

MYD-YT113X_Linux

Software Evaluation Guide



File Status: [<input checked="" type="checkbox"/>] Draft [<input type="checkbox"/>] Release	FILE ID:	MYIR-MYD-YT113X-SW-EG-ZH-L5.4.61
	VERSION:	V1.1[DOC]
	AUTHOR:	Nico
	CREATED:	2023.05.01
	UPDATED:	2023.09.06



Revision History

VERSION	AUTHOR	PARTICIPANT	DATE	DESCRIPTION
V1.0	Nico	Licy	2023.05.01	Initial Version
V1.0	Nico	Licy	2023.09.06	Add applicable model MYD-YT113-I Delete RS232 test and add RS485 test.



CONTENT

Revision History	- 3 -
CONTENT	- 4 -
1. Overview	- 9 -
1.1. Hardware Resources.....	- 9 -
1.2. Software Resources	- 9 -
1.3. Documents.....	- 9 -
1.4. Preparation	- 10 -
2. Core Components	- 11 -
2.1. CPU	- 11 -
2.1.1. View the CPU Information.....	- 11 -
2.1.2. View CPU Utilization.....	- 12 -
2.1.3. Gets CPU Temperature.....	- 13 -
2.2. Memory.....	- 14 -
2.2.1. View memory information	- 15 -
2.2.2. Get memory usage	- 16 -
2.2.3. Memory Stress Test.....	- 16 -
2.3. eMMC and nand test	- 17 -
2.3.1. eMMC test	- 17 -
2.3.2. Nand Test.....	- 20 -
2.4. RTC.....	- 21 -
2.4.1. View the system RTC device.....	- 21 -
2.4.2. Set system time.....	- 21 -
2.5. Watchdog.....	- 22 -
2.5.1. Application testing watchdog.....	- 22 -
2.6. PMIC.....	- 23 -
2.6.1. View the current supported modes of the development board.....	- 24 -
2.6.2. Methods for writing in user space	- 24 -
3. Basic Peripheral Interface	- 25 -





3.1. GPIO	- 25 -
3.1.1. Exporting GPIO.....	- 25 -
3.1.2. Set/View GPIO Direction.....	- 25 -
3.1.3. Set/view GPIO values	- 26 -
3.2. LED.....	- 26 -
3.2.1. The directory to operate LEDs is "/sys/class/leds"	- 26 -
3.2.2. Heartbeat light led-blue as an example to test LED.....	- 26 -
3.3. Key	- 27 -
3.3.1. Device Tree Configuration Information.....	- 27 -
3.3.2. Key Test.....	- 28 -
3.4. USB	- 31 -
3.4.1. View print information for inserted usb.....	- 32 -
3.4.2. U disk mount read/write.....	- 32 -
3.4.3. Uninstall USB drive	- 32 -
3.5. Micro SD	- 33 -
3.5.1. View TF card capacity	- 33 -
3.5.2. Viewing TF card partition information.....	- 33 -
3.5.3. TF card performance test	- 34 -
3.6. ADC.....	- 36 -
3.6.1. Read GPADC test values	- 36 -
3.6.2. Read TPADC test values.....	- 36 -
3.7. Display.....	- 37 -
3.7.1. Device Tree Configuration Information.....	- 38 -
3.7.2. Display Solution Portfolio	- 38 -
3.8. Touch Panel	- 38 -
3.8.1. evtest	- 39 -
3.9. Ethernet.....	- 40 -
3.9.1. Configuring the Ethernet IP Address	- 40 -
3.9.2. Modify Mac address.....	- 42 -
4. Extended Peripheral Interface.....	- 44 -
4.1. MY-WiredCom.....	- 44 -
4.1.1. RS485 Test	- 44 -
4.1.2. CAN Test.....	- 46 -





4.2. Wi-Fi	- 48 -
4.2.1. Wi-Fi Test	- 48 -
4.2.2. STA mode to connect WiFi hotspot.....	- 48 -
4.3. 4G /5G	- 50 -
4.3.1. View VID and PID	- 50 -
4.3.2. View kernel recognition module.....	- 51 -
4.3.3. Initial testing with AT commands.....	- 51 -
4.3.4. ppp dialing test.....	- 53 -
4.3.5. Ping Test.....	- 58 -
5. Web Applications.....	- 59 -
5.1. PING	- 59 -
5.1.1. Wiring and information output.....	- 59 -
5.1.2. Ping Test.....	- 59 -
5.2. SSH	- 61 -
5.3. SCP.....	- 63 -
5.3.1. Copy files from remote to local	- 64 -
5.3.2. Copy files from local to remote	- 64 -
5.4. TFTP.....	- 64 -
5.4.1. Installing the TFTP server	- 65 -
5.5. DHCP.....	- 66 -
5.6. Iptables.....	- 67 -
5.6.1. Configure the development board iptables	- 67 -
5.6.2. ping Test.....	- 68 -
5.7. iperf3.....	- 69 -
5.7.1. Testing TCP Performance	- 69 -
5.7.2. Testing UDP performance	- 71 -
6. Graphics System.....	- 75 -
6.1. QT.....	- 75 -
6.1.1. Get information about qt	- 75 -
6.1.2. Introduction to the QT runtime environment.....	- 75 -
6.1.3. Start Qt program	- 79 -





7. Multimedia Applications.....	- 80 -
7.1. Video Playback.....	- 80 -
7.1.1. xplayerdemo	- 80 -
7.1.2. Play Video	- 83 -
7.2. Audio.....	- 84 -
7.2.1. Debugging tools.....	- 84 -
8. System Tools	- 87 -
8.1. Compression and decompression tools.....	- 87 -
8.1.1. tar.....	- 87 -
8.1.2. gzip.....	- 88 -
8.2. File System Tools	- 89 -
8.2.1. mount	- 89 -
8.3. Disk Management Tools	- 90 -
8.3.1. fdisk.....	- 90 -
8.3.2. dd	- 91 -
8.3.3. du	- 91 -
8.3.4. df	- 92 -
8.4. Process Management Tools.....	- 93 -
8.4.1. ps.....	- 94 -
8.4.2. top	- 96 -
8.4.3. kill.....	- 97 -
9. Development Support	- 100 -
9.1. Development Languages.....	- 100 -
9.1.1. SHELL	- 100 -
9.1.2. C/C++	- 100 -
9.1.3. Python	- 103 -
9.2. Database.....	- 104 -
9.2.1. System SQLite.....	- 105 -
9.3. Qt Application Localization.....	- 106 -
9.3.1. Multilingual	- 106 -
9.3.2. Fonts	- 106 -





9.3.3. Soft Keyboard	- 107 -
10. References	- 110 -
11. Appendix A.....	- 111 -
11.1. Warranty & Technical Support Services	- 111 -



1. Overview

The Linux Software Evaluation Guide is used to describe the testing procedures and evaluation methods for core and peripheral resources under the open source Linux system running on MYIR's development boards. This document can be used as a pre-evaluation guide or as a test guide for general-purpose system development.

1.1. Hardware Resources

MYIR's MYD-YT113X board consists of the core board MYC-YT113X and the backplane MYB-YT113X, which is soldered to the backplane using stamped holes. In addition, MYIR provides a rich set of software resources and documentation. For detailed configuration parameters of the hardware part, please refer to the *"MYD-YT113X Product Manual"*. Some accessories are also available for the user to use during the evaluation and testing process, see the following list.

Table 1-1. Optional Modules

Accessories	Interface	Description
LCD	Ivds	MY-TFT070CV2 (7Inches LCD with Capacitive Touch Panel) https://www.myirtech.com/list.asp?id=634
Expansion Board	Raspberry Pi	MY-WiredCom: https://www.myirtech.com/list.asp?id=665

1.2. Software Resources

MYD-YT113X development board BSP is based on Allwinner's official open source community version of Linux BSP ported and modified, the system image is built using the buildroot project. bootloader, Kernel and various parts of the file system software resources are all open in the form of source code, please see *"MYD-YT113X_SDK Release Notes"*. The development board is shipped with the image burned according to the core board model, you just need to power it up and use it.

1.3. Documents



The SDK contains different types of documents and manuals, such as release notes, evaluation guides, development guides, application notes, and frequently asked questions, depending on the user's stage of using the development board. The specific documentation list is described in Table 2-4 of the *"MYD-YT113X_SDK Release Notes"*.

1.4. Preparation

Before you start evaluating the development board software, you need to do some necessary preparation and configure some basic environment for the development board, including proper hardware wiring, configuring the debug serial port, setting up the startup and other steps. For detailed steps, please refer to the *"MYD-YT113X QSG"*.

The next section focuses on how to evaluate and test the hardware resources and interfaces of the system as well as the software functionality. The tests are performed mainly with the help of some common tools and commands under Linux, as well as with applications that you have developed yourself. The software evaluation guide is divided into several parts, including: core resources, peripheral resources, network applications, multimedia applications, development support applications, system tools, and other major categories. The subsequent chapters will provide a comprehensive explanation of each section and describe in detail the specific evaluation methods and steps for each section.



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

2.1.1. View the CPU Information

The CPU provider and parameter information can be read from the "/proc/cpuinfo" file.

```
[root@myir:/]# cat /proc/cpuinfo
processor       : 0
model name     : ARMv7 Processor rev 5 (v7l)
BogoMIPS      : 48.00
Features      : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idiv
t vfpd32 lpae
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x0
CPU part       : 0xc07
CPU revision   : 5

processor       : 1
model name     : ARMv7 Processor rev 5 (v7l)
BogoMIPS      : 48.00
Features      : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idiv
t vfpd32 lpae
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x0
CPU part       : 0xc07
```



```
CPU revision      : 5
Hardware          : Generic DT based system
Revision          : 0000
Serial            : 0000000000000000
```

- 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
- modelname: 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)

2.1.2. View CPU Utilization

The CPU utilization can be viewed by performing the following operations on the T113 series chip:

```
[root@myir:/]# top
Mem: 38180K used, 68796K free, 84K shrd, 1760K buff, 9704K cached
CPU:  4% usr  0% sys  0% nic 95% idle  0% io  0% irq  0% sirq
Load average: 0.64 0.20 0.07 1/93 1954

  PID  PPID  USER    STAT   VSZ %VSZ %CPU COMMAND
 1954  1468  root     R      2152  2%   2% top
   12    2  root     SW        0  0%   2% [rcuc/0]
 1476    1  root     S     30832 29%   0% adbd
 1953  1483  root     S      4848  5%   0% sleep 1
 1468    1  root     S      2940  3%   0% -sh
 1483    1  root     S      2688  3%   0% {adb_conf.sh} /bin/sh /etc/adb_co
nf.sh start
 1368    1  root     S      2392  2%   0% dbus-daemon --system
    1    0  root     S      2152  2%   0% init
 1350    1  root     S      2152  2%   0% /sbin/klogd -n
 1346    1  root     S      2152  2%   0% /sbin/syslogd -n
 1402    1  root     S      2152  2%   0% udhcpd -R -n -p /var/run/udhcp
c.eth0.pid -i eth0
 1420    1  root     S      2152  2%   0% /usr/sbin/telnetd -F
 1409    1  root     S      1868  2%   0% /usr/sbin/dropbear -R
```



1426	1	root	S	1740	2%	0%	/usr/sbin/tftpd -c -l -s /var/lib/tf
tpboot							
1140	2	root	SW	0	0%	0%	[irq/45-sunxi-gp]
1393	2	root	SW	0	0%	0%	[irq/172-usb_id]
979	2	root	SW	0	0%	0%	[irq/49-dispaly]
38	2	root	IW	0	0%	0%	[kworker/0:1-eve]
9	2	root	SW	0	0%	0%	[ksoftirqd/0]
20	2	root	SW	0	0%	0%	[ksoftirqd/1]
1064	2	root	SW	0	0%	0%	[irq/58-ehci_hcd]
19	2	root	SW	0	0%	0%	[rcuc/1]
10	2	root	IW	0	0%	0%	[rcu_preempt]
39	2	root	IW	0	0%	0%	[kworker/1:1-eve]
1131	2	root	SW	0	0%	0%	[irq/40-sunxi_tp]
28	2	root	IW	0	0%	0%	[kworker/u4:2-ev]
1259	2	root	SW	0	0%	0%	[irq/43-sunxi-i2]
751	2	root	IW	0	0%	0%	[kworker/u4:4-ev]
1032	2	root	SW	0	0%	0%	[irq/37-4500000.]
1324	2	root	IW<	0	0%	0%	[kworker/1:1H-kb]
7	2	root	IW	0	0%	0%	[kworker/u4:0-ev]
352	2	root	IW	0	0%	0%	[kworker/u4:3-ev]
1175	2	root	SW	0	0%	0%	[irq/38-mmc2]
1176	2	root	SW	0	0%	0%	[irq/38-s-mmc2]
1246	2	root	IW<	0	0%	0%	[kworker/0:1H-mm]

- %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
- %irq: The number of hard interrupts processed by the CPU
- %sirq: The number of soft interrupts processed by the CPU

2.1.3. Gets CPU Temperature



CPU built-in temperature sensor as CPU temperature acquisition, you can easily obtain the internal temperature of the CPU.

```
[root@myir:/]# cat /sys/class/thermal/thermal_zone0/temp
43282
```

The number displayed above is one thousandth of a degree, divided by 1000 is the current temperature value.

```
[root@myir:/]# echo "scale=5000;4*a(1)" | bc -l -q &
[1] 2177
3.14159265358979323846264338327950288419716939937510582097494459230
7816406286208998628034825342117067982148086513282306647093844609550
5822317253594081284811174502841027019385211055596446229489549303819
6442881097566593344612847564823378678316527120190914564856692346034
8610454326648213393607260249141273724587006606315588174881520920962
82925
.....
[1]+  Done                  echo "scale=5000;4*a(1)" | bc -l -q
```

The above command will calculate the PI in the background and is accurate to 5000 decimal places. The calculation process takes some time. At this point, we can check the change of CPU utilization with the top command, as follows:

```
[root@myir:/]# top
Mem: 39348K used, 67628K free, 88K shrd, 1832K buff, 9816K cached
CPU:  84% usr   1% sys   0% nic  45% idle   0% io   0% irq   0% sirq
Load average: 0.88 0.65 0.33 2/91 3809
  PID  PPID  USER      STAT  VSZ %VSZ %CPU COMMAND
 3654  1468  root       R     1516  1%  50% bc -l -q
```

After about 3 minutes, the PI results were calculated. During this period CPU usage reaches 100% and no abnormalities occur, indicating that the CPU stress test passes. You can also continue to increase the exact value, which can further increase the test pressure.

2.2. Memory



MYD-YT113X memory is 128MB version, the system will divide the memory into device memory (CMA) and system memory (MEM). Device memory is a contiguous section of space for the driver to use, and system memory is allocated space for the user state.

2.2.1. View memory information

To read the memory parameters of the system, you can use the file `"/proc/meminfo"` to get.

```
[root@myir:/]# cat /proc/meminfo
MemTotal:      106976 kB
MemFree:       67332 kB
MemAvailable:  78668 kB
Buffers:       2240 kB
Cached:        9520 kB
SwapCached:    0 kB
Active:        10988 kB
Inactive:      3188 kB
Active(anon):  2464 kB
Inactive(anon): 32 kB
Active(file):  8524 kB
Inactive(file): 3156 kB
Unevictable:   0 kB
Mlocked:       0 kB
SwapTotal:     0 kB
SwapFree:      0 kB
Dirty:         0 kB
Writeback:     0 kB
AnonPages:     2404 kB
Mapped:        4560 kB
Shmem:         84 kB
KReclaimable:  4788 kB
Slab:          15196 kB
SReclaimable:  4788 kB
```



SUnreclaim: 10408 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

2.2.2. Get memory usage

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

```
[root@myir:/]# free -m
```

	total	used	free	shared	buffers	cached
Mem:	104	38	65	0	2	9
-/+ buffers/cache:		27	77			
Swap:	0	0	0			

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

2.2.3. 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 60MByte, the test method is as follows:

```
[root@myir:/]# memtester 60M 1
```

memtester version 4.3.0 (32-bit)
Copyright (C) 2001-2012 Charles Cazabon.
Licensed under the GNU General Public License version 2 (only).




```

pagesize is 4096
pagesizemask is 0xfffff000
want 60MB (62914560 bytes)
got 60MB (62914560 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
  8-bit Writes       : ok
  16-bit Writes      : ok

Done.

