-static-eth0.network contents are as follows:

```
[Match]
Name=eth0
[Network]
Address=192.168.30.200/24
Netmask=255.255.255.2
Gateway=192.168.30.1
```

After configuring, restart systemd-networkd.service and you will see that the eth0 network address has been configured to 192.168.30.100, as shown below:

```
root@myd-y6ull14x14:~# systemctl restart systemd-networkd.service
root@myd-y6ull14x14:~# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 26:13:52:CE:15:40
          inet addr:192.168.30.200  Bcast:192.168.30.255  Mask:255.255.255.0
          inet6 addr: fe80::2413:52ff:fece:1540/64 Scope:Link
          inet6 addr: fd20:da22:fcca:1b00:2413:52ff:fece:1540/64 Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:238239 errors:0 dropped:752 overruns:0 frame:0
          TX packets:4930 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:17797368 (16.9 MiB)  TX bytes:809419 (790.4 KiB)
          Interrupt:51 Base address:0x8000
```

4.2. Wi-Fi

This section mainly introduces the configuration and use of Wi-Fi under Linux. Generally, the Wi-Fi module can support two working modes, STA mode and AP mode, and some devices also support STA and AP mode to work simultaneously. The STA mode allows the device to connect to an external Wi-Fi hotspot, and the AP mode turns the device into a Wi-Fi hotspot for other devices to connect to.

Currently, STA and AP are not supported to work simultaneously. The driver corresponding to the WM-N-BM-02 Wi-Fi module is:

```
root@myd-y6ull14x14:~# lsmod
Module                  Size  Used by
brcmfmac              176188  0
brcmutil               8416  1 brcmfmac
```

the driver load, the Wi-Fi firmware located at /lib/firmware/brcm will be loaded into the module. After the Wi-Fi module driver is loaded successfully, the network node "wlan0" of the Wi-Fi device is generated, as shown below:

```
root@myir:~# ifconfig -a
wlan0     Link encap:Ethernet  HWaddr C0:84:7D:B6:6B:3E
          UP BROADCAST MULTICAST  MTU:1500  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:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

1) Manually connect to WiFi hotspot for STA mode

Now try manually connecting to a nearby Wi-Fi hotspot "MYIR_TECH", a WPA2 encrypted Wi-Fi hotspot with the password myir@2016.

- **Ensure wlan0 network device is active**

```
root@myd-y6ul14x14:~# ifconfig wlan0 up
```

- **Scan the Wi-Fi hotspots nearby**

Scan the nearby Wi-Fi hotspots to get the list of nearby Wi-Fi hotspots as follows:

```
root@myd-y6ull14x14:~# iw dev wlan0 scan | grep SSID
        SSID: MI8
        SSID: DIRECT-JJFFmsBE
        SSID: feng
        SSID: HP-Print-3C-LaserJet Pro MFP
        SSID: MYIR_TECH
```

- **Use wpa_passphrase to Set the Wi-Fi name and password**

```
root@myir:~# wpa_passphrase MYIR_TECH myir@2016 >> /etc/wpa_supplicant.
conf
root@myir:~# cat /etc/wpa_supplicant.conf
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1

network={
        key_mgmt=NONE
}
network={
        ssid="MYIR_TECH"
        #psk="myir@2016"
        psk=6a85bda077eebd8473c9b585b74c823199c6cbbe66a4f1d9e40c9b94
e9605aa0
}
```

Generates a WPA PSK encrypted key from the ASCII password of an SSID.

- **Stop wpa_supplicant**

If wpa_supplicant is started, close it first.

```
root@myir:~# killall wpa_supplicant
```

- **Start wpa_supplicant**

wpa_supplicant is a tool to connect and configure WI-FI. Its main job is to interact with the driver via socket and report data to the user layer. The user layer can also send commands to wpa_supplicant via socket to dispatch drivers to operate WIFI chip. It usually runs in the background, as shown below:

```
root@myir:~# wpa_supplicant -B -D wext -c /etc/wpa_supplicant.conf -i wlan0
```

- -B : Run daemons in the background
- -D : Driver name
- -c : The path to the configuration file
- -i : Wi-Fi network device node

- **Assign IP Address for Wi-Fi**

```
root@myir:~# udhcpc -i wlan0
udhcpc: started, v1.31.1
udhcpc: sending discover
udhcpc: sending select for 192.168.40.107
udhcpc: lease of 192.168.40.107 obtained, lease time 7200
/etc/udhcpc.d/50default: Adding DNS 223.5.5.5
/etc/udhcpc.d/50default: Adding DNS 201.104.111.114

root@myir:~# ping www.baidu.com
PING www.baidu.com (112.80.248.75) 56(84) bytes of data.
64 bytes from 112.80.248.75 (112.80.248.75): icmp_seq=1 ttl=56 time=28.8 ms
64 bytes from 112.80.248.75 (112.80.248.75): icmp_seq=2 ttl=56 time=37.6 ms
64 bytes from 112.80.248.75 (112.80.248.75): icmp_seq=3 ttl=56 time=38.9 ms
```

2) Connect to Wi-Fi Hotspots Automatically

The manual connection will be lost after the reboot,If you need to automatically connect to "MYIR_TECH" when you start up, you need to write the SSID and KEY of the" MYIR_TECH" hotspot to the */etc/wpa_supplicant.conf* configuration file. Then configure and manage wlan0 network devices through systemd-networkd. All network configurations of systemd-networkd are stored at

/lib/systemd/network or */etc/systemd/network*, and create the wireless interface file "*/lib/systemd/network/10-wireless.network*", which is as follows:

```
root@myir:~# echo "[Match]" > /lib/systemd/network/10-wireless.network
root@myir:~# echo "Name=wlan0" >> /lib/systemd/network/10-wireless.netwo
rk
root@myir:~# echo "[Network]" >> /lib/systemd/network/10-wireless.network
root@myir:~# echo "DHCP=ipv4" >> /lib/systemd/network/10-wireless.network
root@myir:~# cat /lib/systemd/network/10-wireless.network
[Match]
Name=wlan0
[Network]
DHCP=ipv4
```

To connect this wireless interface to a specific network, we need information such as the SSID and password for the MYIR_TECH network. Associate the wireless network with the wireless interface. Here is the configuration file of the wlan0 automatic connection hotspot, the path is /etc/wpa_supplicant/wpa_supplicant-wlan0.conf:

```
root@myir:~# mkdir -p /etc/wpa_supplicant/
root@myir:~# echo "ctrl_interface=/var/run/wpa_supplicant" > /etc/wpa_supplic
ant/wpa_supplicant-wlan0.conf
root@myir:~# echo "eapol_version=1" >> /etc/wpa_supplicant/wpa_supplicant-
wlan0.conf
root@myir:~# echo "ap_scan=1" >> /etc/wpa_supplicant/wpa_supplicant-wlan0.
conf
root@myir:~# echo "fast_reauth=1" >> /etc/wpa_supplicant/wpa_supplicant-wl
an0.conf
root@myir:~# echo "" >> /etc/wpa_supplicant/wpa_supplicant-wlan0.conf
root@myir:~# wpa_passphrase MYIR_TECH myir@2016 >> /etc/wpa_supplicant
/wpa_supplicant-wlan0.conf
```

Once configured, you need to restart the network service.

```
root@myir:~# systemctl enable wpa_supplicant@wlan0.service
```

```
Created symlink /etc/systemd/system/multi-user.target.wants/wpa_supplicant@wlan0.service -> /lib/systemd/system/wpa_supplicant@.service.

root@myir:~# systemctl restart wpa_supplicant@wlan0.service
root@myir:~# systemctl restart systemd-networkd.service
```

Restart the development board and check the wlan0 network connection. It can be found that the wlan0 network has automatically connected to MYIR_TECH hotspot and obtained IP address, gateway, DNS and other information through DHCP.

5. Network Applications

The "myir-image-full" image of the device programmed from the factory contains some common network applications by default, which is convenient for users to develop or debug.

5.1. PING

PING is used primarily to test network connectivity. It can also test network latency and packet loss rates. Once the Ethernet connection is configured as in 4.1.1, PING can be used for simple testing of network connections.

1) Network Connection

When you connect a device to a switch or router via a CAT6 cable, the console displays the connection information that the kernel outputs, as follows:

```
[ 6601.055679] SMSC LAN8710/LAN8720 20b4000.ethernet-1:01: attached PHY driver [SMSC LAN8710/LAN8720] (mii_bus:phy_addr=20b4000.ethernet-1:01, irq=POLL)
```

2) PING Public Network

```
root@myd-y6ull14x14:~# ping www.baidu.com -I eth0
PING www.baidu.com (112.80.248.75) 56(84) bytes of data.
64 bytes from 112.80.248.75 (112.80.248.75): icmp_seq=1 ttl=56 time=28.3 ms
64 bytes from 112.80.248.75 (112.80.248.75): icmp_seq=2 ttl=56 time=26.4 ms
64 bytes from 112.80.248.75 (112.80.248.75): icmp_seq=3 ttl=56 time=34.5 ms
64 bytes from 112.80.248.75 (112.80.248.75): icmp_seq=4 ttl=56 time=106 ms
```

NOTE: Ping public network needs to ensure that DNS is working properly.

The above results show that the IP address of "www.baidu.com" after domain name resolution is 112.80.248.75, ICMP_SEQ stands for the number of ICMP packet, if the number is serial, no packet is lost; Time stands for the latency of the response, and the shorter the better, of course. In addition to testing Ethernet, the ping command can also be used to test Wi-Fi.

5.2. SSH

SSH stands for Secure Shell and was developed by the IETF's Network Working Group. SSH is an application layer based security protocol, which is more reliable and provides security for remote login sessions and other network services. Typically, Linux platforms use Dropbear or OpenSSH to implement both the server and client sides of SSH. Let's test the use of SSH client and server on Ethernet connection respectively. Current factory default includes OpenSSH 7.6 P1 (http://www.openssh.com/) client and server.

The Ethernet interface connection to SSH server is configured in accordance with Section 4.1.1. The configured Ethernet card address is as follows:

```
root@myd-y6ull14x14:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 5a:6a:a5:1f:d3:5b
          inet addr:192.168.30.202  Bcast:192.168.30.255
          Mask:255.255.255.0
          inet6 addr: fe80::586a:a5ff:fe1f:d35b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:102 errors:0 dropped:0 overruns:0 frame:0
          TX packets:163 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:11122 (10.8 KiB)  TX bytes:17354 (16.9 KiB)
```

The SSH server's IP address is 192.168.30.185. The following test can be conducted after the connection between the test device and SSH server is normal with the ping command.

- **SSH Client Test**

The device connects to the SSH server as an SSH client. The SSH command is used to log in the SSH server on the device. The commands and results are as follows:

```
myir@myir-server1:~$ ssh root@192.168.30.185
The authenticity of host '192.168.30.185 (192.168.30.185)' can't be established.
```

```
RSA key fingerprint is SHA256:8Pu4Od1+FdT34uG6eYpRgXbbvPeKOcUzyed/hn
pEsHI.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.30.185' (RSA) to the list of known hosts.
root@myd-y6ull14x14:~#
```

After successful login, automatically enter the console on the SSH server, and the user can perform their Linux user permission control on the remote server on the client side. If exit is required, simply execute the "exit" command from the current console.

- **SSH Server Test**

The device acts as an SSH server, and other devices are connected to the device remotely.

Since SSH service is also started on the device side by default, we can also use SSH commands on other devices with SSH clients to log in to the current device, with the following commands and results:

```
PC $ ssh root@192.168.0.16
myir
root@myir:~#
```

In the example above, we remotely log on to the device as root and go to the console to perform root user privileges on the device. If you need to exit, simply execute the "exit" command from the console.

OpenSSH is the primary connection tool for remote login using the SSH protocol. It encrypts all traffic to eliminate eavesdropping, connection hijacking, and other attacks. In addition, OpenSSH provides a range of large secure tunneling capabilities, multiple authentication methods, and complex and flexible configuration options. Users can modify the configuration files “ssh_config” and “sshd_config” in the “*/etc/ssh/*” directory as they wish.

For example, if you want the SSH server to allow the root account to log in remotely without a password, you can modify "*/etc/ssh/sshd_config*" on the SSH server device to add the following two lines of configuration.

