

MYD-YM62X_Linux Software Evaluation Guide



File status: [] Draft [√] released	FILE ID:	MYIR-MYD-YM62X-SW-EG-EN-L6.1.46
	VERSION:	V1.0[DOC]
	AUTHOR:	Beste Wang
	CREATED:	2023-09-15
	UPDATED:	2023-09-25



Revision History

Editions	Author	Participants	Date	Notes
V1.0[doc]	Beste Wang		20230925	Initial version, for MYD-YM62X



CONTENT

Revision History	- 2 -
CONTENT	- 3 -
1. Overview	- 5 -
1.1. Hardware Resources	- 5 -
1.2. Software resources	- 6 -
1.3. Documentation Resources	- 6 -
1.4. Environment Preparation	- 6 -
2. Core Board Resources	- 8 -
2.1. CPU	- 8 -
2.2. Coprocessor M4F	- 13 -
2.3. Graphics	- 13 -
2.4. Memory	- 13 -
2.5. eMMC	- 18 -
2.6. RTC	- 23 -
2.7. Watchdog	- 25 -
3. Peripheral interface	- 28 -
3.1. GPIO	- 28 -
3.2. LED	- 30 -
3.3. Key(press the key)	- 32 -
3.4. RS485	- 33 -
3.5. CAN	- 34 -
3.6. USB	- 37 -
3.7. typeC interface	- 39 -
3.8. Micro SD card	- 40 -
3.9. Display	- 43 -
3.10. Touch Panel	- 46 -
4. Networking and Communication	- 51 -



4.1. Network Devices	- 51 -
4.2. Network applications	- 75 -
5. Graphics system	- 92 -
5.1. GPU	- 93 -
5.2. Wayland+Weston+QT	- 93 -
6. Multimedia applications	- 97 -
6.1. Camera	- 97 -
6.2. Audio	- 98 -
7. System Tools	- 100 -
7.1. Compression and decompression tools	- 100 -
7.2. Filesystem Tools	- 103 -
7.3. Disk Management Tools	- 106 -
7.4. Process management tools	- 108 -
8. Development support	- 113 -
8.1. Development Language	- 113 -
9. Resources	- 118 -
Appendix A	- 119 -
Warranty & Technical Support Services	- 119 -



1. Overview

Linux Software Evaluation Guide is used to introduce the testing steps and evaluation methods of core resources and peripheral resources under open source Linux system running on MYIR's development board. This article can be used as a pre-evaluation guide, and can also be used as a test guide for general system development.

1.1. Hardware Resources

This document is used in MYD-YM62X series board of MYIR Electronics, which is developed on the embedded development platform of MYIR Electronics based on TI's high-performance embedded ARM processor. The core chips used in this series include three models: YM6254, YM6252 and YM6231. Different models are provided in this series to meet different performance requirements. Each model is equipped with a Cortex-M4F coprocessor for powerful performance and flexibility.

The following table Outlines the three models of the MYD-YM62X series board and their key features:

Table 1-1. Introduction of board models

CPU	Processor core	M4F	Core board	Base Board
AM6231	Single core ARM Cortex-A53	is	MYC-YM6231	MYB-Y62X
AM6252	Dual-core ARM Cortex-A53	is	MYC-YM6252	MYB-Y62X
AM6254	Quad-core ARM Cortex-A53	is	MYC-YM6254	MYB-Y62X

See MYD-YM62X Product Manual for detailed configuration parameters on the hardware part. The user will use some accessories in the evaluation test process, see the list below.

Table 1-2. Optional modules

Accessories	Interface method	Instructions and links
LCD	LVDS port	MY-LVDS070C https://www.myirtech.com/list.asp?id=634
Camera module	CSI port	MY-CAM003M

1.2. Software resources

The BSP of MYD-YM62X series development board is transplanted and modified based on TI's official open source community edition Linux BSP, and the system image is built by Yocto project. The software resources of Bootloader, Kernel and file system are all open in the form of source code. For details, please refer to the MYD-YM62X SDK Release Notes.

The image of the development board has been burned at the factory, you only need to power on to use.