```

2.3. eMMC and nand test

This section focuses on testing of eMMC and nand for development boards configured with eMMC and nand memory. eMMC is a data storage device that includes a MultiMediaCard (MMC) interface, a NAND component. Its cost, small size, Flash technology independence and high data throughput make it ideal for embedded products.

2.3.1. eMMC test



2.3.1.1. View eMMC capacity

The fdisk-l command allows you to query the eMMC partition information and capacity.

```
[root@myir:/]# fdisk -l
Found valid GPT with protective MBR; using GPT

Disk /dev/mmcblk0: 7634944 sectors, 3728M
Logical sector size: 512
Disk identifier (GUID): ab6f3888-569a-4926-9668-80941dcb40bc
Partition table holds up to 8 entries
First usable sector is 73728, last usable sector is 7634910
```

Number	Start (sector)	End (sector)	Size	Name
1	73728	114687	20.0M	boot-resource
2	114688	116735	1024K	env
3	116736	118783	1024K	env-redund
4	118784	180223	30.0M	boot
5	180224	2277375	1024M	rootfs
6	2277376	2279423	1024K	dsp0
7	2279424	2312191	16.0M	private
8	2312192	7634910	2598M	UDISK

2.3.1.2. View eMMC partition information

The df command allows you to query the eMMC partition information, usage, mount directory and other information.

```
[root@myir:/]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       992M  232M  745M  24% /
tmpfs           53M   52K   53M   1% /tmp
tmpfs           53M   32K   53M   1% /run
devtmpfs        43M    0    43M   0% /dev
```

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



2.3.1.3. eMMC Performance Test

The performance test mainly tests the reading and writing speed of files by eMMC in Linux system. The test is generally combined with the dual command of "time" and "dd" .

● Write Performance

```
[root@myir:/]# time dd if=/dev/zero of=tempfile bs=1M count=100 conv=fdat
async
100+0 records in
100+0 records out
104857600 bytes (105 MB, 100 MiB) copied, 8.11714 s, 12.9 MB/s

real    0m8.132s
user    0m0.000s
sys     0m1.564s
```

Here we test that the write disk speed is 12.9MB/s.

● Read Performance

In embedded system, it is often necessary to test the system's performance of reading. If you want to test reading files directly from disk, you need to ignore the impact of cache. In this case, you can specify the parameter "iflag = direct, Nonblock" .

```
[root@myir:/]# time dd if=tempfile of=/dev/null bs=1M count=100 iflag=direc
t,nonblock

100+0 records in
100+0 records out
104857600 bytes (105 MB, 100 MiB) copied, 2.33601 s, 44.9 MB/s

real    0m2.347s
user    0m0.000s
sys     0m0.077s
```

The test results show that reading speed without cache is 44.9MB/s.



2.3.2. Nand Test

2.3.2.1. View nand capacity and partition information

The df command allows you to query the nand partition information, capacity, usage and mount directory information.

```
[root@myir:~]# df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
ubi0_5	147M	49M	99M	33%	/
tmpfs	53M	36K	53M	1%	/tmp
tmpfs	53M	28K	53M	1%	/run
devtmpfs	48M	0	48M	0%	/dev

2.3.2.2. Nand Performance Test

● Write Performance

The performance test is mainly to test the speed of reading and writing files under linux system, generally combined with "time" and "dd" commands to test.

```
[root@myir:~]# time dd if=/dev/zero of=write_file bs=5M count=1 conv=fsync
1+0 records in
1+0 records out
5242880 bytes (5.2 MB, 5.0 MiB) copied, 0.878978 s, 6.0 MB/s
real    0m 0.89s
user    0m 0.00s
sys     0m 0.86s
```

Here we can get a nand write speed of 6.0 MB/s.

● Read Performance

In embedded systems, it is often necessary to test system file read and write performance and ignore the impact of cache when reading files. First, execute the following command to clear cache.

```
[root@myir:~]# sync; echo 3 > /proc/sys/vm/drop_caches
```

In embedded systems, it is often necessary to test system file read and write performance.

```
[root@myir:~]# time dd if=write_file of=read_file bs=5M count=1
```



```
1+0 records in
1+0 records out
5242880 bytes (5.2 MB, 5.0 MiB) copied, 0.0556429 s, 44.2 MB/s
real    0m 0.06s
user    0m 0.00s
sys     0m 0.05s
```

Here we can get the nand read speed of 44.2 MB/s.

2.4. RTC

The RTC (Real-time clock) itself is a clock used to record the real time, when the software system is shut down, the system time is kept and continues to be timed, and the time is synchronized into the software system after the system is turned back on. MYD-YT113X has internal RTC, and external RTC (RX8025), if the actual product is not very demanding on RTC power consumption, RTC testing is usually done with the `hwclock` and `date` commands commonly used in Linux systems, the following test will write the system time into the RTC, read the RTC time and set it as the system time and perform the time power down Hold test.

2.4.1. View the system RTC device

```
[root@myir:/]# ls /dev/rtc* -al
crw-rw---- 1 root root 253, 0 Jan  1 00:00 /dev/rtc0
```

2.4.2. Set system time

After setting the system time to Mon Feb 7 09:28:00 UTC 2023:

```
[root@myir:/]# date 020709282023.00
Mon Feb  7 09:28:00 UTC 2023
```

● Write system time to RTC

Write the system time set by the `date` command in the previous step to the RTC device:

```
[root@myir:/]# hwclock -w
```



- **Read RTC time and set to system time**

```
[root@myir:~]# hwclock -r
Mon Feb  7 09:28:24 2023  0.000000 second
```

- **Power down hold RTC time**

Turn the development board off and disconnect the power, and after a few minutes or so, power it back on. Check the RTC time and system time:

```
[root@myir:~]# hwclock -r
Mon Feb  7 09:32:14 2023  0.000000 second
```

The RTC time and system time viewed after reboot increased by about 20 minutes compared to the previous setting, indicating that the RTC is working properly. If you need to test the accuracy of RTC in detail, you can extend the power-off time for example 24 hours to test the difference between RTC time and standard time.

- **Synchronize system time with RTC time**

```
[root@myir:~]# hwclock -s
[root@myir:~]# date
Mon Feb  7 09:35:23 UTC 2023
```

If you add the hwclock-s command to the boot script, you can ensure that the system time and RTC time are synchronized every time you boot.

2.5. Watchdog

Linux kernel contains Watchdog subsystem, the hardware design process can generally use the internal watchdog timer of the chip or use external watchdog chip to implement the Watchdog function, used to monitor the operation of the system. MYD-YT113X chip has an internal watchdog, this chapter will explain the test method of watchdog under linux.

2.5.1. Application testing watchdog

The following application is used to set the watchdog timeout and dog feeding time. (The source code for the tested application is in the "04_Sources/Example/" directory)



● Set watchdog timeout time

The timeout is implemented through ioctl, the specific command is: WDIOCG_SETTIMEOUT, one parameter is required, i.e. timeout timeout. usage example:

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

The above is the reference code to set the current timeout time of the watchdog. where fd is the file handle of the watchdog device.

● Watchdog application testing

Compile the production executable watchdog and copy it to the development board with the following command:

```
[root@myir:/]# ./watchdog
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 application, timeout is 4s, feed the dog every 1s interval:

```
[root@myir:/]# ./watchdog 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 more than 4s, the required 4s feed time is exceeded and the board will reboot.

2.6. PMIC

This section demonstrates the Suspend function of Linux power management, which allows the development board to sleep and wake up by external events. The MYD-YT113X supports both freeze and me.



2.6.1. View the current supported modes of the development board

```
[root@myir:~]# cat /sys/power/state
freeze mem
```

2.6.2. Methods for writing in user space

```
[root@myir:~]# echo "freeze" > /sys/power/state
[root@myir:~]# echo "mem" > /sys/power/state
```

● mem hibernation mode

After entering the hibernate command, the development board hibernates, the debug serial port can no longer input, at this time the system and device state is saved to memory (in self-refresh mode, has retained its content), all devices enter low-power mode.

```
[root@myir:~]# echo "mem" > /sys/power/state
```

● freeze hibernation mode

After entering the hibernate command, the development board hibernates, the debug serial port can no longer input, at this time the user space is frozen, all I/O devices enter the low-power state, and the processor enters the idle state.

```
[root@myir:~]# echo "freeze" > /sys/power/state
```

At this point, press user button S1 to wake up the system:

```
[root@myir:~]# [ 1558.783587] libphy: gmac1: probed
** 55 printk messages dropped **
[ 154.942339] 001: [DMIC]Enter sunxi_dmic_resume
```

At this point the debug serial port can be re-entered.



3. Basic Peripheral Interface

3.1. GPIO

GPIO testing is achieved through the file system sysfs interface, the following content to the PD20 as an example to illustrate the use of GPIO process.

Calculate the value of the corresponding pin of $gpio = (n-1)*32+x$ (set A as 1, B as 2, and so on, H corresponds to 8) (x in PH2 means 2) e.g.

PH2 corresponds to the value: $(8-1) * 32 + 2 = 226$ PH3 corresponds to the value: $(8-1) * 32 + 3 = 227$

Designing the expansion interface of the base board PD20

$PD20 = (4 - 1) * 32 + 20 = 116$

echo 116 > /sys/class/gpio/export

echo out > /sys/class/gpio116/direction

echo 1 > xxx/value (High level)

echo 0 > xxx/value (Low Level)

Parameters:

- function: Multiplexing
- data: Level data (1 for high level; 0 for low level)
- plevel: Drive Capability
- pull: Pull up and down situation

3.1.1. Exporting GPIO

```
[root@myir:~]# echo 117 > /sys/class/gpio/export
```

After a successful export, the directory

"/sys/class/gpio/" directory will generate the directory gpio117

3.1.2. Set/View GPIO Direction

● Set input

```
[root@myir:~]# echo "in" > /sys/class/gpio/gpio117/direction
```

● Set output

```
[root@myir:~]# echo "out" > /sys/class/gpio/gpio117/direction
```



- **View gpio directions**

```
[root@myir:/]# cat /sys/class/gpio/gpio117/direction
out
```

Returns "in" for input and "out" for output.

3.1.3. Set/view GPIO values

- **Set output low**

```
[root@myir:/]# echo "0" > /sys/class/gpio/gpio117/value
```

- **Set output high**

```
[root@myir:/]# echo "1" > /sys/class/gpio/gpio117/value
```

- **View gpio values**

```
[root@myir:/]# cat /sys/class/gpio/gpio117/value
1
```

You can see PD20 output high level, you can use a multimeter to measure the PD20 pin of J2 expansion IO, you can see the voltage is about 3.3V.

3.2. LED

The Linux system provides a separate subsystem to facilitate the operation of LED devices from user space, which provides interfaces to the LED devices in the form of files. These interfaces are located in the "/sys/class/leds" directory. In the hardware resources list, we have listed all the LEDs on the board, and the LEDs are tested with the command read/write sysfs. The following commands are generic and are a common way to manipulate the LEDs.

3.2.1. The directory to operate LEDs is "/sys/class/leds"

```
[root@myir:/sys/class/leds]# ls
led-blue  led-green
```

The duty cycle of the heartbeat light can be changed by writing different values to "/sys/class/leds/bule/brightness".

3.2.2. Heartbeat light led-blue as an example to test LED



- **Turn off the heartbeat light**

```
[root@myir:/sys/class/leds]# echo none > /sys/class/leds/led-blue/trigger
```

Turn off the heartbeat light, then you can individually led off and on operation.

- **Turn off LED**

```
[root@myir:/sys/class/leds]# echo 1 > /sys/class/leds/led-blue/brightness
```

- **Turn on LED**

```
[root@myir:/sys/class/leds]# echo 0 > /sys/class/leds/led-blue/brightness
```

- **Turn on LED trigger mode**

When "heartbeat" mode is turned on, the LED flashes at 1Hz by default, with a duty cycle of 50%:

```
[root@myir:/]# echo heartbeat > /sys/class/leds/led-blue/trigger
```

3.3. Key

The Linux The "/dev/input/eventx" device can be used to easily debug input devices such as mouse, keyboard, and touchpad. This section focuses on testing the key, using the hexdump command and the dmesg command to see if the keys are responding.

S1 User key

S2 Reset key

3.3.1. Device Tree Configuration Information

Open the companion device tree file "auto-t113-linux/device/config/chips/t113/configs/myir_xxx/board.dts" and you can see the nodes of the key S1 User key: (xxx stands for different configuration)

```
gpio-keys {
    compatible = "gpio-keys";
    status = "okay";
    vol-down-key {
        gpios = <&pio PE 1 GPIO_ACTIVE_LOW>;
        linux,code = <114>;
        label = "user key";
    }
}
```



```

        debounce-interval = <10>;
        wakeup-source = <0x1>;
    };
};

```

3.3.2. Key Test

- View the corresponding input event information

```

[root@myir:~]# cat /proc/bus/input/devices
I: Bus=0019 Vendor=0001 Product=0001 Version=0100
N: Name="sunxi-keyboard"
P: Phys=sunxikbd/input0
S: Sysfs=/devices/virtual/input/input0
U: Uniq=
H: Handlers=kbd event0
B: PROP=0
B: EV=3
B: KEY=1000 800 c0000 0 0 10000000

I: Bus=0019 Vendor=0001 Product=0001 Version=0100
N: Name="sunxi-tpadc"
P: Phys=
S: Sysfs=/devices/virtual/input/input1
U: Uniq=
H: Handlers=kbd event1
B: PROP=0
B: EV=100003
B: KEY=200000 0 0 0 0 0 0 0 8c0000 0 0 0

I: Bus=0019 Vendor=0001 Product=0001 Version=0100
N: Name="sunxi-gpadc0"
P: Phys=sunxigpadc0/input0
S: Sysfs=/devices/virtual/input/input2
U: Uniq=
H: Handlers=event2

```



```

B: PROP=0
B: EV=100003
B: KEY=0

I: Bus=0019 Vendor=0001 Product=0001 Version=0100
N: Name="sunxi-gpadc1"
P: Phys=sunxigpadc1/input0
S: Sysfs=/devices/virtual/input/input3
U: Uniq=
H: Handlers=event3
B: PROP=0
B: EV=11
B: MSC=10

I: Bus=0019 Vendor=0001 Product=0001 Version=0100
N: Name="sunxi-ir"
P: Phys=sunxi-ir/input0
S: Sysfs=/devices/platform/soc@3000000/7040000.s_cir/rc/rc0/s_cir_rx
U: Uniq=
H: Handlers=kbd event4
B: PROP=20
B: EV=100017
B: KEY=2
B: REL=3
B: MSC=10

I: Bus=0000 Vendor=0000 Product=0000 Version=0000
N: Name="audiocodec sunxi Audio Jack"
P: Phys=ALSA
S: Sysfs=/devices/platform/soc@3000000/2030340.sound/sound/card0/input5
U: Uniq=
H: Handlers=kbd event5
B: PROP=0

```



```
B: EV=23
B: KEY=40 0 0 0 0 0 0 0 0 0 0 4 0 0 0 c0000 0 0 0
B: SW=14

I: Bus=0019 Vendor=0001 Product=0001 Version=0100
N: Name="gpio-keys"
P: Phys=gpio-keys/input0
S: Sysfs=/devices/platform/gpio-keys/input/input7
U: Uniq=
H: Handlers=kbd event6
B: PROP=0
B: EV=3
B: KEY=40000 0 0 0
```

From the above, we can see that the corresponding device event for gpio-keys is "event6".

● evtest test key message

Execute the following command and operate key S1, the serial terminal will print out the following message:

```
[root@myir:]/# evtest
No device specified, trying to scan all of /dev/input/event*
Available devices:
/dev/input/event0:      sunxi-keyboard
/dev/input/event1:      sunxi-tpadc
/dev/input/event2:      sunxi-gpadc0
/dev/input/event3:      sunxi-gpadc1
/dev/input/event4:      sunxi-ir
/dev/input/event5:      audiocodec sunxi Audio Jack
/dev/input/event6:      gpio-keys
Select the device event number [0-6]: 6
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 114 (KEY_VOLUMEDOWN)

Properties:

Testing ... (interrupt to exit)

Event: time 1245.584992, type 1 (EV_KEY), code 114 (KEY_VOLUMEDOWN), value 1

Event: time 1245.584992, ----- SYN_REPORT -----

Event: time 1245.724901, type 1 (EV_KEY), code 114 (KEY_VOLUMEDOWN), value 0

Event: time 1245.724901, ----- SYN_REPORT -----

Event: time 1246.034910, type 1 (EV_KEY), code 114 (KEY_VOLUMEDOWN), value 1

Event: time 1246.034910, ----- SYN_REPORT -----

Event: time 1246.174891, type 1 (EV_KEY), code 114 (KEY_VOLUMEDOWN), value 0

Event: time 1246.174891, ----- SYN_REPORT -----

Event: time 1246.324911, type 1 (EV_KEY), code 114 (KEY_VOLUMEDOWN), value 1

Event: time 1246.324911, ----- SYN_REPORT -----

Event: time 1246.454896, type 1 (EV_KEY), code 114 (KEY_VOLUMEDOWN), value 0

Event: time 1246.454896, ----- SYN_REPORT -----

Every time "S1" is pressed the current terminal prints the current event code value, i.e. the key is normal.