```
PermitRootLogin yes
PermitEmptyPasswords yes
```

The above configuration poses a significant security risk and is typically used for remote deployment during the debug phase. In real products, it's usually turned off for security reasons.

5.3. SCP

SCP, short for Secure Copy, is a Secure remote file Copy command based on the SSH protocol on Linux systems and is very useful during system debugging.
In 4.2.2, we have introduced the example of using SSH protocol and SSH client and server for remote login. Here, we introduce the example of remote copy of files through SCP command:

1) Copy Files from Remote to Local

```
myir@myir-server1:~$ scp uboot.dtb root@192.168.30.202:/home/root/
uboot.dtb
                                                100% 1499KB    1.5MB/s
 00:00
myir@myir-server1:~$
```

Go to the development board home directory to see this file, as shown below:

```
root@myd-y6ull14x14:~# ls
uboot.dtb
root@myd-y6ull14x14:~# pwd
/home/root
```

2) Copy Files form Local to Remote

```
root@myd-y6ull14x14:~# scp 1.sh roy@192.168.30.185:~/
roy@192.168.30.185's password:
1.sh
100%    0       0.0KB/s    00:00
root@myd-y6ull14x14:~#
```

During the copying process, please follow the prompts to input. After successful verification, the file is copied from the device to the $HOME directory of the specified account on the server.

You can also copy directories by adding the "-r" parameter, as the SCP command helps.

5.4. FTP

FTP(File Transfer Protocol) is a network protocol for file transfer. The protocol follows the client-server communication model. To transfer files using FTP, the user needs to run an FTP client program and start a connection to a remote computer running FTP server software. After establishing the connection, the client can choose to send and/or receive a copy of the file. The FTP server listens for connection requests from the FTP client on TCP port 21. When a request is received, the server USES this port to control the connection and opens a separate port to transfer file data.

The default factory myir-image-full image contains a FTP client program FTP and server-side proftpd (see section 7.4 in “MYD-Y6ULX Software Development Guide" for migration of FTP application). Start the development board, on target side the root directory of the FTP is at the “/var/lib/ftp”, the following tests within the local area network (LAN) using FTP commands anonymously to the target device and from the development of the host (IP address 192.168.0.2) to the target device (IP address for 192.168.0.60) transfer file example.

1) Login from Host PC to Target Device with FTP

The development host is a Windows10 system PC, open the command window, login to the development board with FTP, user name is “ftp”, password is arbitrary.

```
C:\Users\myir>ftp 192.168.30.185
Connect to the 192.168.30.185 192.168.30.185。
220 ProFTPD Server (ProFTPD Default Installation) [192.168.30.185]
500 OPTS UTF8 not understood
User(192.168.30.185:(none)): ftp
331 Anonymous login ok, send your complete email address as your passwor
d
passwd:
230 Anonymous access granted, restrictions apply
ftp>
```

At this point, the current directory is development board */var/lib/ftp*, and then you can download and upload files.

2) Create Files on Target Device for Test

```
root@myd-y6ull14x14:~#/var/lib/ftp# touch test
root@myd-y6ull14x14:~#/var/lib/ftp# ls
aaa  test
root@myd-y6ull14x14:~#/var/lib/ftp# rm -r aaa
root@myd-y6ull14x14:~#/var/lib/ftp# touch aaa
root@myd-y6ull14x14:~#/var/lib/ftp# ls
aaa  test
```

3) View Files on Target Device with FTP at Host PC

```
ftp>ls
200 PORT command successful
150 Opening ASCII mode data connection for file list
aaa
test
226 Transfer complete
ftp: Received 14Bytes, Time Used:0.00Seconds 14000.00Bytes/Seconds
```

4) Download Files from the Target Device with FTP

The FTP command line supports downloading multiple files using the "mget" subcommand, but each file requires user confirmation, as shown below:

```
ftp>mget aaa test
200 Type set to A
Mget aaa? y
200 PORT command successful
150 Opening ASCII mode data connection for aaa
226 Transfer complete
Mget test? y
200 PORT command successful
150 Opening ASCII mode data connection for test
226 Transfer complete
```

```
ftp>
ftp>
```

5) Upload Files to the Target Device with FTP

FTP need ordinary user or root user login to upload files, here use root user login, user name is "root", no password, directly press" Enter" key, if you need password users can directly set the root password in the target development board:

- **Settings root User Password**

```
root@myir:~# passwd root
New password:
Retype new password:
passwd: password updated successfully
```

- **Modify Configuration and Restart Service**

To log in to the FTP server using a root account, you need to modify /etc/proftpd.conf the file and add a line of configuration "RootLogin on" to the file，and then restart the proftpd service:

```
systemctl restart proftpd
```

- **Connect target board with FTP**

```
D:\01-Document>ftp 192.168.0.60
Connect 192.168.0.60。
220 ProFTPD Server (ProFTPD Default Installation) [192.168.40.152]
500 OPTS UTF8 not understood
User(192.168.0.60:(none)): root
331 Password required for root
Password:
230 User root logged in
ftp> pwd
257 "/home/root" is the current directory
```

- **Upload files**

```
ftp> binary
```

```
200 Type set to I
ftp> put myapp
200 PORT command successful.
150 Opening BINARY mode data connection for 'myapp'.
226 Transfer complete.
1024 bytes sent in 0.02 secs (63.7308 kB/s)
ftp>bye
D:\01-Document>
```

FTP dates back to the early 1970s and was written with no security concerns. It does not use encryption on anything. The login credentials (such as username and password) and the data you download or upload are transmitted in clear text. Therefore, it is not suitable for transmitting sensitive information over the Internet. However, considering its good compatibility and stability, it is very convenient to use in LAN.

5.5. TFTP

Like FTP, TFTP uses client and server software to connect and transfer files between two devices, but the difference is that TFTP uses UDP protocol, which does not have the login function. It is very simple, especially suitable for transferring and backing up firmware, configuration files and other information on the device and server side. For example, TFTP protocol is supported in the common U-boot, which can load the server-side Linux system through the network and realize the function of network startup.

The default image file contains the TFTP client program provided by BusyBox, with the following command syntax:

```
root@myd-y6ull14x14:~# tftp --help
BusyBox v1.31.0 (2020-09-05 00:06:04 UTC) multi-call binary.

Usage: tftp [OPTIONS] HOST [PORT]
```

The detailed parameters are described below:

- -g : Get
- -p : Upload/Put
- -l : Local files
- -r : Remote files
- HOST: Remote host name or IP address
- [PORT]: Optional remote host port, default is 69

Users can choose tftp-hpa on Linux, can also choose tftpd32/64 (http://tftpd32.jounin.net/tftpd32_download.html) on Windows as the TFTP server application. Now, we choose "tftpd-hpa" on Ubuntu as an example to show the TFTP server configuration.

1) Install TFTP Server Application

Install the TFTP server application on the Ubuntu server as follows:

```
PC $ sudo apt-get install tftp-hpa tftpd-hpa
```

2) Configure TFTP Server

Create the TFTP server working directory and open the TFTP service configuration file as follows, please replace <WORKDIR> with yourselves work directory:

```
$ mkdir -p <WORKDIR>/tftpboot
$ chmod -R 777 <WORKDIR>/tftpboot
$ sudo vi /etc/default/tftpd-hpa
```

Modify or add the following fields:

```
TFTP_DIRECTORY="<WORKDIR>/tftpboot"
TFTP_OPTIONS="-l -c -s"
```

3) Restart TFTP Service

Restart the TFTP service on the Ubuntu server as follows:

```
$ sudo service tftpd-hpa restart
```

Once the TFTP server is configured, place a test file "*zImage*" in the configured "*<WORKDIR>/tftpboot/*" directory, and you can download and upload files using the TFTP client on the target device.

```
root@myir:~# tftp -g -r zImage -l zImage 192.168.0.2
```

The above command downloads the "*zImage*" from the TFTP server "*<WORKDIR>/tftpboot/*" directory to the current directory of the target device.

```
root@myir:~# tftp -p -l config -r config_01 192.168.0.2
```

The above command will upload the "*config*" file from the current directory on the target device to the directory previously configured on the TFTP server, under "*<WORKDIR>/tftpboot/*" and rename it to "*config_01*".

5.6. UDHCPC

DHCP (Dynamic Host Configuration Protocol) is a LAN network protocol. This refers to a range of IP addresses controlled by the server so that the client can automatically obtain the IP address and subnet mask assigned by the server when it logs in.

DHCP also includes two roles of server and client. In 4.1.1, we have tested the usage of DHCP client mode to automatically obtain IP address. When configuring WiFi AP mode in 4.1.2, we also tested the DHCP server mode to assign IP addresses to connected WiFi devices. Here is another way to get the IP address manually using the "dhclient" command and the "udhcpc" command, so that users can use it when debugging the network.

Connect the development board to the router with the CAT6 cable, manually assign the IP address to the eth0 network card using the DHCP client command, and observe the DHCP process to obtain the IP.

- **Configure the IP Address Dynamically by udhcpc**

```
root@myir:~# udhcpc -i eth0
udhcpc: started, v1.31.1
udhcpc: sending discover
udhcpc: sending select for 192.168.30.199
[ 3038.408887] IPv4: martian source 255.255.255.255 from 192.168.8.1, on dev
 eth0
[ 3038.414720] ll header: 00000000: ff ff ff ff ff ff 20 da 22 fc ca 1b 08 00
udhcpc: lease of 192.168.30.199 obtained, lease time 3600
/etc/udhcpc.d/50default: Adding DNS 223.5.5.5
/etc/udhcpc.d/50default: Adding DNS 201.104.111.114
```

Either way, you can configure eth0 with the IP address, gateway, subnet mask, DNS, and other information as follows:

```
root@myir:~# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 26:13:52:CE:15:40
          inet addr:192.168.30.199  Bcast:192.168.30.255  Mask:255.255.255.0
```

```
          inet6 addr: fe80::2413:52ff:fece:1540/64 Scope:Link
          inet6 addr: fd20:da22:fcca:1b00:2413:52ff:fece:1540/64 Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:20745 errors:0 dropped:752 overruns:0 frame:0
          TX packets:633 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1529726 (1.4 MiB)  TX bytes:93845 (91.6 KiB)
          Interrupt:51 Base address:0x8000
root@myir:~# cat /etc/resolv.conf
nameserver 223.5.5.5
nameserver 201.104.111.114
```

5.7. IPTables

Iptables is an administrative tool for IPv4 packet filtering and NAT. It is used to set up, maintain, and check the IP packet filtering rule table in the Linux kernel. Several different tables can be defined. Each table contains many built-in chains and can also contain user-defined chains. Each chain is a list of rules that can match a set of packets. Each rule specifies how to handle matching packets.

Devices using Linux systems typically use the "iptables" tool to configure firewalls. "iptables" handles packets based on methods defined by packet filtering rules, such as accept, reject, and drop. Let's use "iptables" to test intercepting ICMP packets, preventing other devices on the network from ping them. Specific commands to use see: https://linux.die.net/man/8/iptables.

1) Configure iptables for Target Device

Use the "iptables" configuration on the target device to discard the input ICMP package and not respond to ping probes from other hosts, with the following command:

```
root@myd-y6ull14x14:~# iptables -A INPUT -p icmp --icmp-type 8 -j DROP
root@myd-y6ull14x14:~# iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
-A INPUT -p icmp -m icmp --icmp-type 8 -j DROP
```

2) Ping the Target Device

Ping the target device on the development host and specifying a deadline of 10, the result is shown as follows:

```
PC$ ping 192.168.30.185 -w 10
PING 192.168.30.2 (192.168.30.2) 56(84) bytes of data.