1.3. Documentation Resources

According to the different stages of the user's use of the development board, the SDK contains different types of documents and manuals in addition to the release instructions, such as getting started guide, evaluation guide, development guide, application notes, common questions and answers. For a detailed list of documents, see the instructions in Table 2-4 of the *MYD-YM62X SDK1.0.0 Release Notes*.

1.4. Environment Preparation

Before you start to evaluate the development board software, you need to do some necessary preparation for the development board and configure some environment, including the correct hardware wiring, configuring the debugging serial port, setting up the boot and other steps. The detailed steps can refer to "*MYD-YM62X Quick Start Guide*".

The following sections focus on how to evaluate and test the hardware resources and interfaces of the system as well as the software functions. Mainly with the help of some commonly used tools and commands under Linux, as well as self-developed applications for testing. Software evaluation guide is divided



ded into several parts to describe, including: core resources, peripheral resources, network applications, multimedia applications, development support applications, system tools and other major categories. The following chapters will give a comprehensive explanation of each part, and describe the specific evaluation methods and steps of each part of the resources in detail.

2. Core Board Resources

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

2.1. CPU

The core chips of the MYD-YM62X series development board are AM6231, AM6252, AM6254, etc. The AM62X series chip adopts multi-core heterogeneous chip architecture, with built-in high-performance Cortex-A53 CPU core and independent security control from the core Cortex M4F core. It supports secure boot and comes with a secure OS, a dedicated security controller with a user-programmable HSM core, and dedicated secure DMA and IPC subsystems for isolated processing. Each A53 core contains a 32KB L1 DCache with SECD ECC functionality and a 32KB L1 ICache with parity protection; Integrated with Ethernet/UART/SPI/IIC/MCASP/JTAG/MIPI and CSI/GPMC/an LVDS/rich peripheral interfaces.

The following will test the status or function of the CPU through some commands, and MYD-YM6254 will be taken as an example.

1) To view the CPU information command:

Read the system CPU provider and parameter information, you can get through the file /proc/cpuinfo.

```
root@myd-am62x:~# cat /proc/cpuinfo
processor       : 0
BogoMIPS      : 400.00
Features       : fp asimd evtstrm aes pmull sha1 sha2 crc32 cpuid
CPU implementer : 0x41
CPU architecture: 8
CPU variant    : 0x0
```



```

CPU part      : 0xd03
CPU revision   : 4

processor     : 1
BogoMIPS      : 400.00
Features      : fp asimd evtstrm aes pmull sha1 sha2 crc32 cpuid
CPU implementer : 0x41
CPU architecture: 8
CPU variant    : 0x0
CPU part      : 0xd03
CPU revision   : 4

processor     : 2
BogoMIPS      : 400.00
Features      : fp asimd evtstrm aes pmull sha1 sha2 crc32 cpuid
CPU implementer : 0x41
CPU architecture: 8
CPU variant    : 0x0
CPU part      : 0xd03
CPU revision   : 4

processor     : 3
BogoMIPS      : 400.00
Features      : fp asimd evtstrm aes pmull sha1 sha2 crc32 cpuid
CPU implementer : 0x41
CPU architecture: 8
CPU variant    : 0x0
CPU part      : 0xd03
CPU revision   : 4
    
```

- Processor: The number of the logical processing cores in the system. For multicore processors, this may be either a physical core or a virtual logical core using hyperthreading

- **BogoMIPS:** A rough estimate of the number of million Instructions Per Second running on a CPU at kernel startup