3.4. USB

This section verifies the feasibility of the USB Host driver through relevant commands or hot-plugging, USB HUB, and implements the function of reading and writing USB drives, usb enumeration.



3.4.1. View print information for inserted usb

● Viewing USB Device Information

Connect the USB flash drive to the USB Host interface (J11) of the development board, the message will be as follows:

```
[root@myir:~]# create /dev/sda
create /dev/sda1
```

From the above information, we can conclude that the device to be mounted is sda1, and it will be automatically mounted under the directory **"/mnt/usb/sda1/"**.

3.4.2. U disk mount read/write

● Read files

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

```
[root@myir:/]# cd /mnt/usb/sda1/
[root@myir:/mnt/usb/sda1:~]# ls
test.txt
[root@myir:/mnt/usb/sda1:~]# cat test.txt
hello!!!!
```

● Write file

```
[root@myir:/mnt/usb/sda1:~]# touch test
[root@myir:/mnt/usb/sda1:~]# echo " helloworld!!!" > test
[root@myir:/mnt/usb/sda1:~]# cp test /mnt
[root@myir:/mnt/usb/sda1:~]# cat /mnt/test
helloworld!!!
```

After writing the file, you need to execute the sync command to ensure that the data is completely written to the USB flash drive before you can uninstall the device.

3.4.3. Uninstall USB drive

● Uninstall operation

Exit the mount directory when unmounting the USB drive

```
[root@myir:/]#umount /mnt/usb/sda1
```



3.5. Micro SD

Micro SD Card, formerly known as Trans-flash Card (TF Card), is an extremely small flash memory card. micro SD card is smaller than the standard SD card and is the smallest SD card among SD card types. Although the form factor and interface shape of the Micro SD card is different from the original SD card, the interface specification remains the same, ensuring compatibility. The Micro SD card generally has 9 pins on the back, contains 4 data lines, and supports 1bit/4bit data transfer width. MYD-YT113X supports three 1bit or 4bit SDMMC interfaces, and the development board uses SDMMC0 to connect to the Micro SD card.

3.5.1. View TF card capacity

The "fdisk-l" command can query the TF card partition information and capacity:

```
[root@myir:]/# fdisk -l
Found valid GPT with protective MBR; using GPT

Disk /dev/sda: 30930944 sectors, 2815M
Logical sector size: 512
Disk identifier (GUID): d05e42e5-9ac6-4bc7-8c63-6d7c228073cb
Partition table holds up to 128 entries
First usable sector is 34, last usable sector is 30930910
```

Number	Start (sector)	End (sector)	Size	Name
1	2048	30928895	14.7G	Basic data partitionm

3.5.2. Viewing TF card partition information

The "df" command can query the TF card partition information, usage, mount directory and other information:

EMMC:

```
[root@myir:]/# df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/root	992M	332M	645M	34%	/
tmpfs	53M	112K	53M	1%	/tmp
tmpfs	53M	28K	53M	1%	/run



```
devtmpfs          43M      0   43M    0% /dev
/dev/sda1         15G    32K   15G    1% /mnt/usb/sda1
```

Nand:

```
[root@myir:/]# df -h
Filesystem      Size  Used Avail Use% Mounted on
ubi0_5         147M   53M   95M   36% /
tmpfs           53M   104K   53M    1% /tmp
tmpfs           53M    28K   53M    1% /run
devtmpfs        48M      0   48M    0% /dev
/dev/sda1       15G    32K   15G    1% /mnt/usb/sda1
```

- devtmpfs : For system creation dev
- tmpfs : In-memory virtual file system, mounted to a different directory
- /dev/sda1: TF Card Catalog

3.5.3. TF card performance test

Performance test is to test the read/write speed of TF card under linux system, usually combined with time and dd commands. Mount the TF card partition to be tested, here take the last partition "/dev/sda1" as an example, the mount directory is "/mnt/usb/sda1".

● Write file test

```
[root@myir:/mnt/lost+found]# ttime dd if=/dev/zero of=test_write bs=1M count=100 conv=fsync
100+0 records in
100+0 records out
104857600 bytes (105 MB, 100 MiB) copied, 13.7005 s, 7.7 MB/s

real    0m13.712s
user    0m0.000s
sys     0m1.891s
```

The write disk speed tested here is 7.7MB/s.

● Read file test

```
[root@myir:/mnt/lost+found]# time dd if=test_write of=/mnt/usb/sda1/test_read bs=1M count=100
```



```
100+0 records in
100+0 records out
104857600 bytes (105 MB, 100 MiB) copied, 19.1825 s, 5.5 MB/s

real    0m19.197s
user    0m0.000s
sys     0m2.419s
```

It is known that the speed of reading data directly from SD card is 5.5MB/s.





3.6. ADC

The GPADC is a 12bit sampling precision analog-to-digital conversion module with an analog input range specified by the platform (1.8V for the T113X platform), with only one GPADC and four TPADC channels. the ADC test is implemented through the file system sysfs interface.

3.6.1. Read GPADC test values

First we select the gpadc0 channel and use the following command to read the default values directly.

```
[root@myir:~]# cat /sys/class/gpadc/data  
1583
```

The data obtained is 1583, which is converted to a voltage value of:
 $1583/1000=1.583v$

Connect the gpadc0 channel to 0v and check the data again, and get the voltage value of 0v.

```
[root@myir:~]# cat /sys/class/gpadc/data  
0
```

Connect the gpadc0 channel to 1.8v and check the data again, and get the voltage value of 1.744v.

```
[root@myir:/]# cat /sys/class/gpadc/data  
1744
```

3.6.2. Read TPADC test values

There are four tpadc channels, here we choose tpadc0 for example, first enable tpadc0 channel.

```
[root@myir:/]# echo 0 > /sys/devices/virtual/input/input1/channel_tpadc
```

Then read the default value and get a voltage of $1709/1000 = 1.709v$.

```
[root@myir:/]# cat /sys/devices/virtual/input/input1/tpadc  
1709
```

Connect TPADC0 to 0v and get the voltage close to 0v.



```
[root@myir:~]# cat /sys/devices/virtual/input/input1/tpadc
2
```

Connect TPADC0 to 1.8v to get 1.752v.

```
[root@myir:~]# cat /sys/devices/virtual/input/input1/tpadc
1752
```

If you need to use other tpadc then enable the channel first and then measure it. The other three tpadc enable commands are as follows, and the measurement method is the same as tpadc0.

```
[root@myir:~]# echo 1 > /sys/devices/virtual/input/input1/channel_tpadc
[root@myir:~]# echo 2 > /sys/devices/virtual/input/input1/channel_tpadc
[root@myir:~]# echo 3 > /sys/devices/virtual/input/input1/channel_tpadc
```

3.7. Display

This module consists of a display engine (DE) and various types of controllers (tcon). The input layers (layers) are processed in the DE for display and output to the display device through one or more interfaces, so that the layers rendered by many applications can be composited and presented to the user on the display. Each independent unit of DE has 1-4 channels (typically, de0 has 4 and de1 has 2), and each channel can process and accept 4 layers of the same format at the same time. sunxi platform has a video channel and a UI channel. The video channel is powerful enough to support both YUV format and RGB layers, while the UI channel supports only RGB layers. In short, the main functions of the display module are as follows:

- Support lcd(hv/lvds/cpu/dsi) output
- Support dual display output
- Support multi-layer overlay blending process
- Support multiple display effects processing (alpha, colorkey, image enhancement, brightness/contrast/saturation/chromaticity adjustment)
- Support intelligent backlight adjustment
- Support multiple image data format input (argb, yuv)



- Support for image scaling - Support for screenshots - Support for image conversion

3.7.1. Device Tree Configuration Information

Open the companion device tree file "auto-t113-linux/device/config/chips/t113/configs/myir_XXX" under board.dts and uboot-board.dts, you can see two display schemes. (XXX stands for different board configurations)

```
#include "sun8iw20p1.dtsi"
#include "myir-t113-lvds.dtsi"
//#include "myir-t113-lvds-dual.dtsi"
```

3.7.2. Display Solution Portfolio

- **Default display**

MYD-YT113X default 7" LVDS display.

- **19" dual LVDS display**

Select the 19" dual LVDS display solution and comment the 7" LVDS display solution in the device tree board.dts and uboot-board.dts before compiling. (You cannot select two display schemes at the same time)

```
//#include "myir-t113-lvds.dtsi"
#include "myir-t113-lvds-dual.dtsi"
```

3.8. Touch Panel

The MYD-YT113X development board hardware currently does not support resistive touch, but does support capacitive touch. Capacitive screens are more sensitive in use and rarely cause problems. In addition, capacitive screens do not need to be more accurate. The capacitive screen is highly sensitive because it can accurately identify the position of the finger in contact with the screen according to the principle of capacitive screen. If we click on the software in the use of the phenomenon of not selected, there is generally only one situation: the screen has a problem. The following is a simple test of capacitive screen touch function through evtest command.



3.8.1. evtest

Execute "evtest" in the terminal to enter the test interface. Select the test peripheral as touch screen, here the default is to input interrupt "5", test interface select "5" and press enter to start the test.

```
[root@myir:~]# evtest
No device specified, trying to scan all of /dev/input/event*
Available devices:
/dev/input/event0:      sunxi-keyboard
/dev/input/event1:      sunxi-tpadc
/dev/input/event2:      sunxi-gpadc0
/dev/input/event3:      sunxi-ir
/dev/input/event4:      audiocodec sunxi Audio Jack
/dev/input/event5:      generic ft5x06 (79)
/dev/input/event6:      gpio-keys
Select the device event number [0-6]: 5
Input driver version is 1.0.1
Input device ID: bus 0x18 vendor 0x0 product 0x0 version 0x0
Input device name: "generic ft5x06 (79)"
Supported events:
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 330 (BTN_TOUCH)
  Event type 3 (EV_ABS)
    Event code 0 (ABS_X)
      Value      0
      Min        0
      Max       1023
    Event code 1 (ABS_Y)
      Value      0
      Min        0
      Max       599
    Event code 47 (ABS_MT_SLOT)
      Value      0
```



```

Min      0
Max      4
Event code 53 (ABS_MT_POSITION_X)
Value    0
Min      0
Max      1023
Event code 54 (ABS_MT_POSITION_Y)
Value    0
Min      0
Max      599
Event code 57 (ABS_MT_TRACKING_ID)
Value    0
Min      0
Max      65535

```

Properties:

Property type 1 (INPUT_PROP_DIRECT)

Testing ... (interrupt to exit)

Event: time 205.770226, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID), value 0

Event: time 205.770226, type 3 (EV_ABS), code 53 (ABS_MT_POSITION_X), value 405

Event: time 205.770226, type 3 (EV_ABS), code 54 (ABS_MT_POSITION_Y), value 180

Event: time 205.770226, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 1

Event: time 205.770226, type 3 (EV_ABS), code 0 (ABS_X), value 405

Event: time 205.770226, type 3 (EV_ABS), code 1 (ABS_Y), value 180

3.9. Ethernet

There are many tools for network configuration under Linux, such as net-tools, iproute2, systemd-networkd, network manager and connman, all of which can be selected during system customization.

3.9.1. Configuring the Ethernet IP Address



- **Manually configure the network using ifconfig in the net-tools toolkit**

First, check the network device information by using the ifconfig command as follows:

```
[root@myir:~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 32:A9:D0:70:C2:14
          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:37

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: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)
```

The following describes how to manually configure eth0 with the IP address 192.168.0.100 with the following command:

```
[root@myir:~] # ifconfig eth0 192.168.0.100 netmask 255.255.255.0 up
```

The above command manually configures eth0 with an IP address of 192.168.0.100, a subnet mask of 255.255.255.0, and the default configured broadcast address of 192.168.0.255, and activates it with the up parameter, as follows:

```
[root@myir:~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 32:A9:D0:70:C2:14
          inet addr:192.168.0.100  Bcast:192.168.0.255  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:37
```

● **Manually configure the network using the ip commands in the iproute2 toolkit**

The ifconfig command to set the IP address manually can also be replaced with ip addr and ip link, see <https://wiki.linuxfoundation.org/networking/iproute2> for more information.

```
[root@myir:/]# ip addr flush dev eth0
[root@myir:/]# ip addr add 192.168.0.101/24 brd + dev eth0
[root@myir:/]# ip link set eth0 up
```

If the IP address has been configured before, the IP address configured with ip addr add will become the second ary address, so use ip addr flush to clear the previous address first and then configure and activate it. After completing the configuration, use the ip addr show command to view the eth0 information as follows:

```
[root@myir:/]# ip addr show eth0
3: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast
state DOWN group default qlen 1000
    link/ether 32:a9:d0:70:c2:14 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.101/24 brd 192.168.0.255 scope global eth0
        valid_lft forever preferred_lft forever
```

3.9.2. Modify Mac address

To manually change the Mac address 00:0C:29:36:97:20, the command is as follows:

```
[root@myir:/]# ifconfig eth0 down
[root@myir:/]# ifconfig eth0 hw ether 00:0C:29:36:97:20
[root@myir:/]# ifconfig eth0 up
[root@myir:/]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:0C:29:36:97:20
```



```
inet addr:192.168.0.101 Bcast:192.168.0.255 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:37
```



4. Extended Peripheral Interface

MYD-YT113X development board provides rich peripheral interfaces, in addition to the basic peripheral interfaces, can also be connected to a variety of expansion modules. This makes the development more flexible and convenient for users. The following is the test procedure of several optional modules introduced by MYIR. Users can purchase them according to their needs. For details of optional modules, please refer to Table 1-1 List of optional modules.

4.1. MY-WiredCom

MY-WiredCom module is MYIR's Raspberry Pi peripheral interface form, including RS232/RS485/CAN/ SPI/I2C peripheral interface. The module needs to be purchased by users according to their needs, please refer to Table 1-1 for details of the optional module list. Before testing, users need to connect the module to the J2 interface of the development board.

4.1.1. RS485 Test

The Linux serial device file is generally named `/dev/ttySn` ($n=0,1,2,3,\dots$). n indicates the device number of the serial port on the Linux system, and "ttyAS" is the kernel-defined serial device name. n indicates the device number of the serial port in the Linux system, and "ttyAS" is the name of the serial device already defined by the kernel. This section tests the J2 interface on the MY-WiredCom expansion board as an example, and the numbered device node of the J2 interface is ttyAS4. The test configuration is shown in the following table:

Table 4-1. RS232 Interface Configuration

Test Points	MYD-YT113X	Windows 10
Hardware Interface	RS485	USB-RS485
Device Nodes	ttyAS4	com12
Test Software	rs485_read、rs485_write	sscom

Here, 485A and 485B of the J2 base of the MY-WiredCom module are connected to 485A and 485B of the USB-RS485 converter, respectively..



(The rs485_read、rs485_write test routines are in the "04_Sources/Example/" directory)

● Test board RS485 data collection

First set the windows serial tool sscom to send a string at regular intervals, e.g. 1000ms to send the string "1234567890".

The development board is ready to receive data, execute the following command on top of the development board to receive data.

After the command is executed, the interrupt will enter a blocking state and wait to receive the data from the computer's serial port, and when the data is received from the computer's serial port, the received string "1234567890" will be printed and displayed on the terminal.