--- 192.168.0.60 ping statistics ---
10 packets transmitted, 0 received, 100% packet loss, time 9064ms
```

The above results show that the development host cannot ping the target device after setting the firewall.

3) Delete Rules for iptables

```
root@myd-y6ull14x14:~# iptables -F
root@myd-y6ull14x14:~# iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
```

4) Ping the Target Device Again

```
PC$ ping 192.168.0.60 -w 5
PING 192.168.0.60 (192.168.0.60) 56(84) bytes of data.
64 bytes from 192.168.0.60: icmp_seq=1 ttl=64 time=0.254 ms
64 bytes from 192.168.0.60: icmp_seq=2 ttl=64 time=0.219 ms
64 bytes from 192.168.0.60: icmp_seq=3 ttl=64 time=0.222 ms
64 bytes from 192.168.0.60: icmp_seq=4 ttl=64 time=0.226 ms
64 bytes from 192.168.0.60: icmp_seq=5 ttl=64 time=0.238 ms
64 bytes from 192.168.0.60: icmp_seq=6 ttl=64 time=0.236 ms

--- 192.168.0.60 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 4996ms
rtt min/avg/max/mdev = 0.219/0.232/0.254/0.019 ms
```

Once the "iptables" rules are cleared, ping the target device again from the development host is ready to ping. The above example is just a simple demonstration, but iptables with various rules can be very powerful, and I won't go into the details here.

5.8. Ethtool

Ethtool is a tool for viewing and modifying Ethernet device parameters, which is useful during network debugging. Use this command to look at the Ethernet card information and try to modify its parameters.

We look at the help information for the command through "ethtool -h".

```
root@myd-y6ull14x14:~# ethtool --help
ethtool version 5.2
Usage:
        ethtool DEVNAME Display standard information about device
        ethtool -s|--change DEVNAME     Change generic options
                [ speed %d ]
                [ duplex half|full ]
                [ port tp|aui|bnc|mii|fibre ]
                [ mdix auto|on|off ]
                [ autoneg on|off ]
                [ advertise %x ]
                [ phyad %d ]
                [ xcvr internal|external ]
                [ wol p|u|m|b|a|g|s|f|d... ]
                [ sopass %x:%x:%x:%x:%x:%x ]
                [ msglvl %d | msglvl type on|off ... ]
        ethtool -a|--show-pause DEVNAME Show pause options.....
.....
```

View the basic information of the current device's Ethernet as follows.

```
root@myd-y6ull14x14:~# ethtool eth0
Settings for eth0:
        Supported ports: [ TP MII ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
        Supported pause frame use: Symmetric
```

```
        Supports auto-negotiation: Yes
        Supported FEC modes: Not reported
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
        Advertised pause frame use: Symmetric
        Advertised auto-negotiation: Yes
        Advertised FEC modes: Not reported
        Link partner advertised link modes:  10baseT/Half 10baseT/Full
        Link partner advertised pause frame use: Symmetric Receive-only
        Link partner advertised auto-negotiation: Yes
        Link partner advertised FEC modes: Not reported
        Speed: 10Mb/s
        Duplex: Full
        Port: MII
        PHYAD: 1
        Transceiver: internal
        Auto-negotiation: on
        Supports Wake-on: g
        Wake-on: d
        Link detected: yes
```

From the "ethtool" command, you can see that the current Ethernet supports 10, 100, and gigabit half-duplex and full-duplex modes. The current connection status is negotiated gigabit, full-duplex mode, MII interface, PHY address is 6, and so on.

We can also use the "ethtool" to set Ethernet parameters, which are useful for debugging and diagnosing Ethernet, such as forcing Ethernet to be set to 100 megabytes full duplex and turning off self-negotiation, as shown below:

```
# ethtool -s eth0 speed 100 duplex full autoneg off
```

For more information about "ethtool", please refer to the man page of "ethtool" at: http://man7.org/linux/man-pages/man8/ethtool.8.html .

5.9. iPerf3

IPerf3 is a tool that actively measures the maximum realized bandwidth over an IP network. It supports tuning various parameters such as test time, buffer size, and protocols (TCP, UDP, SCTP with IPv4 and IPV6). The iPerf3 can be divided into server mode or client mode by role, and we can use it to test and view network bandwidth in TCP mode, TCP window values, probability of retransmission, etc., as well as to test packet loss rates, latency, and jitter under specified UDP bandwidth.

We used an Ubuntu 16.04 system, a computer with gigabit card as the server of iPerf3, and the device under test as the client to test the performance on TCP and UDP for the device's Ethernet card.

Install iPerf3 on the server, as follows:

```
PC $ sudo apt-get install iperf3
```

Connect the server and device directly via CAT6 and configure their respective IP addresses. For example, we set the server IP to 192.168.0.2 and the device IP to 192.168.0.60, and use the ping command to test to make sure they are connected.

Note: Try not to connect to routers or switches to prevent test results from being affected by intermediate devices.

1) Test Performance under TCP Mode

- **Server Side（192.168.30.3）**

Iperf3 on the server uses the -s parameter to indicate that it is operating in the server mode.

```
PC $ $ iperf3 -s -i 2
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
```

- **Client Side（192.168.30.185）**

Iperf3 works on the device as client, under TCP mode, where the parameters are described below:

- -c 192.168.30.168 : Work on the client side, connect to the server side 192.168.30.168
- -i 2 : Test results are reported at an interval of 2 seconds
- -t 60 : The total test time is 60 seconds

```
root@myd-y6ull14x14:~# iperf3  -c 192.168.30.3 -i 2 -t 60
Connecting to host 192.168.30.3, port 5201
[  4] local 192.168.30.156 port 35213 connected to 192.168.30.3 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-2.00   sec  22.9 MBytes  96.2 Mbits/sec    0    229 KBytes
[  4]   2.00-4.00   sec  22.1 MBytes  92.5 Mbits/sec    0    229 KBytes
[  4]   4.00-6.00   sec  22.4 MBytes  93.9 Mbits/sec    0    229 KBytes
[  4]   6.00-8.00   sec  22.7 MBytes  95.1 Mbits/sec    0    229 KBytes
[  4]   8.00-10.00  sec  22.4 MBytes  93.8 Mbits/sec    0    229 KBytes
[  4]  10.00-12.00  sec  22.4 MBytes  93.8 Mbits/sec    0    229 KBytes
[  4]  12.00-14.00  sec  22.4 MBytes  93.8 Mbits/sec    0    229 KBytes
[  4]  14.00-16.00  sec  22.7 MBytes  95.1 Mbits/sec    0    229 KBytes
[  4]  16.00-18.00  sec  22.4 MBytes  93.8 Mbits/sec    0    229 KBytes
[  4]  18.00-20.00  sec  22.4 MBytes  93.8 Mbits/sec    0    229 KBytes
[  4]  20.00-22.00  sec  22.4 MBytes  93.9 Mbits/sec    0    229 KBytes
[  4]  22.00-24.00  sec  22.7 MBytes  95.1 Mbits/sec    0    229 KBytes
[  4]  24.00-26.00  sec  22.4 MBytes  93.8 Mbits/sec    0    229 KBytes
[  4]  26.00-28.00  sec  22.4 MBytes  93.8 Mbits/sec    0    229 KBytes
[  4]  28.00-30.00  sec  22.4 MBytes  93.8 Mbits/sec    0    229 KBytes
[  4]  30.00-32.00  sec  22.7 MBytes  95.1 Mbits/sec    0    229 KBytes
[  4]  32.00-34.00  sec  22.4 MBytes  93.9 Mbits/sec    0    229 KBytes
[  4]  34.00-36.00  sec  22.4 MBytes  93.8 Mbits/sec    0    229 KBytes
[  4]  36.00-38.00  sec  22.4 MBytes  93.8 Mbits/sec    0    229 KBytes
[  4]  38.00-40.00  sec  22.7 MBytes  95.0 Mbits/sec    0    229 KBytes
[  4]  40.00-42.00  sec  22.4 MBytes  93.9 Mbits/sec    0    229 KBytes
[  4]  42.00-44.00  sec  22.4 MBytes  93.8 Mbits/sec    0    229 KBytes
[  4]  44.00-46.00  sec  22.4 MBytes  93.8 Mbits/sec    0    229 KBytes
[  4]  46.00-48.00  sec  22.7 MBytes  95.0 Mbits/sec    0    229 KBytes
[  4]  48.00-50.00  sec  22.4 MBytes  93.9 Mbits/sec    0    229 KBytes
```

```
[  4]   50.00-52.00   sec   22.4 MBytes   93.9 Mbits/sec    0     229 KBytes
[  4]   52.00-54.00   sec   22.4 MBytes   93.8 Mbits/sec    0     229 KBytes
[  4]   54.00-56.00   sec   22.4 MBytes   93.8 Mbits/sec    0     229 KBytes
[  4]   56.00-58.00   sec   22.7 MBytes   95.1 Mbits/sec    0     229 KBytes
[  4]   58.00-60.00   sec   22.4 MBytes   93.8 Mbits/sec    0     229 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-60.00   sec   674 MBytes   94.2 Mbits/sec    0             sender
[  4]   0.00-60.00   sec   673 MBytes   94.1 Mbits/sec                  receive
r
iperf Done.
```

After 60 seconds, the client finished the test and displayed the above test results, indicating that the TCP bandwidth was about 94Mbps, no retransmission.

At the same time, the server also displays the test results as follows, and then continues to listen on port 5201 waiting for the client to connect:

```
$ iperf3 -s -i 2
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
```

2) Test Performance under UDP Mode

- **Server Side（192.168.30.3）**

Continue running iPerf3 on the server and use the -s parameter to indicate that you are working in the server mode.