2) Checking CPU usage

```

root@myd-am62x:~# top
top - 03:41:26 up 3 min, 2 users, load average: 0.07, 0.21, 0.10
Tasks: 184 total, 1 running, 183 sleeping, 0 stopped, 0 zombie
%Cpu(s): 2.5us, 3.8 sy, 0.0 ni, 93.7 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem: 1924.1 total, 1614.7 free, 137.0 used, 172.4 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 1705.5 avail Mem

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
7855 2231 root R 2792 0% 1 6% top
1997 1 root S 11924 1% 2 0% /usr/bin/adbd
1966 1 root S 6876 0% 0 0% sshd: /usr/sbin/sshd [listener]
0
1938 1 root S 5044 0% 3 0% /usr/libexec/bluetooth/bluetoothd
1958 1957 www-data S 4308 0% 3 0% nginx: worker process
1957 1 root S 3876 0% 1 0% nginx: master process /usr/sbin/nginx
1933 1 dbus S 2980 0% 3 0% dbus-daemon --system
2231 1 root S 2928 0% 0 0% -sh
2170 2153 root S 2792 0% 2 0% {led} /bin/sh /usr/bin/led 1
1 0 root S 2792 0% 2 0% init
6828 6826 root S 2792 0% 1 0% {cantest} /bin/sh /usr/bin/cantest
6831 6829 root S 2792 0% 3 0% {cantest} /bin/sh /usr/bin/cantest
2153 1 root S 2792 0% 1 0% {main} /bin/sh /usr/bin/main
1904 1 root S 2792 0% 3 0% /sbin/syslogd -n
1908 1 root S 2792 0% 3 0% /sbin/klogd -n

```

```
6826      1 root      S      2792   0%    1    0% {cantest} /bin/sh /usr/bin/can
test
6829      1 root      S      2792   0%    3    0% {cantest} /bin/sh /usr/bin/can
test
6824  2153 root      S      2660   0%    0    0% sleep 600
7853  6828 root      S      2660   0%    0    0% sleep 1
7854  6831 root      S      2660   0%    3    0% sleep 1
```

- %usr: Indicates cpu usage of userspace programs (not scheduled by nice)
- %sys: Indicates system-space cpu usage, mostly for kernel programs
- %nic: Indicates cpu usage for userspace programs scheduled via nice
- %idle: Idle cpu
- %irq: The number of hard interrupts handled by the cpu
- %sirq: The number of softirqs handled by the cpu

3) Get CPU temperature information

CPU built-in temperature sensor as CPU temperature acquisition, can be very convenient to obtain the CPU internal temperature.

```
root@myd-am62x:~# cat /sys/class/thermal/thermal_zone0/temp
56300
```

It shows the number as thousandths of a degree, and dividing by 1,000 is the current temperature value.

4) CPU Stress Test

There are many ways to test the pressure of the CPU. The bc command can be used to calculate PI to test the stability of the CPU in the operation process.

```
root@myd-am62x:~# echo "scale=5000; 4*a(1)" | bc -l -q &
root@myd-am62x:~#
```

The above command will calculate PI in the background and be accurate to 5000 decimal places. The calculation process will take a while. At this point, we can check the change in CPU utilization with the top command, like this:

```
root@myd-am62x:~# top
top - 03:45:43 up 7 min, 2 users, load average: 0.00, 0.09, 0.07
```

```
Tasks: 155 total,  1 running, 154 sleeping,  0 stopped,  0 zombie
%Cpu(s): 2.5us, 3.8 sy, 0.0 ni, 92.4 id, 0.0 wa, 0.0 hi, 1.3 si, 0.0 st
MiB Mem: 1924.1 total, 1613.8 free, 137.7 used, 172.6 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 1704.8 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
20125	2231	root		R	3260	0%	1	24%	bc -l -q		
1997	1	root		S	11924	1%	2	0%	/usr/bin/adbd		
1966	1	root		S	6876	0%	0	0%	sshd: /usr/sbin/sshd [listener]		

After about 3 minutes, the PI result is calculated. The CPU usage of this device is very high, and there is no exception, indicating that the CPU stress test can pass. The test pressure can be further increased by increasing the accuracy requirement.

```
Root @ myd - am62x: \ ~ # 3.1415926535897932384626433832795028841971
69399375105820974944592307
8164062862089986280348253421170679821480865132823066470938446095505
8 \
2231725359408128481117450284102701938521105559644622948954930381964
4 \
2881097566593344612847564823378678316527120190914564856692346034861
0 \
4543266482133936072602491412737245870066063155881748815209209628292
5 \
```

2.2. Coprocessor M4F

The AM62X series is a multi-core heterogeneous processor consisting of ARM Cortex-A53 and Arm Cortex-M4F. It can run both soft real-time Linux system and hard real-time RTOS system.

For specific use of M4 core, please read the application note "MYD-YM62X M4 Coprocessor Application Development Notes".

2.3. Graphics

MYD-YM62X series, YM6254 and YM6252 both have GPU modules inside the chip, support Vulkan 1.2, OpenGL 3.x/2.0/1.1, support 2D and 3D graphics. Y M6231 does not have GPU,

For the specific operation process, refer to the subsequent section 5.1 graphics and image processing.

2.4. Memory

YM62X device memory has 1G/2G DDR4 to choose from, according to different board type has different collocation available for use.

1) View memory information

The /proc/meminfo file can be used to read the system's memory parameters.