```
[root@myir:/]# ./rs485_read -d /dev/ttyAS4 -b 115200
RECV[05]: 0x12 0x34 0x56 0x78 0x90
RECV[05]: 0x12 0x34 0x56 0x78 0x90
RECV[05]: 0x12 0x34 0x56 0x78 0x90
RECV[05]: 0x12 0x34 0x56 0x78 0x90
RECV[05]: 0x12 0x34 0x56 0x78 0x90
RECV[05]: 0x12 0x34 0x56 0x78 0x90
RECV[05]: 0x12 0x34 0x56 0x78 0x90
```

● Test development board RS485 send data

Execute the following command on the development board, it will send " 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0x0c 0x0d 0x0e 0x0f 0x10 0x11 0x12 0x13 " data to uart4, and the data will be received by the serial port tool under windows at this time.

Execute the following commands before sending data from the development board

```
[root@myir:/]# echo 238 > /sys/class/gpio/export
[root@myir:/]# echo out > /sys/class/gpio/gpio238/direction
[root@myir:/]# echo 1 > /sys/class/gpio/gpio238/value
```

Then execute the send command

```
[root@myir:/]# ./rs485_write -d /dev/ttyAS4 -b 115200
```



```
SEND[20]: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0
x0c 0x0d 0x0e 0x0f 0x10 0x11 0x12 0x13
SEND[20]: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0
x0c 0x0d 0x0e 0x0f 0x10 0x11 0x12 0x13
SEND[20]: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0
x0c 0x0d 0x0e 0x0f 0x10 0x11 0x12 0x13
SEND[20]: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0
x0c 0x0d 0x0e 0x0f 0x10 0x11 0x12 0x13
SEND[20]: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0
x0c 0x0d 0x0e 0x0f 0x10 0x11 0x12 0x13
```

The data sent and received on the development board corresponds to the data sent and received by sscom under windows, i.e. the development board RS485 receives data normally.

4.1.2. CAN Test

This section uses the "cansend" and "candump" commands, which are commonly used in Linux systems, to test SocketCAN communication. The test here uses two development boards docked for testing.

Here, the pins "CANH" and "CANL" of the J2 block of MY-WiredCom module are connected to "CANH" and "CANL" of the same type of board.

● Initializing the CAN network interface

Set CAN baud rate, using can requires setting baud rate and turning on CAN network interface first. Refer to the following commands to set the data baud rate of the two development boards to 20KHz respectively and turn on the CAN function: (the highest tested 1M)

```
[root@myir:/]# ifconfig can0 down
[root@myir:/]# ip link set can0 type can bitrate 20000
[root@myir:/]# ip link set can0 up
```

At this point, the can function is turned on.

● Sending data

Set one of the boards to send and use cansend to send data:



```
[root@myir:/]# cansend can0 123#1122334455667788
[root@myir:/]# cansend can0 123#1122334455667788
[root@myir:/]# cansend can0 123#1122334455667788
[root@myir:/]# cansend can0 123#1122334455667788
[root@myir:/]# cansend can0 123#1122334455667788
[root@myir:/]# cansend can0 123#1122334455667788
[root@myir:/]# cansend can0 123#1122334455667788
```

● Receiving data

Set the other board to receive, you can use candump to view the CAN receive data:

```
[root@myir:/]# candump can0 -L
(0000000274.962878) can0 123#1122334455667788
(0000000275.568764) can0 123#1122334455667788
(0000000276.088995) can0 123#1122334455667788
(0000000276.578937) can0 123#1122334455667788
(0000000277.069075) can0 123#1122334455667788
```

● Statistics can0 information

The value of "clock" represents the clock of the CAN device, the value of "drop" represents packet loss, the value of "overrun" represents overflow, and "error" represents bus error.

```
[root@myir:/]# ip -details -statistics link show can0
2: can0: <NOARP,UP,LOWER_UP,ECHO> mtu 16 qdisc pfifo_fast state UP mode
DEFAULT group default qlen 10
    link/can  promiscuity 0 minmtu 0 maxmtu 0
    can state  ERROR-ACTIVE (berr-counter tx 0 rx 0) restart-ms 0
        bitrate 20000 sample-point 0.850
        tq 2500 prop-seg 8 phase-seg1 8 phase-seg2 3 sjw 1
        sun8i-can: tseg1 1..16 tseg2 1..8 sjw 1..4 brp 1..64 brp-inc 1
        clock 24000000
        re-started bus-errors arbit-lost error-warn error-pass bus-off
            0          3          0          1          1          0          nu
mtxqueues 1 numrxqueues 1 gso_max_size 65536 gso_max_segs 65535
```



RX: bytes	packets	errors	dropped	overrun	mcast
72	9	3	0	0	0
TX: bytes	packets	errors	dropped	carrier	collsns
32	4	0	0	0	0

4.2. Wi-Fi

4.2.1. Wi-Fi Test

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

4.2.2. STA mode to connect WiFi hotspot

Try to manually connect to the nearby Wi-Fi hotspot SSID, which is a Wi-Fi hotspot with WPA2 encryption, please configure it yourself.

Ensure wlan0 network device is active.

```
[root@myir:/]# ifconfig wlan0 up
```

- **Scan for WiFi hotspots nearby**

Scan for nearby wifi hotspots and get a list of nearby Wi-Fi hotspots as follow:

```
[root@myir:/]# iw dev wlan0 scan | grep SSID
    SSID: CMCC-u37Z
    SSID: MYIR
    SSID: HUAWEI-400P0Y
    SSID: CMCC-u37Z-5G
    SSID: MYIR_5G
        * SSID List
    SSID: HUAWEI-400P0Y
```

- **wpa_passphrase set the wifi name and password**

```
[root@myir:/]# wpa_passphrase USER PASSWD >> /etc/wpa_supplicant.conf
[root@myir:/usr/lib/ltsp-testsuite/network]# cat /etc/wpa_supplicant.conf
```




```
ctrl_interface=/var/run/wpa_supplicant
ap_scan=1

network={
    ssid="MYIR"
    #psk="MYIR88888888"
    psk=8c66c06459ebe47f74efcebbb3a7ed74b144e63028fb7947dcead44ed
1714c23
}
```

Generate a WPA PSK from the ASCII password of an SSID for encryption operations.

● Turn off the wpa_supplicant process

Before you can use wpa_supplicant to connect and configure WIFI, you need to shut down the wpa_supplicant process first: :

```
[root@myir:]/# killall wpa_supplicant
```

● Initialize wpa_supplicant

wpa_supplicant is a tool for connecting and configuring WIFI. Its main job is to interact with the driver via socket and report data to the user layer, which can also send commands to wpa_supplicant via socket to trigger the driver to operate on the WiFi chip. It usually runs in the background, as shown below:

```
[root@myir:]/# wpa_supplicant -B -Dnl80211 -c /etc/wpa_supplicant.conf -i wlan0
```

- -B : Run daemons in the background
- -D : Drive name
- -c : Path to configuration information
- -i : Listening wifi interface

● Get ip address

After the configuration is complete, run the WiFi acquisition script "/etc/test/wifi-on.sh" to connect to the wifi, as follows:

```
[root@myir:]/# udhcpc -b -i wlan0 -R
udhcpc: started, v1.29.3
```



```
udhcpc: sending discover
udhcpc: sending select for 192.168.0.166
udhcpc: lease of 192.168.0.166 obtained, lease time 86400
deleting routers
adding dns 192.168.0.1
adding dns 192.168.0.1
```

● **Ping is normal or not**

```
[root@myir:~]# ping www.baidu.com -I wlan0
PING www.baidu.com (36.152.44.95): 56 data bytes
64 bytes from 36.152.44.95: seq=0 ttl=55 time=17.005 ms
64 bytes from 36.152.44.95: seq=1 ttl=55 time=14.794 ms
64 bytes from 36.152.44.95: seq=2 ttl=55 time=21.479 ms
```

4.3. 4G /5G

Linux devices can also be connected to 4G or 5G modules for dial-up Internet access. The MYD-YT113X development board uses the EM05-CE 4G module and the RM500Q-CN 5G module.

The dialing method has pppd, gobinet and qmi_wwan 3 ways, among which pppd is more general, gobinet is not used, qmi_wwan connection is faster and can be used to do 5G module connection, the following EM05-CE module for example, before the test to connect the module to the J16 interface of the development board, in addition to the 4G module need to connect a good IPEX4 generation antenna.

YT113X has two SIM card slots, J16 and J17. 4G module can only use SIM 1 (J16). 5G module can access two SIM cards at the same time, but cannot use SIM 1 and SIM 2 for dialing at the same time, only through the command to switch SIM cards, choose to use SIM 1 or SIM 2, later will introduce how to switch SIM cards to use.

4.3.1. View VID and PID

```
[root@myir:/]# lsusb
Bus 001 Device 001: ID 1d6b:0002
```



```
Bus 001 Device 003: ID 2c7c:0125
Bus 001 Device 002: ID 1a40:0101
Bus 002 Device 001: ID 1d6b:0001
Bus 001 Device 004: ID 0bda:b733
```

2c7c:0125 : VID and PID information of EM05-CE.

4.3.2. View kernel recognition module

If the kernel adds the VID and PID configuration of this module, then the node `"/dev/ttyUSB*"` will be generated:

```
[root@myir:~]# ls -l /dev/ttyUSB*
crw-rw-rw- 1 root root 188, 0 Jan 1 00:00 /dev/ttyUSB0
crw-rw-rw- 1 root root 188, 1 Jan 1 00:00 /dev/ttyUSB1
crw-rw-rw- 1 root root 188, 2 Jan 1 00:00 /dev/ttyUSB2
crw-rw-rw- 1 root root 188, 3 Jan 1 00:00 /dev/ttyUSB3
```

4.3.3. Initial testing with AT commands

The AT command can be used to easily check the signal strength, whether the SIM card is inserted, whether the SIM card is currently searching for an operator, and also to test the current card function by using AT to make a call. Here you also need to know which device is the communication port for AT communication, here you need to check the module file, EM05 and RM500Q use `ttyUSB2` for AT communication. Here we use `microcom` for example, but we can also use `minicom`. e.g.: `microcom "/dev/ttyUSB2"` Enter mode `"ctrl+x"` to exit.

● Query signal quality

```
[root@myir:~]# microcom /dev/ttyUSB2
at+csq
+CSQ: 27,99

OK
```

➤ 27,99: 27 is the signal quality, ranging from 0 to 31 (99 means no signal) the larger the number represents the stronger the signal.

● Check if the card is recognized

```
at+cpin?
```



```
+CPIN: READY
```

```
OK
```

➤ +CPIN:READY : READY Representative Ready.

● **View operators**

```
at+cops?
```

```
+COPS: 0,0,"CHN-UNICOM",7
```

```
OK
```

➤ CHN-UNICOM,7: CHN-UNICOM represents Unicom, 7 represents the use of 2G, 3G, 4G, or 5G according to the module manual to view.

If all the above 3 steps work, you can dial-up the Internet.

● **View Card Slot**

Sometimes the sim card is inserted in slot one (j16), but recognized in slot two (j17), then the dialing will fail. The following command is to check the current card slot position and switch the card slot

Query Card Slot

```
[root@myir:~]# microcom /dev/ttyUSB2
```

```
AT+QUIMSLLOT?
```

```
+QUIMSLLOT: 1
```

```
OK
```

Switching card slot for 1

```
[root@myir:~]# microcom /dev/ttyUSB2
```

```
AT+QUIMSLLOT=1
```

```
OK
```

```
+CPIN: NOT READY
```

```
+QSIMSTAT: 1,0
```

```
+QSIMSTAT: 1,1
```





+CPIN: READY

+QUSIM: 1

+QIND: SMS DONE

+QIND: PB DONE

4.3.4. ppp dialing test

Here we use the pppd dialing command that comes with the development board:

```
[root@myir:~]# pppd call quectel-ppp &
[1] 3349
[root@myir:~]# pppd options in effect:
debug          # (from /etc/ppp/peers/quectel-ppp)
nodetach        # (from /etc/ppp/peers/quectel-ppp)
dump           # (from /etc/ppp/peers/quectel-ppp)
noauth         # (from /etc/ppp/peers/quectel-ppp)
user test      # (from /etc/ppp/peers/quectel-ppp)
password ?????? # (from /etc/ppp/peers/quectel-ppp)
remotename 3gppp # (from /etc/ppp/peers/quectel-ppp)
/dev/ttyUSB3   # (from /etc/ppp/peers/quectel-ppp)
115200        # (from /etc/ppp/peers/quectel-ppp)
lock          # (from /etc/ppp/peers/quectel-ppp)
connect chat -s -v -f /etc/ppp/peers/quectel-chat-connect # (fro
m /etc/ppp/peers/quectel-ppp)
disconnect chat -s -v -f /etc/ppp/peers/quectel-chat-disconnect # (fro
m /etc/ppp/peers/quectel-ppp)
nocrtscts      # (from /etc/ppp/peers/quectel-ppp)
modem          # (from /etc/ppp/peers/quectel-ppp)
hide-password  # (from /etc/ppp/peers/quectel-ppp)
novj           # (from /etc/ppp/peers/quectel-ppp)
novjccomp      # (from /etc/ppp/peers/quectel-ppp)
```



```

ipcp-accept-local          # (from /etc/ppp/peers/quectel-ppp)
ipcp-accept-remote        # (from /etc/ppp/peers/quectel-ppp)
ipparam 3gppp             # (from /etc/ppp/peers/quectel-ppp)
noipdefault               # (from /etc/ppp/peers/quectel-ppp)
ipcp-max-failure 30       # (from /etc/ppp/peers/quectel-ppp)
defaultroute              # (from /etc/ppp/peers/quectel-ppp)
usepeerdns                # (from /etc/ppp/peers/quectel-ppp)
noccip                   # (from /etc/ppp/peers/quectel-ppp)
abort on (BUSY)
abort on (NO CARRIER)
abort on (NO DIALTONE)
abort on (ERROR)
abort on (NO ANSWER)
timeout set to 30 seconds
send (AT^M)
expect (OK)
AT^M^M
OK
-- got it

send (ATE0^M)
expect (OK)
^M
ATE0^M^M
OK
-- got it

send (ATI;+CSUB;+CSQ;+CPIN?;+COPS?;+CGREG?;&D2^M)
expect (OK)
^M
^M
Quectel^M
EM05^M

```



Revision: EM05CEFCR06A04M1G_ND^M

^M

SubEdition: V01^M

^M

+CSQ: 27,99^M

^M

+CPIN: READY^M

^M

+COPS: 0,0,"CHN-UNICOM",7^M

^M

+CGREG: 0,1^M

^M

OK

-- got it

send (AT+CGDCONT=1,"IP","3gnet",,0,0^M)

expect (OK)

^M

^M

OK

-- got it

send (ATD*99#^M)

expect (CONNECT)

^M

^M

CONNECT

-- got it

Script chat -s -v -f /etc/ppp/peers/quectel-chat-connect finished (pid 3355), status = 0x0

Serial connection established.

using channel 1



Using interface ppp0

Connect: ppp0 <--> /dev/ttyUSB3

sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x638d4bf3> <pcomp>
<accomp>]

rcvd [LCP ConfReq id=0x0 <asyncmap 0x0> <auth chap MD5> <magic 0x25c
e332e> <pcomp> <accomp>]

sent [LCP ConfAck id=0x0 <asyncmap 0x0> <auth chap MD5> <magic 0x25c
e332e> <pcomp> <accomp>]

rcvd [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x638d4bf3> <pcomp>
<accomp>]

rcvd [LCP DiscReq id=0x1 magic=0x25ce332e]

rcvd [CHAP Challenge id=0x1 <d805e32118c8f48f5a286b7a3d8f3395>, name =
"UMTS_CHAP_SRVR"]

sent [CHAP Response id=0x1 <168e53c0728c7137458e5c745361ab8f>, name =
"test"]

rcvd [CHAP Success id=0x1 ""]

CHAP authentication succeeded

CHAP authentication succeeded

sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.
0>]

rcvd [IPCP ConfReq id=0x0]

sent [IPCP ConfNak id=0x0 <addr 0.0.0.0>]

rcvd [IPCP ConfNak id=0x1 <addr 10.229.255.26> <ms-dns1 218.104.111.122>
<ms-dns2 218.106.127.114>]

sent [IPCP ConfReq id=0x2 <addr 10.229.255.26> <ms-dns1 218.104.111.122>
<ms-dns2 218.106.127.114>]

rcvd [IPCP ConfReq id=0x1]

sent [IPCP ConfAck id=0x1]

rcvd [IPCP ConfAck id=0x2 <addr 10.229.255.26> <ms-dns1 218.104.111.122>
<ms-dns2 218.106.127.114>]

Could not determine remote IP address: defaulting to 10.64.64.64

local IP address 10.229.255.26

remote IP address 10.64.64.64




```
primary   DNS address 218.104.111.122
secondary DNS address 218.106.127.114
Script /etc/ppp/ip-up started (pid 3369)
Script /etc/ppp/ip-up finished (pid 3369), status = 0x0
```

You can see that it has been connected normally and can get the IP.