```
PC $ $ iperf3 -s -i 2
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
```

- **Client Side（192.168.30.185）**

Iperf3 works on the device as client, under UDP mode, where the parameters are described below:

- -u : Work in UDP mode
- -c 192.168.30.185 : Work on the client side, connect to the server side 192.168.30.3
- -i 2 : Test results are reported at an interval of 2 seconds
- -t 10 : The total test time is 10 seconds
- -b 1G : Set the UDP transmission bandwidth as 1G.

```
root@myd-y6ull14x14:~# iperf3 -c 192.168.30.3 -u -i 2 -t 60 -b 1G -R
Connecting to host 192.168.30.3, port 5201
Reverse mode, remote host 192.168.30.3 is sending
[  4] local 192.168.30.156 port 49521 connected to 192.168.30.3 port 5201
[ ID] Interval           Transfer     Bandwidth       Jitter    Lost/Total Datag
rams
[  4]   0.00-2.00   sec  21.8 MBytes  91.5 Mbits/sec  0.175 ms  29/2822 (1%)
[  4]   2.00-4.00   sec  22.8 MBytes  95.8 Mbits/sec  0.194 ms  0/2923 (0%)
[  4]   4.00-6.00   sec  22.9 MBytes  95.9 Mbits/sec  0.168 ms  0/2926 (0%)
[  4]   6.00-8.00   sec  22.8 MBytes  95.8 Mbits/sec  0.190 ms  0/2924 (0%)
[  4]   8.00-10.00  sec  22.9 MBytes  95.8 Mbits/sec  0.176 ms  0/2925 (0%)
[  4]  10.00-12.00  sec  22.8 MBytes  95.8 Mbits/sec  0.173 ms  0/2924 (0%)
[  4]  12.00-14.00  sec  22.8 MBytes  95.8 Mbits/sec  0.174 ms  0/2924 (0%)
[  4]  14.00-16.00  sec  22.9 MBytes  95.8 Mbits/sec  0.172 ms  0/2925 (0%)
[  4]  16.00-18.00  sec  22.8 MBytes  95.8 Mbits/sec  0.176 ms  0/2924 (0%)
[  4]  18.00-20.00  sec  22.9 MBytes  95.8 Mbits/sec  0.207 ms  0/2925 (0%)
[  4]  20.00-22.00  sec  22.8 MBytes  95.8 Mbits/sec  0.168 ms  0/2924 (0%)
[  4]  22.00-24.00  sec  22.8 MBytes  95.8 Mbits/sec  0.166 ms  0/2924 (0%)
[  4]  24.00-26.00  sec  22.9 MBytes  95.8 Mbits/sec  0.183 ms  0/2925 (0%)
[  4]  26.00-28.00  sec  22.8 MBytes  95.8 Mbits/sec  0.174 ms  0/2924 (0%)
[  4]  28.00-30.00  sec  22.9 MBytes  95.8 Mbits/sec  0.184 ms  0/2925 (0%)
[  4]  30.00-32.00  sec  22.8 MBytes  95.8 Mbits/sec  0.171 ms  0/2924 (0%)
[  4]  32.00-34.00  sec  22.9 MBytes  95.8 Mbits/sec  0.215 ms  0/2925 (0%)
[  4]  34.00-36.00  sec  22.8 MBytes  95.8 Mbits/sec  0.155 ms  0/2924 (0%)
[  4]  36.00-38.00  sec  22.8 MBytes  95.8 Mbits/sec  0.176 ms  0/2924 (0%)
```

```
[  4]  38.00-40.00  sec  22.9 MBytes  95.8 Mbits/sec  0.207 ms  0/2925 (0%)
[  4]  40.00-42.00  sec  22.8 MBytes  95.8 Mbits/sec  0.185 ms  0/2924 (0%)
[  4]  42.00-44.00  sec  22.9 MBytes  95.8 Mbits/sec  0.200 ms  0/2925 (0%)
[  4]  44.00-46.00  sec  22.8 MBytes  95.8 Mbits/sec  0.178 ms  0/2924 (0%)
[  4]  46.00-48.00  sec  22.8 MBytes  95.8 Mbits/sec  0.204 ms  0/2924 (0%)
[  4]  48.00-50.00  sec  22.9 MBytes  95.8 Mbits/sec  0.173 ms  0/2925 (0%)
[  4]  50.00-52.00  sec  22.8 MBytes  95.8 Mbits/sec  0.171 ms  0/2924 (0%)
[  4]  52.00-54.00  sec  22.9 MBytes  95.8 Mbits/sec  0.163 ms  0/2925 (0%)
[  4]  54.00-56.00  sec  22.8 MBytes  95.8 Mbits/sec  0.179 ms  0/2924 (0%)
[  4]  56.00-58.00  sec  22.8 MBytes  95.8 Mbits/sec  0.174 ms  0/2924 (0%)
[  4]  58.00-60.00  sec  22.9 MBytes  95.8 Mbits/sec  0.178 ms  0/2925 (0%)
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Jitter    Lost/Total Datag
rams
[  4]  0.00-60.00  sec  685 MBytes  95.7 Mbits/sec  0.177 ms  29/87640 (0.
033%)
[  4] Sent 87640 datagrams

iperf Done.[  5]  0.00-10.00  sec  119 MBytes  100 Mbits/sec  0.078 ms
0/85604 (0%)  receiver
iperf Done.
```

The client completes the test after 10 seconds and displays the above test results, showing that UDP did not lose packets when specifying a bandwidth of 100Mbps.

At the same time, the server also displays the test results as follows, and then continues to listen on port 5201 waiting for the client to connect:

```
$ $ iperf3 -s -i 2
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
Accepted connection from 192.168.30.185, port 38372
[  5] local 192.168.30.2 port 5201 connected to 192.168.30.206 port 42492
[ ID] Interval       Transfer     Bandwidth       Jitter    Lost/Total Datagrams
```

```
[  5]   0.00-2.00   sec  23.4 MBytes  98.0 Mbits/sec  0.079 ms  0/16918 (0%)
[  5]   2.00-4.00   sec  23.8 MBytes   100 Mbits/sec  0.077 ms  0/17264 (0%)
[  5]   4.00-6.00   sec  23.8 MBytes   100 Mbits/sec  0.065 ms  0/17268 (0%)
[  5]   6.00-8.00   sec  23.8 MBytes   100 Mbits/sec  0.084 ms  0/17265 (0%)
[  5]   8.00-10.00  sec  23.8 MBytes   100 Mbits/sec  0.074 ms  0/17267 (0
%)
[  5]  10.00-10.04  sec   486 KBytes   100 Mbits/sec  0.068 ms  0/344 (0%)
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval       Transfer     Bandwidth       Jitter    Lost/Total Datagrams
[  5]   0.00-10.04  sec  119 MBytes 99.6 Mbits/sec  0.068 ms  0/86326 (0%)
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
```

The client adjust the “-b” parameter and continues to increase the specified UDP bandwidth. When packet loss occurs, the UDP bandwidth measured is the actual UDP bandwidth.If the specified bandwidth reaches the Ethernet card's highest bandwidth of 1Gbps and still no packets are lost, the bandwidth returned by the iPerf3 test is the actual UDP bandwidth. For example, the following example specifies a bandwidth of 1000Mbps, still without packet loss, but the actual bandwidth is around 600Mbps.

Iperf3 also has a number of parameters that can be configured during the test so that users can tailor the test to their actual application needs. For example, you can increase the value of the “-t” parameter for long time stress test, or specify the “-P” parameter for multiple concurrent stress tests. For more information about the IPerf3 test, please refer to https://iperf.fr/iperdoc. Ph #3doc.

6. Linux Graphics System

Linux graphics system is a more complex subsystem of Linux system, has been in constant change. It sits between the underlying display related device drivers and the upper user interface applications, masking various underlying hardware differences while providing a unified API for the upper user interface.

6.1. QT

QT is a cross-platform C++ graphical user interface application development framework. It can be used to develop both GUI programs and non-GUI programs, such as console tools and servers. Qt is an object-oriented framework that uses special code-generation extensions and macros. Qt is easy to extend and allows for true component programming.
The development board burns systems with the Qt runtime library at delivery and provides a rich HMI demonstration system. See "MEasy HMI2.0 Development Manual" for details.

1) Introduction of QT Running Environment

When running Qt applications, the Qt operating environment, such as platform plug-ins, display parameters, input devices and cursor pointers, can be appropriately configured according to different hardware and software requirements.

- **Platform Plugins Configuration**

On embedded Linux systems, multiple platform plug-ins can be used: EGLFS, LinuxFB, DirectFB, or Wayland. However, the availability of these plug-ins depends on the nature of the actual hardware platform and how Qt is configured. EGLFS is the default plug-in on many motherboards. If not, use the QT_QPA_PLATFORM environment variable to request another plug-in. Also, for quick tests, use -platform with command-line arguments that have the same syntax. For example, in the user console, you can configure it using the following command.

```
root@myir:/home# export QT_QPA_PLATFORM=linuxfb
root@myir:/home# ./mxapp2
```

Or use the "-platform linufb" instruction platform plug-in, as shown below:

```
root@myir:/home# ./mxapp2 -platform linuxfb
```

- **Display Parameters Configuration**

QT applications can write applications that match the screen by using the QScreen class or QDesktopWidget to get the parameters associated with the screen display. Getting the screen resolution from a QScreen or QDesktopWidget is generally fine, but sometimes the physical size is not necessarily correct due to the display drive. At this point, you can configure and adjust the following parameters to make the elements displayed on the actual interface fit the size of the display screen.

Figure 5-1. Environment variables associated with the QT EGLS plug-in

Environment Variable	Description
fb=/dev/fbN	Specify the frame buffer device. On multiple monitor settings, this setting allows you to run the application on different monitors. Currently, it is not possible to use multiple frame buffers from one Qt application.
size=<width>x<height>	Specify the screen size (in pixels). The plug-in tries to query the physical and logical display size from the frame buffer device. However, this query may not always lead to correct results. You may need to specify these values explicitly.
mmsize=<width>x<height>	Specify the physical width and height (in millimeters).
offset=<width>x<height>	Specify the offset (in pixels) of the upper left corner of the screen. The default position is (0, 0).
nographicsmodeswitch	Specifies not to switch the virtual terminal to graphics mode (KD_GRAPHICS). Generally, enabling graphics mode disables blinking cursors and screen blanking. However, after setting this parameter, these two functions will also be skipped.
tty=/dev/ttyN	Override the virtual console. Only used when nographicsmodeswitch is not set.

In general, the default configuration works well, but if there is a real mismatch between the display elements and the actual screen, the relevant parameters can be adjusted as described above.

- **Input Devices Configuration**

If you want to enable TSLIB support, you need to set the QT_QPA_EGLFS_TSLIB (for EGlFS) or QT_QPA_FB_TSLIB (for linuxfb) environment variable to 1.Refer to TSLIB for specific usage at https://github.com/libts/tslib/blob/master/README.md.

NOTE：The TSLIB input handler, often used for resistive touches, generates mouse events and supports only single touch, and requires screen calibration for initial use.

2) Start Qt Applications

Devices that typically use touch do not need to display the mouse pointer, and users can set environment variables to hide the mouse pointer before executing the application.

If you are using the "eglfs" platform plug-in, set it as follows:

```
export QT_QPA_EGLFS_HIDECURSOR=1
```

If you are using a "linuxfb" platform plug-in, the Settings are as follows:

```
export QT_QPA_FB_HIDECURSOR=1
```

"myir-image-full" image of MYD-Y6ULX development board factory has MEasy HMI demo program, using linuxfb platform plug-in and EvdevTouch handler, corresponding boot command program is as follows:

```
root@myd-y6ull14x14:/home/root/# export QT_QPA_FB_HIDECURSOR=1
root@myd-y6ull14x14:/home/root/# ./mxapp2 -platform linuxfb &
qt.qpa.input: xkbcommon not available, not performing key mapping
qml: index=0
qml: currentIndex=0
qml: index=0
libpng warning: iCCP: known incorrect sRGB profile
libpng warning: iCCP: known incorrect sRGB profile
libpng warning: iCCP: known incorrect sRGB profile
```

Mxapp2 is started by default. If you need to run your own Qt application, you need to stop MXAPP2 before starting other applications.

7. Multimedia

7.1. Camera

This chapter uses the open source uvc_stream to demonstrate the use of camera. MYD-Y6ULX provides a parallel Camera interface (J9), which can be connected to MY-CAM011B camera modules, and FPC cables are used to connect the modules. Due to the influence of the signal sequence, please do not directly insert the module of other models of Camera, otherwise it will cause damage to the module or development board. uvc_stream is the data transmission through the network. You need to set the Ethernet IP address of the MYD-Y6ULX board first, corresponding to the eth0 device in the system. The device of the MY-CAM011B module in the Linux system can be queried through the v4l2-ctl command. The output information i.MX6S_CSI represents the Camera controller, and the corresponding device is /dev/video1. The'-y' in the uvc_stream parameter is to use the yuyv method, and after the'-P' is the login password for setting the web interface, the user name is uvc_user by default. '-r' is the specified resolution, currently only supports 800x600. You can use ctrl + c to stop.

1) View the Device Topology

```
root@myd-y6ull14x14:~# Ifconfig eth0 192.168.30.42
root@myd-y6ull14x14:~# v4l2-ctl --list-devices
i.MX6S_CSI (platform:21c4000.csi):
        /dev/video1

pxp (pxp_v4l2):
        /dev/video0
```

2) Start Display Server

```
# ./uvc_stream -d /dev/video1 -y -P 123456 -r 800x600
```

The uvc_stream program supports two kinds web functions, snapshot and streaming. The snapshot function request URL is snapshot.jpeg, and streaming function request URL is stream.mjpeg.

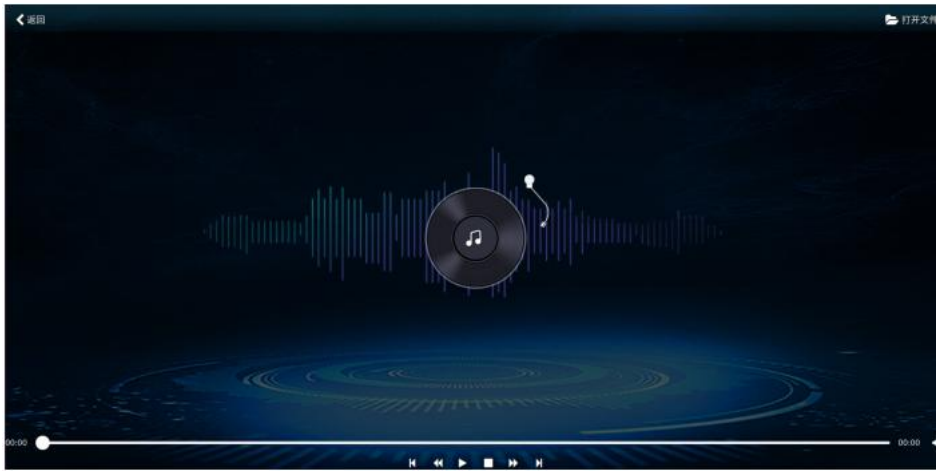
Let PC and board has same network, open your browser, visit http://192.168.1.42:8080/stream.mjpeg.After enter, you can see the login window, login with uvc_user, 123456.Now, you can see video stream from web on MY-CAM011B captured.

7.2. Audio

MYD-Y6ULX uses the audio encoding chip WM8904CGEFL/V to expand 1 3.5mm headphone output and 1 audio linear input. The I2S end of WM8904CGEFL/V is connected to the SAI2 controller of the processor, and the audio chip I2C is hung on the CPU I2C2 interface. This section is to test playing audio. There are two ways, one is to play directly from the HMI2.0 application, and the other is to play the audio manually through the aplay tool.

1) Play Music on MEasy HMI 2.0

For HMI multimedia music playing, please refer to the HMI manual "MEasy HMI2.0 Development Manual".



7-2 Audio playback

2) Play Music by Command

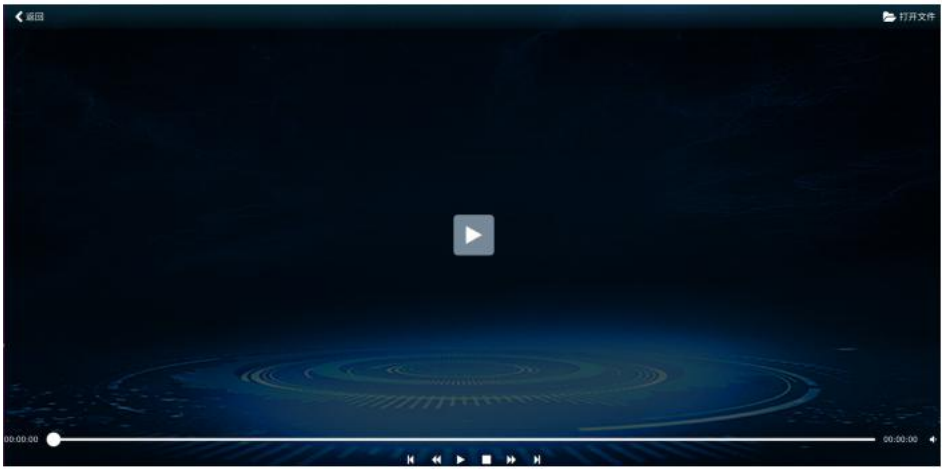
```
root@myir:~/audio# aplay att16k.wav
Press 'k' to see a list of keyboard shortcuts.
Now playing /home/root/audio/att16k.wav
Redistribute latency...
0:00:14.9 / 0:00:14.9
Reached end of play list.
```

7.3. Video

This section mainly tests the two ways to play the video, one is direct HMI click to play the video, the other is to play the video by command.

1) Play Videos on MEasy HMI 2.0

For HMI multimedia video playback, please refer to the HMI manual "MEasy HMI2.0 Development Manual".



7-3 Video playback

8. System Tools

Some commonly used system tools are included in the default image of the factory, which is convenient for users to view and manage various resources of the system in system debugging or actual deployed products, and can also be called in SHELL scripts or other applications. These tools may not fully meet the needs of the user's system customization, at which point the system developer needs to make appropriate adjustments based on the actual situation.

8.1. Compress and Decompress Tools

This section mainly tests the compression and decompression tools of the system. Compression is to compress multiple files into a compression package , then it will be convenient for file transfer. And decompression will make the compressed files back to the original size for easy use. This section illustrates the file system with tools such as "tar", "gzip", "gunzip", and so on.

1) tar Tool

The "tar" tool, now commonly used in Linux, not only packages files, but also compresses, views, adds, and unzips them. Here is the package operation.

- **Syntax Format**

Use the --help parameter to see the "tar" syntax format as follows:

```
root@myir:~# tar --help
Usage: tar [OPTION...] [FILE]...
GNU 'tar' saves many files together into a single tape or disk archive, and ca
n
restore individual files from the archive.

Examples:
  tar -cf archive.tar foo bar  # Create archive.tar from files foo and bar.
  tar -tvf archive.tar         # List all files in archive.tar verbosely.
```

```
tar -xf archive.tar          # Extract all files from archive.tar.
```

The detailed parameters are described below:

- -c : Create a compressed file parameter instruction
- -x : Unpacking a compressed file parameter instruction
- -t : Check out the files in a tar file! Note in particular that c/x/t can only exist in one parameter. Do not exist at the same time! Because it is impossible to compress and uncompress at the same time.
- -z : Do you have gzip properties at the same time? Is gzip compression required?
- -j : Do you have a bzip2 attribute at the same time? Is bzip2 compression necessary?
- -v : Show files during compression! This is often used, but is not recommended when running in the background
- -f : Use file name, please note, the file name must be followed -f parameter immediately
- -p : Use the original properties of the original file (properties do not change depending on the user)
- -P : Can use absolute path to compress
- --exclude FILE: During compression, do not pack a FILE

- **Use tar to Compress**

Create a new test.txt file and type the following command to package the file in ".gz" format:

```
root@myir:~# tar -czf test.tar.gz test.txt
root@myir:~# ls
OpenAMP_TTY_echo.elf  rs485_write  test.tar.gz  uart_test
rs485_read            test.c       test.txt     wifi.conf
```

With the z parameter, the ".tar.gz" or ".tgz" represents a gzip-compressed tar file.

- **Use tar to Decompress**

Unpack the file which has been compressed as tar.gz.

```
root@myir:~# tar -xvf test.tar.gz
test.txt
root@myir:~# ls
OpenAMP_TTY_echo.elf   rs485_write   test.tar.gz   uart_test
rs485_read             test.c        test.txt      wifi.conf
```

2) gzip Tool

"gzip" is a file compression and decompression command frequently used in Linux systems, both convenient and easy to use.

- **Syntax Format**

Enter the following command at the development board terminal to view the "gzip" syntax:

```
root@myir:~# gzip --help
BusyBox v1.31.1 () multi-call binary.
Usage: gzip [-cfkdt] [FILE]...
```

- **Use gzip to Compress**

```
root@myir:~# gzip test.txt
root@myir:~# ls
OpenAMP_TTY_echo.elf   rs485_write   test.tar.gz   uart_test
rs485_read             test.c        test.txt.gz   wifi.conf
```

- **Use gzip to Decompress**

Unzip the file with "gunzip", as shown below:

```
root@myir:~# gunzip test.txt.gz
root@myir:~# ls
OpenAMP_TTY_echo.elf   rs485_write   test.tar.gz   uart_test
rs485_read             test.c        test.txt      wifi.conf
```

8.2. File System Tools

The main test system file system tools, this section will introduce several common file system management tools. System comes with file system tools “mount”, “mkfs”, “fsck”, “dumpe2fs”.

1) mount tool

“mount” is a command under Linux that connects a partition to a folder on Linux, associating the partition with that directory, so simply accessing the folder is equivalent to accessing the partition, in the following syntax format:

```
root@myir:~# mount -h

Usage:
 mount [-lhV]
 mount -a [options]
 mount [options] [--source] <source> | [--target] <directory>
 mount [options] <source> <directory>
 mount <operation> <mountpoint> [<target>]

Mount a filesystem.
```

An example of mounting the fourth partition of an SD card is shown below:

```
root@myir:~# mount /dev/mmcblk1p4 /mnt/
[10677.124115] EXT4-fs (mmcblk1p4): mounted filesystem with ordered data mode. Opts: (null)
[10677.130984] ext4 filesystem being mounted at /mnt supports timestamps until 2038 (0x7fffffff)
```

2) mkfs tool

After partitioning the hard disk, the next step is to set up the Linux file system. Similar to a formatted hard disk in Windows. Setting up a file system on a hard disk partition washes out the data on the partition and is not recoverable, so make sure that the data on the partition is no longer used before setting up the

file system. The command to set up the file system is mkfs, in the following syntax format:

```
root@myir:~# mkfs -h
Usage:
 mkfs [options] [-t <type>] [fs-options] <device> [<size>]

Make a Linux filesystem.