```
root@myd-am62x:~# cat /proc/meminfo
```

```
MemTotal:      950828 kB
MemFree:       591912 kB
MemAvailable:  680040 kB
Buffers:       9500 kB
Cached:        147800 kB
SwapCached:      0 kB
Active:        28512 kB
```



```

Inactive:          195140 kB
Active(anon):      1260 kB
Inactive(anon):    81408 kB
Active(file):      27252 kB
Inactive(file):    113732 kB
Unevictable:       0 kB
Mlocked:           0 kB
SwapTotal:         0 kB
SwapFree:          0 kB
Dirty:             112 kB
Writeback:         0 kB
AnonPages:         66472 kB
Mapped:            79856 kB
Shmem:             16316 kB
KReclaimable:      18192 kB
Slab:              43020 kB
SReclaimable:      18192 kB
SUnreclaim:        24828 kB
KernelStack:       2768 kB
PageTables:        2280 kB
SecPageTables:     0 kB
NFS_Unstable:      0 kB
Bounce:            0 kB
WritebackTmp:      0 kB
CommitLimit:       475412 kB
Committed_AS:      304988 kB
VmallocTotal:      133143592960 kB
VmallocUsed:        10756 kB
VmallocChunk:       0 kB
Percpu:            536 kB
HardwareCorrupted: 0 kB
AnonHugePages:     8192 kB
ShmemHugePages:    0 kB

```

```
ShmemPmdMapped:      0 kB
FileHugePages:       0 kB
FilePmdMapped:       0 kB
CmaTotal:            262144 kB
CmaFree:             245508 kB
HugePages_Total:     0
HugePages_Free:      0
HugePages_Rsvd:      0
HugePages_Surp:      0
Hugepagesize:        2048 kB
Hugetlb:             0 kB
```

- **MemTotal:** The total amount of system memory in kilobytes (KB).
- **MemFree:** The amount of available memory, which represents the memory that is not currently being used.
- **MemAvailable:** An estimate of the available memory, taking into account the memory management policies of the kernel.
- **Buffers:** The amount of memory used for filesystem buffering, that is, the buffers used when files are read and written. In this example, buffers take up 9,500 KB.
- **Cached:** The amount of memory used for the file system cache, including cached file and directory data.
- **SwapCached:** The amount of memory that has been cached into swap space; typically 0 indicates that no swap space is being used.
- **Active:** The amount of memory that is currently being used. This includes Active anonymous memory (anon) and Active file memory (file).
- **Inactive:** Inactive memory, the amount of memory that was once used but is no longer used. This includes Inactive anonymous memory (Inactive(anon)) and inactive file memory (inactive (file)).

2) Getting memory usage

Memory usage can be read using the `free` command with the `-m` argument in MByte. The following is shown in 1G DDR.

```
root@myd-am62x:~# free -m
```

	total	used	free	shared	buff/cache	available
Mem:	928	190	638	16	99	654
Swap:	0	0	0			

- total: Total amount of memory
- used: The amount of memory that is being used
- free: The amount of memory that can be used

3) Memory stress test

By specifying the size and number of memory tests, the existing memory of the system can be tested under pressure. The system tool memtester can be used for testing, such as specifying the memory size 200MB, the test number is 10, and the test command is "memtester 200M 10".

The following is an example of using 300MB memory space for a single test:

```
root@myd-am62x:~# memtester 300M 1
memtester version 4.5.1 (64-bit)
Copyright (C) 2001-2020 Charles Cazabon.
Licensed under the GNU General Public License version 2 (only).