```
[root@myir:/]# ifconfig
eth0      Link encap:Ethernet  HWaddr 76:93:32:78:12:87
          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:37

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: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)

ppp0      Link encap:Point-to-Point Protocol
          inet addr:10.229.255.26  P-t-P:10.64.64.64  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:
1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:52 (52.0 B)  TX bytes:58 (58.0 B)
```

➤ ppp0 : ppp0 is the dial-up network card device, ip address is normally obtained.



4.3.5. Ping Test

```
[root@myir:/]# ping www.baidu.com -l ppp0
PING www.baidu.com (112.80.248.76): 56 data bytes
64 bytes from 112.80.248.76: seq=0 ttl=55 time=56.226 ms
64 bytes from 112.80.248.76: seq=1 ttl=55 time=54.201 ms
64 bytes from 112.80.248.76: seq=2 ttl=55 time=61.772 ms
64 bytes from 112.80.248.76: seq=3 ttl=55 time=30.343 ms
64 bytes from 112.80.248.76: seq=4 ttl=55 time=41.040 ms
--- www.baidu.com ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 30.343/48.716/61.772 ms
```





5. Web Applications

The factory burned image of the development board contains some common network applications by default, which is convenient for users to develop or debug.

5.1. PING

PING is mainly used to test network connectivity, network latency and packet loss rate. Once the Ethernet connection is configured, you can use PING to perform a simple test of the network connection.

5.1.1. Wiring and information output

Connect the development board to the switch or router via CAT6 cable, the console will show the connection information output by the kernel as follows: (tested with CAT6 cable speed stable, using other cable speed may not be stable).

```
[root@myir:~]# dmesg
[ 844.954902] 000: sunxi-gmac 4500000.eth eth0: Link is Up - 1Gbps/Full - flow control off
```

5.1.2. Ping Test

```
[root@myir:/]# ping www.baidu.com
PING www.baidu.com (36.152.44.95): 56 data bytes
64 bytes from 36.152.44.95: seq=0 ttl=55 time=12.555 ms
64 bytes from 36.152.44.95: seq=1 ttl=55 time=12.235 ms
64 bytes from 36.152.44.95: seq=2 ttl=55 time=12.169 ms
^C
--- www.baidu.com ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 12.169/12.319/12.555 ms
```

The above result shows that the IP address of www.baidu.com after domain name resolution is 36.152.44.95, icmp_seq represents the number of icmp packets, if the number is consecutive, it means there is no packet loss; time represents the



response delay time, of course, the shorter the time is, the better. In addition to testing Ethernet, the ping command can also be used to test Wi-Fi.



5.2. SSH

SSH stands for Secure Shell, developed by the IETF's Network Working Group; SSH is a security protocol built on the application layer, and is a more reliable protocol designed to provide security for remote login sessions and other network services. Typically dropbear or OpenSSH is used on Linux platforms to implement SSH server and client.

The following tests the use of the SSH client and server on the Ethernet connection, the client and server programs provided by openssh 7.6p1 (<http://www.openssh.com/>) are included in the current factory default. First configure the connection from the development board Ethernet interface to the SSH server, the configured Ethernet card address is as follows:

```
[root@myir:/]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr CE:6F:68:C9:B6:CA
          inet addr:192.168.0.134  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::bd96:938:88b8:ebb6/64  Scope:Link
          inet6 addr: 2409:8a4d:c7d:8002:9df:a35c:219b:17e3/64  Scope:Global
          inet6 addr: 2409:8a4d:c7d:8002::3/128  Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:27  errors:0  dropped:0  overruns:0  frame:0
          TX packets:42  errors:0  dropped:0  overruns:0  carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2820 (2.7 KiB)  TX bytes:3994 (3.9 KiB)
          Interrupt:37
```

The IP address of the current SSH server is 192.168.0.134, use the ping command to test whether the connection between the development board and the SSH server is normal.

● SSH client testing

The development board is used as a client to connect to the SSH server, and the ssh command is used on the development board to log in to the server:

```
[root@myir:/]# ssh zhaoy@192.168.1.13
The authenticity of host '192.168.0.179 (192.168.0.179)' can't be established.
```



ECDSA key fingerprint is SHA256:i8HTmmrawvyVAgfWzHuJKCTiZiO/KgDNbpFbXRatY/U.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added '192.168.0.179' (ECDSA) to the list of known hosts.

sur@192.168.0.179's password:

Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-84-generic x86_64)

- * Documentation: <https://help.ubuntu.com>
- * Management: <https://landscape.canonical.com>
- * Support: <https://ubuntu.com/advantage>

37 updates can be applied immediately.

To see these additional updates run: apt list --upgradable

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.

New release '20.04.6 LTS' available.

Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2023.

*** System restart required ***

Last login: Wed May 3 17:23:28 2023 from 192.168.0.131

sur@myir:~\$

where "sur" is the username on the server

After a successful login, the console on the SSH server is automatically accessed and the user can perform control of the remote server from the client with "sur" user rights. If you need to exit, you can do so by executing the "exit" command directly from the current console.

● SSH server side testing

The development board acts as the SSH server, and other external devices connect to this board remotely. The SSH service is started by default on the



development board, so we can also use the ssh command on other external devices (development board or PC) with SSH client to log in to the current development board, the command and the result are as follows: (you need to set a password to connect for the first time, the current development board is passwordless)

```
sur@myir:~$ ssh root@192.168.0.102
The authenticity of host '192.168.0.102 (192.168.0.102)' can't be established.
ECDSA key fingerprint is SHA256:0RvKGGFq8awWygR0xLg9zviXlru1Ctk3FVN7+qkpGIQ.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.0.102' (ECDSA) to the list of known hosts.
root@192.168.0.102's password:
COLUMNS=145;LINES=49;export COLUMNS LINES;
[root@myir:~]#
```

In the example above, we have logged in to the board as root from a remote location and entered the console, which allows us to control the board with root user rights. If you need to exit, just 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 offers a large set of secure tunneling features, 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 of the host computer to suit their needs. For example, if you want the SSH server to allow the root account to log in remotely without a password, you can modify the "/etc/ssh/" file on the SSH server. "/etc/ssh/sshd_config" and add the following two lines. "PermitRootLogin yes" "PermitEmptyPasswords yes" The above configuration has a high security risk and is usually used for remote deployment during the debugging phase. In real products, it is usually turned off for security reasons.

5.3. SCP



SCP stands for Secure Copy, which is a secure remote file copy command based on SSH protocol for linux systems, and is very useful in the debugging phase of the system. We have already introduced the example of remote login using SSH protocol and SSH client and server, and here is the example of remote file copy using SCP command:

5.3.1. Copy files from remote to local

```
sur@myir:~$ scp test root@192.168.0.102:/root
root@192.168.0.102's password:
test
100% 13 3.0KB/s 00:00
```

This file can be found by entering the "/root" directory of the development board, as follows:

```
[root@myir:/root]# ls
test
```

5.3.2. Copy files from local to remote

```
[root@myir:/root]# scp test sur@192.168.0.179:~/
sur@192.168.1.13's password:
```

After successful verification, the files are copied from the development board to the \$HOME directory of the specified account on the server.

```
sur@myir:~$ ls
test
```

By adding the "-r" parameter, you can also make a copy of the directory, please refer to the help of the scp command for details.

5.4. TFTP

TFTP uses client and server software to connect and transfer files between devices. TFTP uses the UDP protocol and does not have a login function. For example, the TFTP protocol is supported in the common u-boot, which can be used to load the Linux system on the server via the network and implement the network boot



function. The default image file contains the tftp client program provided by busybox, with the following command syntax:

```
[root@myir:/]# tftp --help
Usage: tftp [-4][-6][-v][-l][-m mode] [host [port]] [-c command]
```

Detailed parameters are described as follows:

- -g : Get the file
- -p : Upload a file
- -l : local files
- -r : Remote files
- HOST: remote host IP address

You can choose tftp-hpa for Linux or tftpd 32/64

(http://tftpd32.jounin.net/tftpd32_download.html) for windows. The following is an example of the tftp server configuration for ubuntu platform.

5.4.1. Installing the TFTP server

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

● Configuring the TFTP Service

Create the TFTP server working directory and open the TFTP service configuration file, as follows:

```
$ mkdir -p /home/sur/tftpboot
$ chmod -R 777 /home/sur/tftpboot
$ sudo vi /etc/default/tftpd-hpa
```

Modify or add the following fields:

```
TFTP_DIRECTORY="/home/sur/tftpboot"
TFTP_OPTIONS="-l -c -s"
```

● Restart the TFTP service

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

After configuring the tftp server, place a test file zImage into the "<WORKDIR>/tftpboot/" directory configured above, and you can use the tftp client on the board to download and upload files..



```
[root@myir:~]# tftp -g -r zImage -l zImage 192.168.0.179
```

The above command will download the "zImage" from the "/tftpboot" directory of the tftp server to the current directory of the development board.

```
[root@myir:~]# tftp -p -l config -r config_01 192.168.0.179
```

The above command will upload the "config" file from the current directory on the development board to the "<WORKDIR>/tftpboot" directory previously configured by the tftp server, and rename it to "config_01".

5.5. DHCP

DHCP (Dynamic Host Configuration Protocol) is a network protocol for local area networks. DHCP also contains two roles: server-side and client-side. DHCP server-side mode was tested in 4.2 when configuring WiFi's AP mode to assign IP addresses to connected WiFi devices. Here is another description of how to manually obtain an IP address using the udhcpc command for users to use when debugging the network.

- **Use the udhcpc command to configure the IP address**

```
[root@myir:~]# udhcpc -i eth0
udhcpc: started, v1.29.3
udhcpc: sending discover
udhcpc: sending select for 192.168.0.102
udhcpc: lease of 192.168.0.102 obtained, lease time 86400
deleting routers
route: SIOCADDRT: Network is unreachable
adding dns 192.168.0.1
adding dns 192.168.0.1
```

Either way, you end up with an IP address for eth0, as well as the gateway, subnet mask, DNS, and other information as follows:

```
[root@myir:~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr C2:35:61:AF:A2:D8
          inet addr:192.168.0.102  Bcast:192.168.0.255  Mask:255.255.255.0
```



```
inet6 addr: fe80::c035:61ff:feaf:a2d8/64 Scope:Link
inet6 addr: 2409:8a4d:c7d:8002:a39c:43fa:f405:7e2c/64 Scope:Global
inet6 addr: 2409:8a4d:c7d:8002::4/128 Scope:Global
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:63051 errors:0 dropped:0 overruns:0 frame:0
TX packets:63064 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:35047097 (33.4 MiB) TX bytes:2916065 (2.7 MiB)
Interrupt:37
```

5.6. Iptables

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

Development boards using Linux systems typically use the iptables utility to configure the firewall. iptables handles various packets, such as accept, reject, and drop, according to the methods defined by the packet filtering rules.

The following test uses iptables to block icmp packets and disable other external devices on the network from pinging them. See

<https://linux.die.net/man/8/iptables> for specific command usage.

5.6.1. Configure the development board iptables

Use iptables to configure the development board to drop incoming icmp packets and not respond to ping probes from other hosts.

```
[root@myir:/]# iptables -A INPUT -p icmp --icmp-type 8 -j DROP
[root@myir:/]# iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
-A INPUT -p icmp -m icmp --icmp-type 8 -j DROP
```



5.6.2. ping Test

ping the development board on the development host and specify the deadline as 10, the result is as follows:

```
C:\Users\40938>ping 192.168.0.102 -w 10

Pinging 192.168.0.102 with 32 bytes of data:
Request timed out.
Request timeout.
Request timeout.
Request timeout.

Ping statistics for 192.168.0.102:
    Packets: Sent = 10, Received = 0, Missing = 10 (100% missing)
```

The above results show that the development host cannot ping through the development board after setting the firewall.

- **Delete the corresponding firewall rule**

```
[root@myir:/]# iptables -F
[root@myir:/]# iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
```

- **Test again ping development board**

```
C:\Users\40938>ping 192.168.0.102 -w 10

Pinging 192.168.0.102 with 32 bytes of data:
Reply from 192.168.0.102: Bytes=32 Time=1ms TTL=64
Reply from 192.168.0.102: Bytes=32 Time=1ms TTL=64
Reply from 192.168.0.102: bytes=32 time=1ms TTL=64
Reply from 192.168.0.102: bytes=32 time=1ms TTL=64

Ping statistics for 192.168.0.102:
    Packets: Sent = 10, Received = 10, Missing = 0 (0% missing).
```



Estimated time in milliseconds for round-trip travel.

Min = 1ms, Max = 1ms, Avg = 1ms

After clearing the iptables rules, you can ping the development board again from the development host, and then you can ping through. The above example is just a simple demonstration, in fact iptables with various rules can achieve very powerful functions, here will not be described in detail.

5.7. iperf3

iperf3 is a tool for actively measuring the maximum achievable bandwidth on an IP network. It supports various parameters such as test time, buffer size, and protocol (TCP, UDP, SCTP under IPV4 and IPV6). iperf3 can be divided into server mode or client mode by role, and we can use it to test and view network bandwidth, TCP window value, retransmission probability, etc. in TCP mode, and also to test packet loss rate, delay and jitter in specified UDP bandwidth. Delay and Jitter.

We open Windows PowerShell on the development host, the host computer with a Gigabit network card as the server side of iperf3, and the development board under test as the client side to test the TCP and UDP performance of the development board network card respectively. First, iperf3 is installed on the host computer as follows::

Connect the server and the development board directly via CAT6 cable and configure their respective IP addresses. For example, we set the server ip to 192.168.0.179, set the development board IP to 192.168.0.102, and use the ping command to test to ensure that they are connected to each other.

Note: Try not to connect to a router or switch to avoid the test results being affected by the transmission and forwarding of intermediate devices.

5.7.1. Testing TCP Performance

● Servers (192.168.0.179)

On the server, iperf3 uses the -s parameter to indicate that it is working in server-side mode.