Options:
 -t, --type=<type>   filesystem type; when unspecified, ext2 is used
      fs-options       parameters for the real filesystem builder
      <device>          path to the device to be used
      <size>            number of blocks to be used on the device
 -V, --verbose        explain what is being done;
                        specifying -V more than once will cause a dry-run
 -h, --help           display this help
 -V, --version        display version

For more details see mkfs(8).
```

An example of formatting the seventh partition of an SD card is shown below:

```
root@myir:~# mkfs -t ext3 -V -c /dev/mmcblk1p7
mkfs from util-linux 2.32.1
mkfs.ext3 -c /dev/mmcblk1p7
mke2fs 1.44.3 (10-July-2018)
/dev/mmcblk1p7 contains a ext4 file system labelled 'userfs'
        created on Thu Aug 13 04:32:02 2020
Proceed anyway? (y,N) y
Creating filesystem with 330532 1k blocks and 82656 inodes
Filesystem UUID: 46bd5bb8-1882-4a68-901a-e11b4e71924b
Superblock backups stored on blocks:
        8193, 24577, 40961, 57345, 73729, 204801, 221185
Checking for bad blocks (read-only test): done
```

```
Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
```

3) fsck tool

The "fsck" command is mainly used to check the correctness of the file system. When the file system fails, the fsck instruction can be used to try to fix it. And fix the Linux disk. For example:

```
root@myir:~# fsck -a /dev/mmcblk1p7
fsck from util-linux 2.32.1
/dev/mmcblk1p7: clean, 11/82656 files, 20726/330532 blocks
```

4) dumpe2fs tool

Prints information about super blocks and blocks groups of existing file systems on a particular device. The development board enters the following command to see the application syntax:

```
root@myir:~# dumpe2fs -h
dumpe2fs 1.45.4 (23-Sep-2019)
Usage: dumpe2fs [-bfghimxV] [-o superblock=<num>] [-o blocksize=<num>]
device
```

View detailed properties of the file system formatted, for example, by entering a command to view the details of a disk:

```
root@myir:~# dumpe2fs /dev/mmcblk1p7
dumpe2fs 1.44.3 (10-July-2018)
Filesystem volume name:   <none>
Last mounted on:          <not available>
Filesystem UUID:          46bd5bb8-1882-4a68-901a-e11b4e71924b
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:      has_journal ext_attr resize_inode dir_index filetype sp
arse_super large_file
```

```
Filesystem flags:         unsigned_directory_hash
```

Check the number of inodes on a disk. Inodes also consume disk space, so when the disk is formatted, the operating system automatically divides the disk into two areas. One is the data area, where file data is stored; The other is the inode section (inode table), which holds the information contained in the inode.

```
root@myir:~# dumpe2fs /dev/mmcblk1p7 | grep -i "inode size"
dumpe2fs 1.44.3 (10-July-2018)
Inode size:               128
```

Look at the number of blocks on a disk. When the operating system reads the hard disk, it will not read sectors one by one, which is too inefficient. Instead, it will read multiple sectors in a row at one time, that is, read a "block" at one time. This "block" of multiple sectors is the smallest unit of file access.

```
root@myir:~# dumpe2fs /dev/mmcblk1p7 | grep -i "block size"
dumpe2fs 1.44.3 (10-July-2018)
Block size:               1024
```

8.3. Disk Management Utils

This section mainly test system disk management tools, including several common disk management tools. The system comes with disk management tools "fdisk", "dd", "du", "df", "cfdisk" and so on, These commands allow you to monitor your daily disk usage.

1) fdisk Disk Partitioning Tool

The "fdisk" disk partitioning tool has applications for DOS, Windows, and Linux. In Linux, "fdisk" is a menu-based command. To partition a hard disk with "fdisk", you can add the hard disk to partition directly after the fdisk command as a parameter. The syntax is as follows:

```
root@myir:~# fdisk -h
Usage:
 fdisk [options] <disk>         change partition table
 fdisk [options] -l [<disk>] list partition table(s)
```

Partition the eMMC as follows:

```
root@myir:~# fdisk /dev/mmcblk2p2
Welcome to fdisk (util-linux 2.32.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
The old ext4 signature will be removed by a write command.
Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xd2dcd6ef.
Command (m for help):
```

2) dd tool

The "dd" command is used to copy the specified input file to the specified output file. And the format can be converted during replication. The difference between the "dd" command and the "cp" command is that the "dd" command can be done on a floppy disk that has not been created, and the data

copied to the floppy disk is actually a mirror file. Similar to the "diskcopy" command in DOS. The format of the "dd" command is:

```
dd [<if=input file name/device name>] [<of=output file name/device name>]
 [bs=block size(byte)] [count=block count]
```

An example of creating a file of 2M size is shown below:

```
root@myir:~# time dd if=/dev/zero of=ffmpeg1 bs=2M count=1 conv=fsync
1+0 records in
1+0 records out
2097152 bytes (2.1 MB, 2.0 MiB) copied, 0.195515 s, 10.7 MB/s
real    0m 0.20s
user    0m 0.00s
sys     0m 0.05s
```

3) du tool

The du command is used to display disk space usage. This command shows, level by level, how each level of subdirectory of the specified directory occupies a block of file system data. "du" is generally used in the following syntax:

```
root@myir:~# du --help
Usage: du [OPTION]... [FILE]...
  or:  du [OPTION]... --files0-from=F
Summarize disk usage of the set of FILEs, recursively for directories.
```

Partial parameter are as follows:

- -a: Displays the size of all directories or files
- -h: Take K,M and G Bytes as units to improve the readability of inform ation
- -k: Output in Kilo Bytes
- -m: Output in Mega Bytes

Statistics the file size generated by the dd command:

```
root@myir:~# du ffmpeg1
2048    ffmpeg1
root@myir:~# du -h ffmpeg1
```

```
2.0M      ffmpeg1
```

4) df tool

Used to display disk usage statistics for the current file system on a Linux system, commonly used as follows:

```
root@myir:~# df --help
Usage: df [OPTION]... [FILE]...
Show information about the file system on which each FILE resides,
or all file systems by default.
```

Partial parameter are as follows:

- -h: The appropriate units can be displayed based on the size used
- -i: View the number of inodes and inode usage under the partition
- -T: Print out the file system type

To see the number of inodes and inode usage under the partition, use the following command:

```
root@myir:~# df -i
Filesystem      Inodes IUsed  IFree IUse% Mounted on
devtmpfs         37543   477  37066    2% /dev
/dev/mmcblk1p6 289600 25068 264532    9% /
tmpfs            54499    17  54482    1% /dev/shm
tmpfs            54499   704  53795    2% /run
tmpfs            54499    14  54485    1% /sys/fs/cgroup
tmpfs            54499    22  54477    1% /tmp
/dev/mmcblk1p4   16384    24  16360    1% /boot
/dev/mmcblk1p5    4096    34   4062    1% /vendor
tmpfs            54499    26  54473    1% /var/volatile
/dev/mmcblk1p7   65536   438  65098    1% /usr/local
tmpfs            54499    10  54489    1% /run/user/0
```

The inode is defined by the system during formatting, and the inode depends on the size of the disk partition. When we reach 100 percent inode usage, we cannot write data to disk even though we still have disk space left. Please refer to Section 2.5 for other application examples.

8.4. Process Management Utils

Process is also an important concept in the operating system, it is a process of the execution of a program, the program is a static description of the process, the system run every program is in its process run. All processes in Linux are interconnected, and all processes have a parent except for initializing the process. Instead of being created, new processes are copied or copied from previous processes. All processes in Linux are derived from an "init" process with process number 1. Linux consists of three different types of processes, each with its own characteristics and attributes:

- The Interaction Process: A process started by a Shell can run both in the foreground and in the background.
- The Batch Process: This process has no connection to the terminal and is a sequence of processes. This process is committed to a process waiting for execution in queue order.
- Monitor Processes (daemons): Daemons are always active and usually run in the background, and daemons are usually started automatically by the system at the beginning through script activation or root.

For Linux system, process management is an important link, and process management is usually achieved through process management tools. There are several commonly used process management commands in Linux system: "ps", "top", "vmstat", "kill" and so on.

1) ps tool

"ps" is a command to show the status of current processes. The general syntax is as follows:

```
root@myir:~# ps --help
Usage:
 ps [options]

 Try 'ps --help <simple|list|output|threads|misc|all>'
  or 'ps --help <s|l|o|t|m|a>'
```

```
 for additional help text.
For more details see ps(1).
```

Partial parameter are as follows:

- -u: Organize a user-centric display of process status information.
- -a: A process that is not terminal dependent.
- -x: The process associated with the terminal.

Usually the above commands are combined with: aux

- -e: Show all processes; The equivalent of ax.
- -f: Displays full format program information.

Usually the above commands are combined with: ef

- -H: Shows the number of processes at the process level
- -F: Display more program information

Usually the above commands are combined with: eHF

An example of displaying information for all processes is shown below:

```
root@myir:~# ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.6  1.1  23804  4820 ?        Ss   06:57   0:07 /sbin/init
root         2  0.0  0.0      0     0 ?        S    06:57   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        I<   06:57   0:00 [rcu_gp]
root         4  0.0  0.0      0     0 ?        I<   06:57   0:00 [rcu_par_gp]
root         8  0.0  0.0      0     0 ?        I<   06:57   0:00 [mm_percpu_wq]
root         9  0.0  0.0      0     0 ?        S    06:57   0:00 [ksoftirqd/0]
root        10  0.0  0.0      0     0 ?        I    06:57   0:00 [rcu_preempt]
root        11  0.0  0.0      0     0 ?        S    06:57   0:00 [migration/0]
root        12  0.0  0.0      0     0 ?        S    06:57   0:00 [cpuhp/0]
root        13  0.0  0.0      0     0 ?        S    06:57   0:00 [cpuhp/1]
```

A brief explanation of some of the taskbars above:

- VSZ: Virtual memory size
- RSS: Resident size
- STAT: There are several process states

- R: Running
- S: Interruptable sleeping
- D: Uninterruptable sleeping
- T: Stopped
- Z: Zombie process
- +: Foreground process
- N: Low priority process
- l: Multithreaded process

2) top tool

The "top" command puts quite a bit of overall system performance information on one screen. The display can also be changed in an interactive manner. Dynamic continuous monitoring of the running state of the process, the "top" syntax is generally as follows:

```
root@myir:~# top -help
  procps-ng 3.3.16
Usage:
  top -hv | -bcEHiOSs1 -d secs -n max -u|U user -p pid(s) -o field -w [cols]
```

See section 2.1 for an example of dynamically viewing system processes.

3) vmstat: Virtual memory statistics tool

This command looks at the usage status of the memory space and gives you a snapshot of the performance of the entire system. "vmstat" runs in two modes: sample mode and average mode. If no parameters are specified, "vmstat" statistics run in average mode and "vmstat" displays the average of all statistics since system startup. The common syntax and parameters are as follows:

```
root@myir:~# vmstat -h
Usage:
 vmstat [options] [delay [count]]
Options:
 -a, --active           active/inactive memory
 -f, --forks            number of forks since boot
 -m, --slabs             slabinfo
```

```
-n, --one-header         do not redisplay header
-s, --stats              event counter statistics
-d, --disk               disk statistics
-D, --disk-sum           summarize disk statistics
-p, --partition <dev>    partition specific statistics
-S, --unit <char>        define display unit
-w, --wide               wide output
-t, --timestamp          show timestamp

-h, --help       display this help and exit
-V, --version    output version information and exit
```

"vmstat" runs in average mode, showing the average of all statistics since system startup:

```
root@myir:~# vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free  buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  0      0 200640 13556 131852    0    0    39    11  148  227  6  3 90  0  0
```

A brief explanation of some of the taskbars above:

- r: Number of processes currently running. Instead of waiting for I/O, these processes have ready numbers to run. Ideally, the number of processes that can run is equal to the number of CPU available
- b: The number of blocked processes waiting for I/O to complete
- forks: The number of times a new process was created
- in: The number of system interrupts
- cs: The number of times a context switch occurred in the system
- us: Percentage of total CPU time consumed by user processes
- sy: The percentage of total CPU time consumed by the system code, which includes time consumed in the System, IRQ, and Softirq states
- wa: The percentage of total CPU consumed waiting for I/O
- id: The percentage of total CPU time consumed by the system idle

Statistical system various data details are as follows:

```
root@myir:~# vmstat -s
        435992 K total memory
         90008 K used memory
         82380 K active memory
         88924 K inactive memory
        200528 K free memory
         13556 K buffer memory
        131900 K swap cache
             0 K total swap
             0 K used swap
             0 K free swap
         20273 non-nice user cpu ticks
          4412 nice user cpu ticks
         19691 system cpu ticks
        630164 idle cpu ticks
           241 IO-wait cpu ticks
             0 IRQ cpu ticks
            22 softirq cpu ticks
             0 stolen cpu ticks
        127150 pages paged in
         36195 pages paged out
             0 pages swapped in
             0 pages swapped out
        969112 interrupts
       1492562 CPU context switches
    1581090644 boot time
         10649 forks
```

A brief explanation of some of the information above:

- total memory: Total system memory
- used memory: Used memory

- CPU ticks：This shows the CPU time since the system started, where "tick" is a unit of time
- forks：Roughly speaking, this represents the number of new processes that have been created since system startup

"vmstat" provides a great deal of information about the performance of Linux systems. It is one of the core tools for investigating system problems.

4) kill tool

Sends the specified signal to the appropriate process. Not specifying the model sends SIGTERM (15) to terminate the specified process. If the program cannot be terminated with the available "-KILL" parameter, the signal it sends is SIGKILL(9), which forces the process to be terminated. The process number can be viewed using the "ps" command or the "jobs" command. A root user will affect the user's process, and a non-root user can only affect his or her own process. The general syntax of the kill command is as follows：

```
kill [ -s signal | -p ] [ -a ] pid ...
kill -l [ signal ]
```

Partial parameter are as follows：

- -s：Specifies the signal to send
- -p：simulate transmitting signal
- -l：Specifies the name list of signals
- pid：The ID number of the process to abort
- Signal：According to the signal

First use "ps -ef" and the pipe command to determine the PID to kill the process.

```
root@myir:~# ps -ef | grep mxapp2
root      1045     1 15 15:51 ?     00:00:02 /home/mxapp2 -platform eglfs
root      1174  1117  0 15:51 ttySTM0  00:00:00 grep mxapp2
```

Then type the following command to terminate the process：

```
root@myir:~# kill 1045
```

The “killall” command terminates all processes within the same process group, allowing you to specify the name of the process to be terminated instead of the PID process number.

```
root@myir:~# killall mxapp2
root@myir:~#
```

9. Application Development

This chapter mainly introduces some basic information for secondary development of the current SDK. The current SDK provides two configuration reference images, one is "myir-image-core", which is mainly for non-GUI applications; the other is "myir-image-full", which adds some GUI applications on the basis of "myir-image-core", such as Qt runtime libraries, graphics processing related libraries and demo applications. For information about these two images, please refer to the "MYD-Y6ULX SDK2.0.0 Release notes".

9.1. Development Language

1) SHELL

Shell is a program written in C language, which is a bridge for users to use Linux. Shell is both a command language and a programming language. There are many types of common Linux shells:

- Bourne Shell (/usr/bin/sh or /bin/sh)
- Bourne Again Shell (/bin/bash)
- C Shell (/usr/bin/csh)
- K Shell (/usr/bin/ksh)
- Shell for Root (/sbin/sh)

Bash, which is also supported by the current SDK as default, is highlighted here. The following example is the script "start.sh" in myir-image-full image when MEasy HMI 2.0 boots up. The contents are as follows:

```
#!/bin/sh -e

echo "Start MYiR HMI V2.0..."

export TSLIB_TSDEVICE=/dev/input/event1
export TSLIB_CONFFILE=/etc/ts.conf
export TSLIB_CALIBFILE=/etc/pointercal
```

```
export  TSLIB_PLUGINDIR=/usr/lib/ts
export  TSLIB_CONSOLEDEVICE=none
export  QT_QPA_FB_TSLIB=1
export  QT_QPA_GENERIC_PLUGINS=tslib:/dev/input/event1

/home/root/mxapp2  -platform  linuxfb  &

exit  0
```

This script first sets the environment variable MEasy HMI V2.0, and then calls the /home/mxapp2 executable to start MEasy HMI V2.0 according to different kernel device tree information with different parameters. The "&" at the end of the parameter means the program will enter the background to run.

2) C/C++

C/C++ is the most commonly used programming language for low-level application development under the Linux platform, and is also the most efficient language after assembly. Development using C/C++ usually adopts the mode of cross-development, that is, development is carried out on the development host side, binary execution files are compiled and generated running on the target machine, and then deployed to run on the target machine.In this way, the SDK built based on Yocto needs to be installed first. Please refer to "MYD-Y6ULX Yocto Software Development Guide" for installation steps. After installation, the SDK environment needs to be configured as follows:

```
PC  $:source  /home/alex/workspace/imx6ul_tools_5.4/tools_qt5/environment-setup-cortexa7t2hf-neon-poky-linux-gnueabi
```

This section demonstrates application development by writing a simple Hello World instance, and the following demo program "hello.c" is written on the development host side:

```
#include<stdio.h>
int  main(int  argc,char  *argv[])
{
```

```
    printf("hello world!\n");
    return 0;
}
```

Next, the application is compiled with "$CC", because the corresponding header file and link are needed when compiling. "$CC" contains the corresponding system library and configuration information. If you directly compile with "arm-poky-linux-gnueabi-gcc", you will not find the header files.

```
myir@myir-server1:~/test$ source /home/alex/workspace/imx6ul_tools_5.4/tools_qt5/environment-setup-cortexa7t2hf-neon-poky-linux-gnueabi
myir@myir-server1:~/test$ cat main.c
#include <stdio.h>


int main(int argc, char *argv[])
{
        print("myir test!\n");
        return 0;
}
myir@myir-server1:~/test$ $CC main.c -o main
myir@myir-server1:~/test$ file main
```

The generated execution file is then copied to the target machine through the "scp" command and executed as follows:

```
root@myir:~# ./main
hello world!
```

For more complex examples and development methods, refer to the "MYD-Y6ULX Yocto Software Development Guide" for instructions on application migration.

3) Python

Python is a high-level programming language with interpreted, object-oriented, dynamic data types. Python was invented by Guido van Rossum in late 1989, and the first public release was released in 1991. Like the Perl language, the Python

source code is also licensed under the GPL(GNU General Public License). This section focuses on testing the use of Python, both from the Python command line and from the script.

- **Check the Supported Version of Python**

```
root@myd-y6ull14x14:~# ls /usr/bin/python*
/usr/bin/python3       /usr/bin/python3.7m
/usr/bin/python3.7     /usr/bin/python3.7m-config-lib
```

- **Python Interactive Mode**

Start Python and enter the following text at the Python prompt, then press Enter to see how it works:

```
root@myd-y6ull14x14:~# python3
Python 3.7.5 (default, Sep  5 2020, 00:13:46)
[GCC 9.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print("myir test")
```

In Python version 3.7.5, the output from the above instance is as follows:

```
Hello, Python!
>>>
```

Run "exit()" function or press "ctrl+D" to exit Python:

```
>>> exit()
root@myir:~#
```

- **Write a Python Script for Testing**

Write a simple Python script, and all Python files will have the.py extension.

```
root@myir:~# vi test.py
root@myir:~# cat test.py
#!/usr/bin/python3
print ("Hello, Python!")
```

The Python3 interpreter executes the script in the */usr/bin* directory using the following command.

```
root@myir:~# chmod +x test.py
root@myir:~# ./test.py
Hello, Python!
```

The Python3 interpreter is called with script parameters to start executing the script until it is finished. When the script has finished executing, the interpreter is no longer valid. The current system does not support “pip” commands , and clients need to port the “pip” tool if they need it.

9.2. Database

A Database is a warehouse that organizes, stores, and manages data in terms of data structures. There are many types of databases, commonly used databases are Access, Oracle, Mysql, SQL Server, SQLite and so on.

- **System SQLlite**

SQLite is an embedded SQL database engine. Unlike most other SQL databases, SQLite does not have a separate server process. SQLite reads and writes directly to a normal disk file. A complete SQL database containing multiple tables, indexes, triggers, and views is contained in a single disk file. This is a lightweight database, an ACID-compliant relational database management system, designed for embedded use and currently used in many embedded products. Its footprint is very low, and in embedded devices, it may only need a few hundred K of memory. This database runs faster than MySQL or PostgreSQL, so here's a quick test.

Start sqlite3 and create a new database <testdb.db>. Enter the following command into the terminal interface.

```
root@myir:~# sqlite3 testDB.db
SQLite version 3.31.1 2020-01-27 19:55:54
Enter ".help" for usage hints.
sqlite>
```

The above command creates a file "*testDB.db*", in the current directory. This file will be used as a database by the SQLite engine. Notice that the sqlite3 command provides a "sqlite>" prompt after successfully creating the database file.

With the database created, you can use the ".databases" command of SQLite to check if it is in the database list, as shown below:

```
sqlite> .databases
main: /home/root/testDB.db
sqlite>
```

Exit the SQLite prompt with the ".quit" command, as shown below:

```
sqlite>  .quit
root@myir:~#
```

If you want to learn more about SQLlite related information, please refer to the website https://www.sqlite.org/docs.html.

9.3. Application Localization for Qt

This section discusses localization-related issue and localization test with Qt applications. Localization refers to that a program or software, on the basis of supporting internationalization, gives the language information of a specific area of a program so that it can adapt to the use of the population of a specific area in the processing of information input and output. Some locale variables used by the program are allowed to be configured dynamically at program execution time. This chapter is mainly for the localization of QT applications, taking MEasy HMI 2.0 as an example for illustration.

1) Multiple Language

This section mainly uses MEasy HMI 2.0 as an example to illustrate the practical application of multiple languages in QT projects.

- **Open mxapp2 Demo Project**

MEasy HMI 2.0 corresponding project source code is "*Mxapp2.tar.gz*", copy to the environment built by the above document, and use QT Creator to open the project.

- **Generate *.ts File for Qt Project**

Enter the source directory of the MXAPP2 project through the terminal, and execute the following command to generate the translation file.