```
PS D:\iperf-3.1.3-win64> .\iperf3.exe -s
```

```
-----  
Server listening on 5201  
-----
```

● Clients (192.168.0.102)

The iperf3 program running on the development board works in client, TCP mode, where the parameters are described as follows:

- -c 92.168.0.102 : working on the client side, connecting to the server 192.168.0.179
- -i 2 : Test results are reported at 2-second intervals
- -t 10 : Total test duration is 10 seconds

```
[root@myir:/]# iperf3 -c 192.168.0.179 -i 2 -t 10
```

```
Connecting to host 192.168.0.179, port 5201
```

```
[ 5] local 192.168.0.102 port 49692 connected to 192.168.0.179 port 5201
```

[ID]	Interval	Transfer	Bitrate	Retr	Cwnd
[5]	0.00-2.00 sec	218 MBytes	914 Mb/s	242	625 KBytes
[5]	2.00-4.00 sec	219 MBytes	918 Mb/s	14	551 KBytes
[5]	4.00-6.00 sec	221 MBytes	927 Mb/s	0	609 KBytes
[5]	6.00-8.00 sec	220 MBytes	923 Mb/s	0	631 KBytes
[5]	8.00-10.00 sec	220 MBytes	923 Mb/s	369	554 KBytes

[ID]	Interval	Transfer	Bitrate	Retr	
[5]	0.00-10.00 sec	1.07 GBytes	921 Mb/s	625	sender
[5]	0.00-10.00 sec	1.07 GBytes	918 Mb/s		receiver

```
iperf Done.
```

The client finishes the test after 10 seconds and displays the above test results, indicating that the TCP bandwidth is around 921 Mb/s without retransmission and the TCP window value is 625KBytes at the time of the test. The server also displays the test results as follows, and then continues to listen to the port waiting for the client to connect:

```
PS D:\iperf-3.1.3-win64> .\iperf3.exe -s
```



```
-----
Server listening on 5201
-----
```

```
Accepted connection from 192.168.0.102, port 35716
```

```
[ 5] local 192.168.0.102 port 5201 connected to 192.168.0.179 port 35718
```

```
[ ID] Interval          Transfer      Bandwidth
```

```
[ 5] 0.00-2.00 sec 220 MBytes 921 Mbits/sec
```

```
[ 5] 2.00-4.00 sec 219 MBytes 917 Mbits/sec
```

```
[ 5] 4.00-6.00 sec 221 MBytes 925 Mbits/sec
```

```
[ 5] 6.00-8.00 sec 219 MBytes 920 Mbits/sec
```

```
[ 5] 8.00-10.00 sec 219 MBytes 917 Mbits/sec
```

```
[ 5] 10.00-10.02 sec 1.38 MBytes 530 Mbits/sec
```

```
- - - - -
```

```
[ ID] Interval          Transfer      Bandwidth
```

```
[ 5] 0.00-10.02 sec 1.07 GBytes 921 Mbits/sec 78 sender
```

```
[ 5] 0.00-10.02 sec 1.07 GBytes 919 Mbits/sec receiver
```

```
-----
```

```
Server listening on 5201
-----
```

5.7.2. Testing UDP performance

● Servers (192.168.0.179)

Continue running iperf3 on the server using the -s parameter to indicate that it is working in server-side mode.

```
PS D:\iperf-3.1.3-win64> .\iperf3.exe -s
```

```
-----
```

```
Server listening on 5201
-----
```

● Clients (192.168.0.102)

iperf3 on the device works in client, UDP mode, where the parameters are described as follows:

➤ -u : working in UDP mode



- -c 192.168.0.102 : working on the client side, connecting to the server side 192.168.0.179
- -i 2 : test result reporting interval is 2 seconds
- -t 10 : total test duration is 10 seconds
- -b 100M : Set the UDP transmission bandwidth to 100Mbps.

```
[root@myir:/]# iperf3 -c 192.168.0.179 -u -i 2 -t 10 -b 100M
Connecting to host 192.168.0.102, port 5201
[ 5] local 192.168.0.102 port 36915 connected to 192.168.0.179 port 5201
[ ID] Interval            Transfer        Bitrate          Total Datagrams
[ 5]  0.00-2.00      sec   23.8 MBytes    100 Mb/s/sec    17259
[ 5]  2.00-4.00      sec   23.8 MBytes    100 Mb/s/sec    17265
[ 5]  4.00-6.00      sec   23.8 MBytes    100 Mb/s/sec    17265
[ 5]  6.00-8.00      sec   23.8 MBytes    100 Mb/s/sec    17265
[ 5]  8.00-10.00     sec   23.8 MBytes    100 Mb/s/sec    17265
-----
[ ID] Interval            Transfer        Bitrate          Jitter    Lost/Total Datagrams
ms
[ 5]  0.00-10.00     sec   119 MBytes    100 Mb/s/sec    0.000 ms    0/86319
(0%)  sender
[ 5]  0.00-10.00     sec   119 MBytes    99.4 Mb/s/sec    0.186 ms   466/86313
(0.54%) receiver

iperf Done.
```

The client finishes the test after 10 seconds and displays the above test results, showing that UDP has no packet loss at the specified bandwidth of 100Mbps.

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

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




```
Accepted connection from 192.168.0.102, port 49694
[ 5] local 192.168.0.102 port 5201 connected to 192.168.0.179 port 40126
[ ID] Interval            Transfer      Bandwidth      Jitter    Lost/Total Datagrams
[ 5]  0.00-2.00    sec   23.8 MBytes  99.8 Mb/s     0.230 ms    0/17240
(0%)
[ 5]  2.00-4.00    sec   23.8 MBytes  99.6 Mb/s     0.161 ms    67/17267 (0.39%)
[ 5]  4.00-6.01    sec   23.6 MBytes  98.7 Mb/s     0.926 ms    124/17208 (0.72%)
[ 5]  6.01-8.00    sec   23.7 MBytes  99.8 Mb/s     0.171 ms    136/17330 (0.78%)
[ 5]  8.00-10.00   sec   23.6 MBytes  99.2 Mb/s     0.186 ms    139/17258 (0.81%)
[ 5] 10.00-10.00   sec   14.1 KBytes  61.1 Mb/s     0.186 ms     0/10 (0%)
-----
[ ID] Interval            Transfer      Bandwidth      Jitter    Lost/Total Datagrams
[ 5]  0.00-10.00   sec   119 MBytes  100 Mb/s     0.186 ms    466/86313 (0.54%)
-----
Server listening on 5201
-----
```

Client modify -b parameter, continue to increase the specified UDP bandwidth, the maximum rate that the sender can achieve is the maximum bandwidth, packet loss rate depends on the server CPU performance, NIC buffer size, the following method to send a fixed size of packets to test the packet loss rate:

```
[root@myir:/]# iperf3 -u -c 192.168.0.179 -i 2 -t 10 -b 1000M
Connecting to host 192.168.0.102, port 5201
[ 5] local 192.168.0.102 port 45388 connected to 192.168.0.179 port 5201
[ ID] Interval            Transfer      Bitrate      Total Datagrams
[ 5]  0.00-2.00    sec   59.0 MBytes  248 Mb/s     42746
[ 5]  2.00-4.00    sec   59.1 MBytes  248 Mb/s     42766
```



```
[ 5]  4.00-6.00  sec  59.5 MBytes  250 Mb/s  43116
[ 5]  6.00-8.00  sec  60.0 MBytes  252 Mb/s  43455
[ 5]  8.00-10.00 sec  60.0 MBytes  252 Mb/s  43454
- - - - -
[ ID] Interval      Transfer      Bitrate      Jitter      Lost/Total Datagrams
ms
[ 5]  0.00-10.00  sec  298 MBytes  250 Mb/s  0.000 ms  0/215537
(0%) sender
[ 5]  0.00-10.00  sec  298 MBytes  250 Mb/s  0.044 ms  0/215536
(0%) receiver

iperf Done.
```

There are many parameters that can be configured in the process of testing iperf3, and users can make targeted adjustments to the test according to the actual application needs. For example, you can increase the value of the -t parameter for long time stress testing, or specify the -P parameter for concurrent stress testing of multiple connections, etc. For more information about iperf3 testing, please refer to: <https://iperf.fr/iperf-doc.php#3doc>.



6. Graphics System

6.1. QT

QT is a cross-platform C++ GUI application development framework. It can be used to develop both GUI programs and non-GUI programs, such as console tools and servers. qt is object-oriented framework, using special code generation extensions and some macros, qt is easy to extend and allows true component programming.

The development board will be shipped with the Qt runtime library burned in, and a rich HMI demo system is provided, which can be viewed in the “MEasy HMI2.x Development Manual” .

6.1.1. Get information about qt

First check the current system support QT version, as follows:

```
[root@myir:~]# ls /usr/local/  
Qt_5.12.5
```

6.1.2. Introduction to the QT runtime environment

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

- **qtenv.sh**

On embedded Linux systems, several platform plugins can be used: EGLFS, LinuxFB, DirectFB or Wayland. however, the availability of these plugins depends on the characteristics of the actual hardware platform and the way Qt is configured, on the MYD-YT113X platform the linuxfb plugin is used.

In the MYD-YT113X platform, we load the environment variables needed to run the QT program via the script qtenv.sh, which reads:

```
[root@myir:~]# cat /etc/qtenv.sh  
  
export QTDIR=/usr/local/Qt_5.12.5  
if [ -d $QTDIR ];then
```





```

export QT_ROOT=$QTDIR
export PATH=$QTDIR/bin:$PATH
export LD_LIBRARY_PATH=$QTDIR/lib:/usr/lib/cedarx/:$LD_LIBRARY_PA
TH

export QT_QPA_PLATFORM_PLUGIN_PATH=$QT_ROOT/plugins
export QT_QPA_PLATFORM=linuxfb:tty=/dev/fb0
export QT_QPA_FONTDIR=$QT_ROOT/fonts

export QML_IMPORT_PATH=$QTDIR/qml
export QML2_IMPORT_PATH=$QTDIR/qml

TouchDevice="generic ft5x06 (79)"
for InputDevices in /sys/class/input/input*
do
    DeviceName=`cat $InputDevices/name`
    if [[ $DeviceName == $TouchDevice ]];then
        TouchDeviceNum=${InputDevices##*input}
        export QT_QPA_EVDEV_TOUCHSCREEN_PARAMETERS=/dev/i
nput/event$TouchDeviceNum
        echo "add "/dev/input/event$TouchDeviceNum "to Qt Appli
cation."
        break
    fi
done
if [ ! -n "$TouchDeviceNum" ]; then
    echo "Error:Input device $TouchDevice can not be found,plz check it!"
fi

export QT_QPA_PLATFORM=linuxfb

```



```

export set TSLIB_TSDEVICE=/dev/input/dev/input/event$TouchDeviceNum
export set TSLIB_CONFFILE=/etc/ts.conf
export set TSLIB_PLUGINDIR=/usr/lib/ts
export set TSLIB_CALIBFILE=/etc/pointercal
export set TSLIB_CONSOLEDEVICE=none
export set TSLIB_FBDEVICE=/dev/fb0
export QT_QPA_GENERIC_PLUGINS=evdevtouch,evdevmouse:/dev/input/
event6(Touch Devices)
#export QT_QPA_EGLFS_INTEGRATION=eglfs_mali
#export QT_QPA_FB_HIDE_CURSOR=1
#export QT_QPA_EGLFS_HIDE_CURSOR=1
#export QT_QPA_EGLFS_ROTATION=90

#export QWS_MOUSE_PROTO=Intellimouse:/dev/input/event6

#export DBUS_SESSION_BUS_ADDRESS=`cat /tmp/dbusaddr`
mkdir -p /dev/shm
#ulimit -c unlimited
mxapp2 &
echo "find qt5 installed done"
fi

```

● Display parameter configuration

QT applications can use the QScreen class or QDesktopWidget to get the parameters related to the screen display to write applications that match the screen. The screen resolution and color depth obtained through QScreen or QDesktopWidget are generally fine, but sometimes the physical size obtained may not be correct due to the display driver. In this case, you can configure and adjust the following parameters to make the actual interface display elements fit the size of the display screen.

In general, the default configuration can be used, but if there is indeed a mismatch between the displayed elements and the actual screen, the relevant parameters can be adjusted appropriately according to the above description.



● Input Peripheral Configuration

When there is no windowing system on the embedded Linux device (e.g. XWindow or Weston), the mouse, keys or haptic devices get input device information by reading evdev directly or by using other intermediate libraries such as libinput or tslib. eglfs and linuxfb platform plugins include these two input methods. The Qt5 input device configuration is as follows:

The linuxfb platform plug-in uses EvdevTouch input handler by default, this way is often used to handle capacitive touch, if the event coordinates reported by the capacitive touch driver correspond exactly to the actual screen area coordinates, there is no need to do additional processing, if the reverse occurs, some adjustments can be made through environment variables, EvdevTouch input handler supports some additional parameters as follows.

From the above code, we can see that the linuxfb platform plug-in uses EvdevTouch input handler by default, this way is often used to handle capacitive touch, the event coordinates reported by the capacitive touch driver correspond exactly to the actual screen area coordinates, there is no need to do additional processing, if there is a reverse, some adjustments can be made through the environment variables, EvdevTouch input handler supports some additional parameters as follows:

Table 6-1. QT Linuxfb Touch Handler-related Environment Variable Parameters

Parameters	Description
/dev/input/...	Specify the name of the input device. If not specified, Qt will find the right device by libudev or by traversing the available nodes.
rotate	On some touchscreens, the coordinates must be rotated by setting them to 90, 180 or 270.
invertx/inverty	Specify the parameters used to invert the X or Y coordinates in the input event.

For example, if QT_QPA_EVDEV_TOUCHSCREEN_PARAMETERS passes the following value to the platform plug-in before starting the application, it explicitly specifies the touch device as "/dev/input/event6" with its coordinates flipped 180 degrees. This is useful when the orientation of the actual screen and the touch screen do not match.



```
export QT_QPA_EVDEV_TOUCHSCREEN_PARAMETERS=/dev/input/event6:rotate=180
```

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. For details on the use of tslib, refer to <https://github.com/libts/tslib/blob/master/README.md>.

Note: The tslib input handler is commonly used for resistive touch, generates mouse events and only supports single-touch, and requires screen calibration for initial use.

6.1.3. Start Qt program

When we need to run our own QT program on MYD-YT113X platform, we need to modify the qtenv script file to add the path of the executable QT program to the script file, so that the correct environment variables can be configured after running the script. To run the QT program qt_test, you need to make the following changes at the end of the "qtenv.sh" script.

```
export QWS_MOUSE_PROTO=
    export DBUS_SESSION_BUS_ADDRESS=`cat /tmp/dbusaddr`
    mkdir -p /dev/shm
    ulimit -c unlimited
    #debug Launcher &
    #mxapp2 &
    /etc/qt_test &
    echo "find qt5 installed done"
fi
```

Here we comment out the QT application mxapp2, which was executed by default before, and add our own QT application qt_test path to the script.

Since mxapp2 was already started by default, if you need to run your own Qt application at this point, you need to terminate mxapp2 first before starting another application.

You can exit the process directly by kill.

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



7. Multimedia Applications

7.1. Video Playback

7.1.1. xplayerdemo

Allwinner system comes with player xplayerdemo which can decode and play video.

```
[root@myir:/mnt/u/video]# xplayerdemo
WARNING: awplayer <log_set_level:30>: Set log level to 3
ERROR : awplayer <ReadPluginEntry:194>: read plugin entry adecoder-15 fail!
ERROR : awplayer <ReadPluginEntry:194>: read plugin entry vdecoder-8 fail!
INFO : cedarc <CedarPluginVDInit:79>: register h264 decoder success!
INFO : cedarc <CedarPluginVDInit:84>: register mjpeg decoder success!
INFO : cedarc <CedarPluginVDInit:86>: register mpeg2 decoder success!
WARNING: awplayer <DIOpenPlugin:112>: Invalid plugin,function CedarPluginVDInit not found.
INFO : cedarc <CedarPluginVDInit:92>: register mpeg4dx decoder success!
INFO : cedarc <CedarPluginVDInit:79>: register mpeg4H263 decoder success!
INFO : cedarc <CedarPluginVDInit:90>: register mpeg4Normal decoder success!
INFO : cedarc <CedarPluginVDInit:74>: register vc1 decoder success!
ERROR : awplayer <ReadPluginEntry:194>: read plugin entry plugin-0 fail!
DEBUG : awplayer <AwStreamInit:99>: aw stream init...
```

- * This program implements a simple player,
- * you can type commands to control the player.
- * To show what commands supported, type 'help'.
- * Implemented by Allwinner ALD-AL3 department.

```
DEBUG : awplayer <XPlayerCreate:216>: XPlayerCreate.
DEBUG : awplayer <LogVersionInfo:34>:
```





View player support commands:

```
demoPlayer# help
```

```
*****
```

```
* This is a simple media player, when it is started, you can input commands to tell
```

```
* what you want it to do.
```

```
* Usage:
```

```
* # ./demoPlayer
```

```
* # set url: http://www.allwinner.com/ald/al3/testvideo1.mp4
```

```
* # show media info
```

```
* # play
```

```
* # pause
```

```
* # stop
```

```
* Command and it param is seperated by a colon, param is optional, as below:
```

```
* Command[: Param]
```

```
* here are the commands supported:
```

```
* help:
```

```
* show this help message.
```

```
* quit:
```

```
* quit this program.
```

```
* set url:
```

```
* set url of the media, for example, set url: ~/testfile.mkv.
```

```
* play:
```

```
* start playback.
```

```
* pause:
```

```
* pause the playback.
```

```
* stop:
```

```
* stop the playback.
```

```
* set speed:
```

```
* stop the playback.
```

```
* seek to:
```



```
*          seek to specific position to play, position is in unit of second,
ex, seek to: 100.
*  show media info:
*          show media information of the media file.
*  show duration:
*          show duration of the media file.
*  show position:
*          show current play position,in unit of second.
*  switch audio:
*          switch audio to a track, for example, switch audio: 2, track is
start counting from 0.
*
*****
```

7.1.2. Play Video

Let's take playing "h264_output.mp4" from a mounted USB drive as an example:

```
demoPlayer# set url:/mnt/usb/sda1/h264_output.mp4
DEBUG : awplayer <XPlayerSetDataSourceUrl:456>: setDataSource(url), url='/m
nt/u/video/4KCanada_h264.mp4'
INFO : awplayer <XPlayerThread:1707>: process message XPLAYER_COMMA
ND_SET_SOURCE.
DEBUG : awplayer <XPlayerPrepare:741>: prepare
DEBUG : awplayer <XPlayerThread:1960>: process message XPLAYER_COMMA
ND_PREPARE. mPriData->mStatus: 1
DEBUG : demuxComponent <DemuxThread:1783>: process message DEMUX_
COMMAND_PREPARE.
DEBUG : demuxComponent <DemuxThread:1850>: === prepare msg
DEBUG : awplayer <CdxParserPrepare:757>: source uri 'file:///mnt/u/video/4K
Canada_h264.mp4'
.....
INFO : awplayer <XPlayerThread:1996>: xxxxxxxxxx video size: width = 1920,
height = 960
++++ video width: 1920, height: 960
```



```
DEBUG : awplayer <CallbackForAwPlayer:440>: info : preared
info: prepare ok.
preparing...
```

This means that playback is ready, execute the command play to start playback.

```
demoPlayer# play
DEBUG : awplayer <XPlayerStart:771>: start
DEBUG : awplayer <XPlayerThread:2140>: process message XPLAYER_COMMA
ND_START.
DEBUG : awplayer <PlayerStart:730>: player start
DEBUG : awplayer <BaseCompPostAndWait:61>: video decoder receive cmd:
start
debug : cedarc <SbmFrameReset:613>: ** wait for reset sem
debug : cedarc <ProcessThread:1653>: *** post reset sem
debug : cedarc <SbmFrameReset:615>: ** wait for reset sem ok
debug : cedarc <SbmFrameReset:620>: SbmFrameReset finish
DEBUG : awplayer <BaseCompPostAndWait:61>: audio decoder receive cmd:
start
debug : cedarc <H264ProcessExtraData2:543>: H264ProcessNaluUnit, bNeedFi
ndSPS = 0, bNeedFindPPS = 0
(Allwinner Audio Middle Layer),line(958) : Create Decoder!!=====
DEBUG : audioDecIrf <handleStart:1064>: Create libadecoder success...
(Allwinner Audio Middle Layer),line(592) : AudioDec_Installaudiolib ok
(Allwinner Audio Middle Layer),line(595) : audio decoder init start ...
(AllwinnerAlibs),line(626) : libaw_aacdec.so open, use dlopen!
(AllwinnerAlibs),line(660) : Khan----Loading 'libaw_aacdec.so' success!
.....
```

7.2. Audio

This section is a test play audio.

7.2.1. Debugging tools



● Tinyplay

Put the audio file into the u disk, and then plug the speaker into the j3 interface.

Usage: tinyplay file.wav [-D card] [-d device] [-p period_size] [-n n_periods] [-T capture time]

```
[root@myir:/mnt/usb/sda1]# tinyplay 01+Singalongsong.wav
playing '01+Singalongsong.wav': 2 ch, 48000 hz, 16 bit
```

● alsamixer

Volume adjustment tool, execute alsamixer command, can adjust the left and right channel and volume and other parameters.

```
[root@myir:/]# alsamixer -a
```

alsamixer: option requires an argument -- 'a'

Usage: alsamixer [options]

Useful options:

-h, --help	this help
-c, --card=NUMBER	sound card number or id
-D, --device=NAME	mixer device name
-V, --view=MODE	starting view mode: playback/capture/all

Debugging options:

-g, --no-color	toggle using of colors
-a, --abstraction=NAME	mixer abstraction level: none/basic

The following figure so, use the direction "←" "→" key to control the cursor to select the modified items, use "↑" "↓" to adjust the size of the parameters.

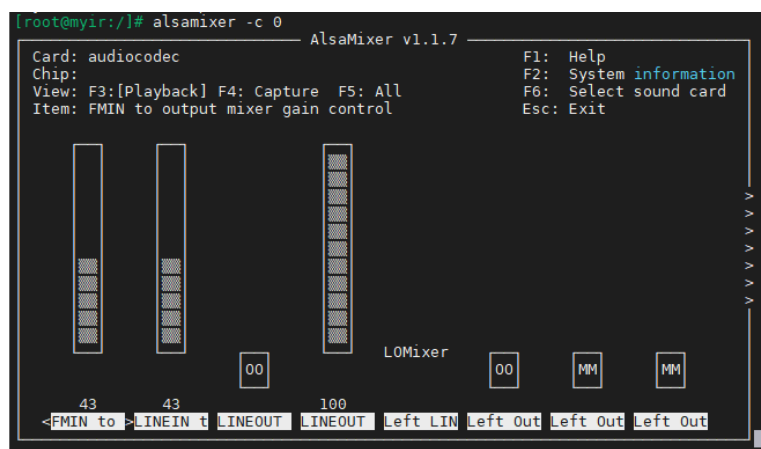


Figure 6-1. Player parameter adjustment UI





8. System Tools

The default image contains a number of common system tools that can be used to view and manage system resources during system debugging or in the actual deployed product, and can also be invoked in SHELL scripts or other applications. These tools may not fully meet the user's system customization needs, and the system developer will need to make appropriate adjustments as appropriate.

8.1. Compression and decompression tools

This section tests the system's decompression tool. Compression is the ability to compress multiple files into one zip package can compress multiple files into one zip package to facilitate file transfer. And decompression can be compressed compressed files back to the original size for ease of use. This section will be in the file system to tar, gzip, gunzip and other tools as an example to explain.

8.1.1. tar

Now we use the tar tool in Linux, which not only packs files, but also compresses, views, adds and unpacks them. Here is the package operation. Enter the following command to see the tar syntax format:

```
[root@myir:/]# tar --help
```

```
BusyBox v1.29.3 (2022-01-28 15:52:25 CST) multi-call binary.
```

```
Usage: tar c|x|t [-hvokO] [-f TARFILE] [-C DIR] [-T FILE] [-X FILE] [--exclude PA  
TTERN]... [FILE]...
```

Create, extract, or list files from a tar file

c	Create
x	Extract
t	List
-f FILE	Name of TARFILE ('-' for stdin/out)
-C DIR	Change to DIR before operation
-v	Verbose
-O	Extract to stdout



```
-o      Don't restore user:group
-k      Don't replace existing files
-h      Follow symlinks
-T FILE File with names to include
-X FILE File with glob patterns to exclude
--exclude PATTERN      Glob pattern to exclude
```

● Compressing with tar

Create a new "test.txt" file, and enter the following command to package the file into .gz format:

```
[root@myir:/]# tar -cf test.tar.gz test.txt
[root@myir:/]# ls
test.tar.gz
```

● Decompress with tar

Unpack the file in tar.gz format

```
[root@myir:/]# tar -xvf test.tar.gz
test.txt
[root@myir:/]# ls
test.txt
```

8.1.2. gzip

● Syntax format

gzip is a frequently used command for compressing and decompressing files on Linux systems, which is both convenient and easy to use. To view the gzip syntax, enter the following command in the development board terminal:

```
[root@myir:/]# gzip --hple
gzip: unrecognized option '--hple'
BusyBox v1.29.3 (2022-01-28 15:52:25 CST) multi-call binary.
Usage: gzip [-cfkdt] [FILE]...
Compress FILEs (or stdin)

        -d      Decompress
```




```
-t      Test file integrity
-c      Write to stdout
-f      Force
-k      Keep input files
```

● Compressing files with gzip

```
[root@myir:/]# gzip test.txt
[root@myir:/]# ls
test.tar.gz
```

● Decompress the file with gunzip

```
[root@myir:/]# gunzip test.txt.gz
[root@myir:/]# ls
test.txt
```

8.2. File System Tools

The main test system file system tools, this section will introduce several common file system management tools. The system's own file system tool "mount".

8.2.1. mount

mount is a command under Linux that hooks up a partition to a folder in Linux, thus linking the partition to that directory, so that if we access the folder, it is equivalent to accessing the partition, with the following application 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.

● Mounted USB flash drive



```
[root@myir:/]# mount /dev/sda1 /mnt/
```

8.3. Disk Management Tools

The main test is the system's disk management tools, and this section will introduce several common disk management tools. The system comes with disk management tools "fdisk", "dd", "mkfs", "du", "df", "cfdisk", "fsck". You can monitor the usual disk usage with these commands.

8.3.1. fdisk

The fdisk disk partitioning utility has corresponding applications in 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 be partitioned as an argument directly after the "fdisk" command, with the following application syntax format :

```
[root@myir:/]# fdisk -h
BusyBox v1.29.3 (2022-01-28 15:52:25 CST) multi-call binary.
Usage: fdisk [-ul] [-C CYLINDERS] [-H HEADS] [-S SECTORS] [-b SSZ] DISK
Change partition table

    -u                Start and End are in sectors (instead of cylinders)
    -l                Show partition table for each DISK, then exit
    -b 2048           (for certain MO disks) use 2048-byte sectors
    -C CYLINDERS      Set number of cylinders/heads/sectors
    -H HEADS          Typically 255
    -S SECTORS        Typically 63
```

Partitioning the eMMC:

```
[root@myir:/]# fdisk /dev/mmcblk0
Found valid GPT with protective MBR; using GPT

Command (m for help): m
Command Action
o      create a new empty DOS partition table
p      print the partition table
```



```
q      quit without saving changes
s      create a new empty Sun disklabel
```

Command (m for help):

8.3.2. dd

The "dd" command is used to copy the specified input file to the specified output file. And format conversion can be performed during the copying process. The difference between the "dd" command and the "cp" command is that the dd command can be used on a floppy disk without a file system created, and the data copied to the floppy disk is actually an image file. It is similar to the function of diskcopy command in DOS. The format of the "dd" command is: dd [<if=input filename/device name>] [<of=output filename/device name>] [bs=block byte size] [count=block count].

Create a file with a size of 2M.

```
[root@myir:/]# time dd if=/dev/zero of=ffmpeg1 bs=2M count=1 conv=fsync
1+0 records in
1+0 records out

real    0m0.094s
user    0m0.000s
sys     0m0.027s
```

8.3.3. du

The "du" command is used to display the disk space usage. This command displays the file system block usage for each level of subdirectory of the specified directory, step by step. The general syntax of "du" is as follows:

```
[root@myir:/]# du --help
BusyBox v1.29.3 (2022-01-28 15:52:25 CST) multi-call binary.
Usage: du [-aHLdclsxhmk] [FILE]...
Summarize disk space used for each FILE and/or directory

-a      Show file sizes too
```



- L Follow all symlinks
- H Follow symlinks on command line
- d N Limit output to directories (and files with -a) of depth < N
- c Show grand total
- l Count sizes many times if hard linked
- s Display only a total for each argument
- x Skip directories on different filesystems
- h Sizes in human readable format (e.g., 1K 243M 2G)
- m Sizes in megabytes
- k Sizes in kilobytes (default)

Partial description of parameters:

- -a: show the size of all directories or files
- -h: output in K,M,G to improve readability of information
- -k: output in KB
- -m: output in MB
- Count the size of the files generated by the "dd" command:

```
[root@myir:/]# du ffmpeg1
2048    ffmpeg1
[root@myir:/]# du -h ffmpeg1
2.0M    ffmpeg1
```

8.3.4. df

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

```
[root@myir:/]# df -help
df: invalid option -- 'e'
BusyBox v1.29.3 (2022-01-28 15:52:25 CST) multi-call binary.
Usage: df [-PkmhT] [FILESYSTEM]...
Print filesystem usage statistics

-P      POSIX output format
-k      1024-byte blocks (default)
```



-m	1M-byte blocks
-h	Human readable (e.g. 1K 243M 2G)
-T	Print filesystem type

Description of some of the parameters:

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

To view the number of inodes under a partition and the usage of inodes, use the following command:

```
[root@myir:]/# df -h
Filesystem                Size      Used    Available    Use%     Mo
nted on
/dev/mmcblk0p4            1.9G      805.6M    1.1G          41%      /
tmpfs                     490.1M    64.0K      490.0M         0%
  /tmp
tmpfs                     490.1M    32.0K      490.1M         0%
  /run
devtmpfs                  480.7M     0          480.7M         0%
  /dev
/dev/mmcblk0p8            2.9G       9.0M      2.7G          0%      /
media
tmpfs                     490.1M     0          490.1M         0%
  /dev/shm
```

The inode is partitioned by the system when we format it, and is related to the size of the disk partition. When our inode usage has reached 100%, we can't write data to the disk even if we still have space left.

8.4. Process Management Tools



A process is also an important concept in an operating system. It is a process that is executed once by a program. A program is a static description of a process, and every program running in the system runs in its process. All processes in a Linux system are interconnected, and all processes except the initialization process have a parent process. New processes are not created, but copied, or copied from previous processes. All processes in Linux are derived from an init process with process number 1. The Linux system includes 3 different types of processes, each with its own characteristics and properties:

- Interactive process: A process started by a Shell that can run both in the foreground and in the background.
- Batch process: This kind of process is not connected to the terminal and is a sequence of processes. Such processes are submitted to a waiting queue for sequential execution of processes.
- Monitored processes (daemons): daemons are always active and usually run in the background. Daemons are usually started by the system at the beginning by automatic activation of a script or by root.

For Linux system, process management is an important part, for process management is usually achieved through process management tools, Linux system more commonly used process management commands are the following: ps, top, vmstat, kill.

8.4.1. ps

- **Syntax format**

Displays the current running status of the system processes, with the following general syntax:

```
[root@myir:/]# ps --help
BusyBox v1.29.3 (2022-01-28 15:52:25 CST) multi-call binary.