```
PC$ lupdate mxapp2.pro
Info: creating stash file /home/qinlh/download/MXAPP/.qmake.stash
Updating 'languages/language_zh.ts'...
    Found 202 source text(s) (0 new and 202 already existing)
Updating 'languages/language_en.ts'...
Found 202 source text(s) (0 new and 202 already existing)
```

The project has completed the translation work before the release, here will not regenerate the "*language_zh.ts*" and "*language_en.ts*" files. The "*MXAPP2*" application display Chinese by loading the *qm files generated from

language_zh.ts, and display English by loading the *qm files generated from language_en.ts.

- **Translation**

Open the language_en.ts file, for the translated source string is <source> node inside the content, the translated target string is <translation> inside the content, the user can modify according to the needs.

```
<message>
        <location filename="../Album.qml" line="50"/>
        <source>返回</source>
        <translation type="unfinished">Return</translation>
</message>
```

- **Generate *.qm Files for Qt**

*.ts file needs to be manually generated for translation template file after modification. The following command can be used to generate translation template file.

```
PC$ lrelease language_en.ts -qm language_en.qm
Updating 'language_en.qm'...
    Generated 183 translation(s) (2 finished and 181 unfinished)
Ignored 21 untranslated source text(s)
PC$ lrelease language_zh.ts -qm language_zh.qm
Updating 'language_en.qm'...
    Generated 183 translation(s) (2 finished and 181 unfinished)
    Ignored 21 untranslated source text(s)
```

- **Apply Translation Files**

The use of the translation file needs to be called through the application code, the specific call method refers to the load Language member function in the source code “*translator.cpp*”.

2) Fonts

- **Install Fonts for Qt Applications**

Font files can be placed directly into the development board file system /usr/lib/ directory.

```
# ls /usr/lib/fonts/msyh.ttc
/usr/lib/fonts/msyh.ttc
```

Or add directly to the QT project inside.

```
PC $ tree
└── fontawesome-webfont.ttf
```

- **Use Fonts in Qt Applications**

The font file needs to be called through the application code, In addition to using "*msyh. ttc*" to display text, MXAPP2 uses a special font called "*fontawesome-webfont.ttf*" to display icons.Regarding the specific method for loading fonts , please refer to IconFontInit () function in "main.cpp".

```
void iconFontInit()
{
    int fontId_digital = QFontDatabase::addApplicationFont(":/fonts/DIGITAL/DS-DIGIB.TTF");
    int fontId_fws = QFontDatabase::addApplicationFont(":/fonts/fontawesome-webfont.ttf");
    QString fontName_fws = QFontDatabase::applicationFontFamilies(fontId_fws).at(0);
    QFont iconFont_fws;
    iconFont_fws.setFamily(fontName_fws);
    QApplication::setFont(iconFont_fws);
    iconFont_fws.setPixelSize(20);
}
```

3) Virtual Keyboard from Qt

This chapter mainly takes MEasy HMI 2.0 as an example to illustrate the practical application of soft keyboard in QT project. Before reading the following contents, please refer to Chapter 3.1 Environment Construction of "MEasy HMI 2.0 Development Manual", and setup the compilation environment for MEasy HMI.

Since version 5.7, QT Virtual Keyboard component has been added to the official source code. QT 5.12 version has been used by MYIR in the published MEasy HMI 2.0 application, and the Virtual Keyboard component has been configured.

- **Embedded Virtual Keyboard into QML**

The soft keyboard provided by Qt can only be used in QML code, and the location of the soft keyboard pop-up and the size of the soft keyboard need to be defined before the call, as shown in the following code.

```
InputPanel {
id: inputPanel
x: adaptive_width/8
y: adaptive_height/1.06
z:99
anchors.left: parent.left
anchors.right: parent.right

states: State {
name: "visible"
when: inputPanel.active
PropertyChanges {
target: inputPanel
        y: adaptive_height/1.06 - inputPanel.height
}
}
```

- **Trigger Virtual Keyboard**

The soft keyboard is triggered by QML TextField, TextEdit or other editable components. The user only needs to add this component to the UI component to call up the soft keyboard on the UI and use it. If it is a QT system, you can use the Qt VirtualKeyboard input that comes with Qt.

```
TextField{
id: netmask_input
InputMethodHints: Qt.ImhFormattedNumbersOnly
onAccepted: digitsField.focus = true
font.family: "Microsoft YaHei"
color: "white"
}
```

- **Use Virtual Keyboard**

For the use of soft keyboard, please refer to Chapter 2.6, "MEasy HMI 2.0 Development Manual", for the UI interface of system setting. The user clicks on the editable interface component and the soft keyboard pops up.

10. Reference

- **Yocto Project BSP Development Guide**

https://www.yoctoproject.org/docs/3.1.1/bsp-guide/bsp-guide.html

- **Yocto Project Linux Kernel Development Guide**

https://www.yoctoproject.org/docs/3.1.1/kernel-dev/kernel-dev.html

- **Linux Kernel Watchdog Information**

https://www.kernel.org/doc/html/latest/watchdog/index.html

- **Qt for Embedded Linux**

https://doc.qt.io/qt-5/embedded-linux.html

- **Wayland Architecture**

https://wayland.freedesktop.org/architecture.html

- **Systemd Network Configuration**

https://www.freedesktop.org/software/systemd/man/systemd.network.html

Appendix A

Warranty & Technical Support Services

MYIR Electronics Limited is a global provider of ARM hardware and software tools, design solutions for embedded applications. We support our customers in a wide range of services to accelerate your time to market.

MYIR is an ARM Connected Community Member and work closely with ARM and many semiconductor vendors. We sell products ranging from board level products such as development boards, single board computers and CPU modules to help with your evaluation, prototype, and system integration or creating your own applications. Our products are used widely in industrial control, medical devices, consumer electronic, telecommunication systems, Human Machine Interface (HMI) and more other embedded applications. MYIR has an experienced team and provides custom design services based on ARM processors to help customers make your idea a reality.

The contents below introduce to customers the warranty and technical support services provided by MYIR as well as the matters needing attention in using MYIR’ s products.

Service Guarantee

MYIR regards the product quality as the life of an enterprise. We strictly check and control the core board design, the procurement of components, production control, product testing, packaging, shipping and other aspects and strive to provide products with best quality to customers. We believe that only quality products and excellent services can ensure the long-term cooperation and mutual benefit.

Price

MYIR insists on providing customers with the most valuable products. We do not pursue excess profits which we think only for short-time cooperation. Instead, we hope to establish long-term cooperation and win-win business with customers. So we will offer reasonable prices in the hope of making the business greater with the customers together hand in hand.

Delivery Time

MYIR will always keep a certain stock for its regular products. If your order quantity is less than the amount of inventory, the delivery time would be within three days; if your order quantity is greater than the number of inventory, the delivery time would be always four to six weeks. If for any urgent delivery, we can negotiate with customer and try to supply the goods in advance.

Technical Support

MYIR has a professional technical support team. Customer can contact us by email (support@myirtech.com), we will try to reply you within 48 hours. For mass production and customized products, we will specify person to follow the case and ensure the smooth production.

After-sale Service

MYIR offers one year free technical support and after-sales maintenance service from the purchase date. The service covers:

Technical support service

MYIR offers technical support for the hardware and software materials which have provided to customers;

- To help customers compile and run the source code we offer;
- To help customers solve problems occurred during operations if users follow the user manual documents;
- To judge whether the failure exists;
- To provide free software upgrading service.

However, the following situations are not included in the scope of our free technical support service:

- Hardware or software problems occurred during customers' own development;
- Problems occurred when customers compile or run the OS which is tailored by themselves;
- Problems occurred during customers' own applications development;
- Problems occurred during the modification of MYIR's software source code.

After-sales maintenance service

The products except LCD, which are not used properly, will take the twelve months free maintenance service since the purchase date. But following situations are not included in the scope of our free maintenance service:

- The warranty period is expired;
- The customer cannot provide proof-of-purchase or the product has no serial number;
- The customer has not followed the instruction of the manual which has caused the damage the product;
- Due to the natural disasters (unexpected matters), or natural attrition of the components, or unexpected matters leads the defects of appearance/function;
- Due to the power supply, bump, leaking of the roof, pets, moist, impurities into the boards, all those reasons which have caused the damage of the products or defects of appearance;
- Due to unauthorized weld or dismantle parts or repair the products which has caused the damage of the products or defects of appearance;
- Due to unauthorized installation of the software, system or incorrect configuration or computer virus which has caused the damage of products.

Warm tips

1. MYIR does not supply maintenance service to LCD. We suggest the customer first check the LCD when receiving the goods. In case the LCD cannot run or no display, customer should contact MYIR within 7 business days from the moment get the goods.

2. Please do not use finger nails or hard sharp object to touch the surface of the LCD.

3. MYIR suggests user purchasing a piece of special wiper to wipe the LCD after long time use, please avoid clean the surface with fingers or hands to leave fingerprint.

4. Do not clean the surface of the screen with chemicals.

5. Please read through the product user manual before you using MYIR’ s products.

6. For any maintenance service, customers should communicate with MYIR to confirm the issue first. MYIR’ s support team will judge the failure to see if the goods need to be returned for repair service, we will issue you RMA number for return maintenance service after confirmation.

Maintenance period and charges

- MYIR will test the products within three days after receipt of the returned goods and inform customer the testing result. Then we will arrange shipment within one week for the repaired goods to the customer. For any special failure, we will negotiate with customers to confirm the maintenance period.
- For products within warranty period and caused by quality problem, MYIR offers free maintenance service; for products within warranty period but out of free maintenance service scope, MYIR provides maintenance service but shall charge some basic material cost; for products out of warranty period, MYIR provides maintenance service but shall charge some basic material cost and handling fee.

Shipping cost

During the warranty period, the shipping cost which delivered to MYIR should be responsible by user; MYIR will pay for the return shipping cost to users when the product is repaired. If the warranty period is expired, all the shipping cost will be responsible by users.

Products Life Cycle

MYIR will always select mainstream chips for our design, thus to ensure at least ten years continuous supply; if meeting some main chip stopping production, we will inform customers in time and assist customers with products updating and upgrading.

Value-added Services

1. MYIR provides services of driver development base on MYIR’ s products, like serial port, USB, Ethernet, LCD, etc.

2. MYIR provides the services of OS porting, BSP drivers’ development, API software development, etc.

3. MYIR provides other products supporting services like power adapter, LCD panel, etc.

4. ODM/OEM services.

MYIR Electronics Limited

Room 04, 6th Floor, Building No.2, Fada Road,

Yunli Inteiligent Park, Bantian, Longgang District.

Support Email: support@myirtech.com

Sales Email: sales@myirtech.com

Phone: +86-755-22984836

Fax: +86-755-25532724

Website: www.myirtech.com