Usage: ps [-o COL1,COL2=HEADER]

Show list of processes
```



-o COL1,COL2=HEADER Select columns for display

Description of some parameter combinations:

- -u: user-centric organization of process status information display;
- -a: processes not related to the terminal;
- -x: processes related to the terminal; (threads, that is, lightweight processes;)

Usually the above commands are used in combination: aux.

- --e: display all processes; equivalent to ax;
- -f: displaying full-format program information;

Usually the above commands are used in combination: ef

- -H: display the number of processes in process hierarchy
- -F: display more program information

Commonly used in combination: eHF.

● Display information about all processes

```
[root@myir:/]# ps
PID    USER    COMMAND
  1  root      init
  2  root      [kthreadd]
  4  root      [kworker/0:0H]
  6  root      [ksoftirqd/0]
  7  root      [rcu_preempt]
  8  root      [rcu_sched]
  9  root      [rcu_bh]
 10  root      [migration/0]
 11  root      [lru-add-drain]
 12  root      [cpuhp/0]
 13  root      [cpuhp/1]
 14  root      [migration/1]
 15  root      [ksoftirqd/1]
 17  root      [kworker/1:0H]
 18  root      [cpuhp/2]
```



```
19 root    [migration/2]
20 root    [ksoftirqd/2]
22 root    [kworker/2:0H]
23 root    [cpuhp/3]
24 root    [migration/3]
.....
```

8.4.2. top

● Syntax format

The "top" command puts a considerable amount of overall system performance information on one screen. The display can also be changed interactively. The syntax of "top", which dynamically and continuously monitors the running status of processes, is generally as follows:

```
[root@myir:/]# top --help
```

```
BusyBox v1.29.3 (2022-01-28 15:52:25 CST) multi-call binary.
```

```
Usage: top [-b] [-nCOUNT] [-dSECONDS]
```

Provide a view of process activity in real time.

Read the status of all processes from /proc each SECONDS and display a screenful of them.

Keys:

N/M/P/T: sort by pid/mem/cpu/time

R: reverse sort

Q,^C: exit

Options:

-b Batch mode

-n N Exit after N iterations

-d N Delay between updates

● Dynamic view of system processes




```
[root@myir:/]# top
Mem: 234716K used, 769024K free, 88K shrd, 10208K buff, 62924K cached
CPU:  0% usr  2% sys  0% nic 97% idle  0% io  0% irq  0% sirq
Load average: 0.00 0.00 0.00 1/132 32743

  PID  PPID  USER      STAT  VSZ  %VSZ  %CPU  COMMAND
32736  1903  root       R      2580   0%    2%  top
 1705    1  root       S     294m  30%    0%  /etc/video2lcd
 1532    1  root       S     156m  16%    0%  adbd
 1903    1  root       S     3624   0%    0%  -/bin/sh
 1345    1  root       S     2716   0%    0%  dbus-daemon --system
    1    0  root       S     2580   0%    0%  init
 1316    1  root       S     2580   0%    0%  /sbin/syslogd -n
 1321    1  root       S     2580   0%    0%  /sbin/klogd -n
 1417    1  root       S     2328   0%    0%  /usr/sbin/dropbear -R
 1398    1  root       S     2192   0%    0%  /sbin/dhcpd -f /etc/dhcpd.conf
 1455    1  root       S     2160   0%    0%  /usr/sbin/tftpd -c -l -s /var/lib/tf
tpboot
  589    2  root       SW        0   0%    0%  [vsync proc 0]
    7    2  root       SW        0   0%    0%  [rcu_preempt]
 1226    2  root       SW        0   0%    0%  [cec thread]
 6256    2  root       SW        0   0%    0%  [kworker/u8:0]
28128    2  root       SW        0   0%    0%  [kworker/u8:1]
  921    2  root       SW        0   0%    0%  [kworker/0:1]
 1225    2  root       SW        0   0%    0%  [hdmi proc]
 1232    2  root       SW        0   0%    0%  [tve detect]
  551    2  root       SW        0   0%    0%  [kworker/2:1]
 1504    2  root       SW        0   0%    0%  [mali-simple-pow]
 8359    2  root       SW        0   0%    0%  [kworker/2:0]
  548    2  root       SW        0   0%    0%  [kworker/1:1]
  554    2  root       SW        0   0%    0%  [kworker/3:1]
```

8.4.3. kill

- Syntax format



Sends the specified signal to the corresponding process. If no model is specified, SIGTERM (15) will be sent to terminate the specified process. If you cannot terminate the process, you can use the "-KILL" parameter, which sends the signal SIGKILL(9) to force the process to end, and use the ps command or jobs command to check the process number. root user will affect the user's processes, non-root user can only affect their own processes. The general syntax of the "kill" command is as follows:

```
[root@myir:~]# kill --help
```

```
kill: kill [-s sigspec | -n signum | -sigspec] pid | jobspec ... or kill -l [sigspec]
    Send a signal to a job.
```

Send the processes identified by PID or JOBSPEC the signal named by SIGSPEC or SIGNUM. If neither SIGSPEC nor SIGNUM is present, then SIGTERM is assumed.

Options:

```
-s sig    SIG is a signal name
-n sig    SIG is a signal number
-l        list the signal names; if arguments follow '-l' they are
          assumed to be signal numbers for which names should be listed
-L        synonym for -l
```

Kill is a shell builtin for two reasons: it allows job IDs to be used instead of process IDs, and allows processes to be killed if the limit on processes that you can create is reached.

Description of some parameter combinations:

- -s: specifies the signal to be sent
- -p: simulates the sent signal
- -l: specifies the list of names of the signals
- pid: ID number of the process to be aborted



➤ Signal: indicates a signal

First use `ps -ef` with the pipe command to determine the PID of the process to be killed.

```
[root@myir:/]# ps -ef | grep mxapp2
1705 root    /etc/mxapp2
6254 root    grep /etc/mxapp2
```

Then enter the following command to terminate the process:

```
[root@myir:/]# kill 1705
```

The "killall" command terminates all processes in the same process group, allowing the name of the process to be terminated to be specified instead of the PID process number:

```
[root@myir:/]# killall mxapp2
```



9. Development Support

This chapter introduces some basic information about secondary development for the current SDK. The current SDK provides three kinds of reference images for configuration, one is "t113_linux_myir_emmc_core" and "t113_linux_myir_nand", mainly for applications without GUI; the other one is "t113_linux_myir_emmc_full", which adds some requirements to "t113_linux_myir_emmc_core". The other is "t113_linux_myir_emmc_full", which adds some applications that require GUI to "t113_linux_myir_emmc_core". Please refer to "MYD-YT113X_SDK Release Notes" for information about these three images.

9.1. Development Languages

9.1.1. SHELL

Shell is a program written in C language that serves as 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, the common ones are:

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)

MYD-YT113X supports 2 types of bourne shell and Bourne Again Shell:

```
[root@myir:/]# echo "echo 'myir test'" > shell_demo.sh
[root@myir:/]# sh shell_demo.sh
myir test
[root@myir:/]# bash shell_demo.sh
myir test
```

9.1.2. C/C++



C/C++ is the most commonly used programming language for underlying application development on Linux platforms, and is the most efficient language after assembly. Development using C/C++ is usually done in a cross-development manner, i.e., development is done on the development host side, compiled to generate a binary executable that runs on the target machine, and then deployed to run on the target machine.

Using this approach, you first need to install the SDK-based, please refer to the "MYD-YT113X_Linux Software Development Guide" for installation steps, after the installation is complete, you need to configure the SDK environment.

First add the compilation toolchain to the environment variables:

```
sur@myir:~/T113$ export PATH=$PATH: /home/sur/opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi/bin
sur@myir:~/T113$ export CROSS_COMPILE=/home/sur/opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi/bin/arm-linux-gnueabi-gcc
```

```
sur@myir:~/T113$ arm-linux-gnueabi-gcc -v
```

Using built-in specs.

COLLECT_GCC=arm-linux-gnueabi-gcc

COLLECT_LTO_WRAPPER=/home/sur/opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi/bin/./libexec/gcc/arm-linux-gnueabi/5.3.1/lto-wrapper

Target: arm-linux-gnueabi

Configured with: /home/tcwg-buildslave/workspace/tcwg-make-release/label/docker-trusty-amd64-tcwg/target/arm-linux-gnueabi/snapshots/gcc-linaro-5.3-2016.05/configure SHELL=/bin/bash --with-mpc=/home/tcwg-buildslave/workspace/tcwg-make-release/label/docker-trusty-amd64-tcwg/target/arm-linux-gnueabi/_build/builds/destdir/x86_64-unknown-linux-gnu --with-mpfr=/home/tcwg-buildslave/workspace/tcwg-make-release/label/docker-trusty-amd64-tcwg/target/arm-linux-gnueabi/_build/builds/destdir/x86_64-unknown-linux-gnu --with-gmp=/home/tcwg-buildslave/workspace/tcwg-make-release/label/docker-trusty-amd64-tcwg/target/arm-linux-gnueabi/_build/builds/destdir/x86_64-unknown-linux-gnu --with-gnu-as --with-gnu-ld --disable-libstdcxx-pch --disable-libmudflap --with-cloog=no --with-ppl=no --with-isl=no --disable-nls --enable-c99 --with-tune=co



```
rtex-a9 --with-arch=armv7-a --with-fpu=vfpv3-d16 --with-float=softfp --with-m
ode=thumb --disable-multilib --enable-multiarch --with-build-sysroot=/home/t
cwg-buildslave/workspace/tcwg-make-release/label/docker-trusty-amd64-tcwg/t
arget/arm-linux-gnueabi/_build/sysroots/arm-linux-gnueabi --enable-lto --enabl
e-linker-build-id --enable-long-long --enable-shared --with-sysroot=/home/tcw
g-buildslave/workspace/tcwg-make-release/label/docker-trusty-amd64-tcwg/targ
et/arm-linux-gnueabi/_build/builds/destdir/x86_64-unknown-linux-gnu/arm-linux
-gnueabi/libc --enable-languages=c,c++,fortran,lto --enable-checking=release -
-disable-bootstrap --build=x86_64-unknown-linux-gnu --host=x86_64-unknown
-linux-gnu --target=arm-linux-gnueabi --prefix=/home/tcwg-buildslave/workspa
ce/tcwg-make-release/label/docker-trusty-amd64-tcwg/target/arm-linux-gnueabi
/_build/builds/destdir/x86_64-unknown-linux-gnu
Thread model: posix
gcc version 5.3.1 20160412 (Linaro GCC 5.3-2016.05)
```

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

```
#include<stdio.h>
int main(int argc,char *argv[])
{
    printf("hello world!\n");
    return 0;
}
```

C++ written demo program "hello-CXX.cpp"

```
//file: hello-CXX.cpp
#include <iostream>
using namespace std;
int main(int argc,char *argv[])
{
    cout << "hello world!";
    return 0;
}
```



Next, the application is compiled, using the compilation toolchain "arm-linux-gnueabi-gcc" imported above for the hello.c file, and the C++ file "hello-CXX.cpp" using "arm-linux-gnueabi-cpp" toolchain.

```
$arm-linux-gnueabi-gcc hello.c -o hello
or
$ arm-linux-gnueabi-cpp hello-CXX.cpp -o hello-CXX
.....
```

The generated executable is then copied to the target machine via the "scp" command and executed as follows"

```
[root@myir:/]# ./hello
hello world!
or
[root@myir:/]# ./hello-CXX
hello world!
```

For more complex examples and development methods, please refer to the application porting section of the "MYD-YT113X_Linux Software Development Guide" for instructions.

9.1.3. Python

Python is an interpreted, object-oriented, dynamically typed, high-level programming language invented by Guido van Rossum in late 1989, with the first public release in 1991. Like the Perl language, Python source code is also under the GPL (GNU General Public License). This section tests the use of python, in terms of both the python command line and scripts.

- **Check the python version supported by your system**

```
[root@myir:/]# python
python      python3     python3.7   python3.7m
```

- **Python Test**

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



```
[root@myir:/]# python
```

```
Python 3.7.2 (default, Apr 25 2023, 14:12:09)
```

```
[GCC 7.3.1 20180425 [linaro-7.3-2018.05 revision d29120a424ecfbc167ef90065c0eeb on linux
```

```
Type "help", "copyright", "credits" or "license" for more information.
```

```
>>> print("myir test")
```

```
myir test
```

Exit the Python command line, execute exit() to exit Python:

```
>>> exit()
```

```
[root@myir:/]#
```

● Writing scripts to test Python

Write a simple Python script program, all Python files will have the ".py" extension:

```
[root@myir:/]# vi test.py
```

```
[root@myir:/]# cat test.py
```

```
#!/usr/bin/env python3
```

```
print("myir test")
```

Execute the script file, with the interpreter in the "/usr/bin/env" directory running as python2, using the following command:

```
[root@myir:/]# chmod a+x test.py
```

```
[root@myir:/]# ./test.py
```

```
myir test
```

The Python3 interpreter is invoked with script parameters to start executing the script until it finishes. When script execution is complete, the interpreter is no longer valid.

9.2. Database



Database (Database) is a warehouse that organizes, stores and manages data according to data structure. There are many types of databases, commonly used databases are Access, Oracle, Mysql, SQL Server, SQLite, etc.

9.2.1. System SQLite

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 and from common disk files. A complete SQL database containing multiple tables, indexes, triggers and views is contained in a single disk file. This is a lightweight database, an associative database management system that adheres to ACID. It is designed to be embedded and is already in use in many embedded products, it takes up very low resources and may only require a few hundred K of memory in an embedded device. This database runs faster processing speed than both Mysql and PostgreSQL.

- **SQLiteCreate database**

Start sqlite3 and create a new database <testDB.db>, enter the following command in the terminal interface to enter the operation interface.

```
[root@myir:/]# sqlite3 testDB.db
SQLite version 3.25.3 2018-11-05 20:37:38
Enter ".help" for usage hints.
sqlite>
```

The above command will create 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 will provide a sqlite> prompt after the database file is successfully created.

Once the database is created, you can use SQLite's .databases command to check if it's in the database list, as follows:

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

Use the .quit command to exit the sqlite prompt, as follows:

```
sqlite> .quit
```



```
[root@myir:/]
```

If you want to know more about SQLite, please refer to the official website;
<https://www.sqlite.org/docs.html>.

9.3. Qt Application Localization

This section discusses localization-related setup and testing. Localization refers to a program or software that supports internationalization by giving the program region-specific language information to adapt it to the use of region-specific people in terms of processing information input and output, etc. This allows some of the language environment variables used by the program to be dynamically configured during program execution. This chapter focuses on the localization of QT applications, using the MYIR demo image "MEasy HMI 2.x" as an example.

9.3.1. Multilingual

This section mainly uses the MYIR demo image "MEasy HMI 2.x" as an example to illustrate the practical application of multilingualism in QT projects.

Please refer to Chapter 3.1 of the *"MYD-YT113X_MEasy HMI Software Development Guide"* to set up the compilation environment for MEasy HMI before reading the following contents.

- **Open mxapp2 project**

The source code of the "MEasy HMI 2.x" project is mxapp2.tar.gz, copy it to the environment built in the above document, and open the project with QT Creator.

9.3.2. Fonts

This section mainly uses the "MEasy HMI 2.x" demo image as an example to illustrate the practical application of fonts in QT projects. Please refer to Chapter 3.1 of the *"MYD-YT113X_MEasy HMI Software Development Guide"* to set up the compilation environment for MEasy HMI before reading the following contents.

- **Installing Font Files**

Font files can be placed directly in the "/usr/lib/fonts/" directory of the development board file system.

```
[root@myir:/]# ls /usr/lib/fonts/msyh.ttc
```



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

Or added directly to the QT project.

```
sur@myir:~/download/mxapp2/fonts$ tree
├── DIGITAL
│   ├── DIGITAL.TXT
│   └── DS-DIGIB.TTF
└── fontawesome-webfont.ttf
```

● Using Font Files

The use of the font file is required to be called by the application code, the specific call method refers to the 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);
}
```

9.3.3. Soft Keyboard

This section illustrates the practical application of the soft keyboard in QT projects using the Mir demo image "MEasy HMI 2.x" as an example.

Before reading the following content, please refer to chapter 3.1 of *"MYD-YT113X_MEasy HMI Software Development Guide"* to set up the compilation environment of MEasy HMI.



QT has added the qt virtual keyboard component to the official source code since version 5.7. myir has used QT version 5.12 and configured the virtual keyboard component in the "MEasy HMI 2.x" image released to accompany it.

● **Soft keyboard embedded into qml code**

The soft keyboard provided by Qt can only be called inside the qml code, you need to define the location of the soft keyboard and the size of the soft keyboard before calling, 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 soft keyboard**

The soft keyboard is triggered by QML's "TextField" and "TextEdit" components, and the user only needs to add this component to the UI components to call up the soft keyboard and use it. If it is a qt system, you can use QT virtualkeyboard which comes with QT to input.

```
TextField{
id: netmask_input
InputMethodHints: Qt.ImhFormattedNumbersOnly
```



```
onAccepted: digitsField.focus = true  
font.family: "Microsoft YaHei"  
color: "white"  
}
```

- **Using the soft keyboard**

Please refer to *"MYD-YT113X_MEasy HMI Software Development Guide"*, Chapter 2.6 System Setup UI Interface for how to use the soft keyboard. User click on the editable interface component will pop up the soft keyboard.





10. References

- **Linux kernel Open Source Community**
<https://www.kernel.org/>
- **Allwinner Development Community**
<https://www.aw-ol.com>
- **Linux Introduction to Kernel Watchdog**
<https://www.kernel.org/doc/html/latest/watchdog/index.html>
- **Qt on Embedded Linux**
<https://doc.qt.io/qt-5/embedded-linux.html>
- **Systemd**
<https://www.freedesktop.org/software/systemd/man/systemd.network.html>
- **Allwinner Official Website**
<https://www.allwinnertech.com/>





11. Appendix A

11.1. 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 Intelligent 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