pagesize is 4096
pagesizemask is 0xfffffffffff000
want 300MB (314572800 bytes)
got 300MB (314572800 bytes), trying mlock ... locked.
Loop 1/1:
  Stuck Address      : ok
  Random Value       : ok
  Compare XOR        : ok
  Compare SUB        : ok
  Compare MUL        : ok
  Compare DIV        : ok
  Compare OR         : ok
  Compare AND        : ok
  Sequential Increment: ok
  Solid Bits         : ok
```



Block Sequential : ok
Checkerboard : ok
Bit Spread : ok
Bit Flip : ok
Walking Ones : ok
Walking Zeroes : ok

Done.

2.5. eMMC

eMMC is a data storage device that includes a MultiMediaCard (MMC) interface, a NAND Flash component. Its cost, small size, Flash technology independence, and high data throughput make it ideal for embedded products. MYD-YM62X series comes standard with 8GB eMMC, this section will explain the steps and methods to view and operate eMMC under Linux system.

1) Check eMMC capacity

You can query the eMMC partition information and capacity by using the `fdisk -l` command.

```
root@myd-am62x:~# fdisk -l
Disk /dev/mmcbk0: 7.28 GiB, 7818182656 bytes, 15269888 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xd31fb445

Device            Boot  Start      End  Sectors  Size Id Type
/dev/mmcbk0p1          204800  450559    245760   120M  c W95 FAT32 (LBA)
/dev/mmcbk0p2 450560 15269887 14819328   7.1G  83 Linux
```

```
Disk /dev/mmcbk0boot0: 4 MiB, 4194304 bytes, 8192 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

```
Disk /dev/mmcbk0boot1: 4 MiB, 4194304 bytes, 8192 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
```

I/O size (minimum/optimal): 512 bytes / 512 bytes

2) View eMMC partition information

The df command can query the eMMC partition information, usage, mount directory and other information.

```
root@myd-am62x:~# df -h
Filesystem                Size      Used Available Use% Mounted on
/dev/root                 6.6G    1.3G    5.0G    21% /
devtmpfs                  335.2M    4.0K    335.2M    0% /dev
tmpfs                     464.3M    0      464.3M    0% /dev/shm
tmpfs                     185.7M   11.0M   174.7M    6% /run
tmpfs                     4.0M     0      4.0M     0% /sys/fs/cgroup
tmpfs                     464.3M    4.0K    464.3M    0% /tmp
tmpfs                     16.0M     0     16.0M    0% /media/ram
tmpfs                     50.0M    12.0K   50.0M     0% /var/volatile
tmpfs                     93.9M     0     92.9M     0% /run/user/1000
/dev/mmcblk0p1            119.7M    3.1M   116.6M     3% /run/media/mmcblk0p1
```

- /dev/root : The root filesystem, mounted to the root directory.
- tmpfs : in-memory virtual filesystem, mounted under different directories.
- devtmpfs : Used to create dev for the system.

3) Performance test for eMMC

The performance test mainly tests the reading and writing speed of eMMC to files under linux system. The following will introduce the reading and writing performance test using iotop command.

● Read and write tests

Use the following command for reading and writing tests:

```
root@myd-am62x:~# iotop -e -l -a -s 100M -r 1M -i 0 -i 1 -i 2 -b emmc.xls
```

```
iotop: Performance Test of File I/O
Version $Revision: $3.489
Compiled for 64 bit mode.
```



```

Build: linux
slightly
File size set to 102400 kB
Record Size 1024 kB
Command line used: iozone -e -l -a -s 100M -r 1M -i 0 -i 1 -i 2
Output is in kBytes/sec
Time Resolution = 0.000001 seconds.
Processor cache size set to 1024 kBytes.
Processor cache line size set to 32 bytes.
File stride size set to 17 * record size.

                                random  rand
om   bkwd   record   stride
      kB reflen   write rewrite   read   reread   read   write
    read  rewrite   read  fwrite frewrite   fread  freread
      102400 1024 36526 37075 43635 43882 43683 36825

iozone test complete.
Excel output is below:
"Writer report"
    "1024"
"102400" 35996

"Re-writer report"
    "1024"
"102400" 36536

"Reader report"
    "1024"
"102400" 44490

"Re-Reader report"
    "1024"
"102400" 44549

```

"Random read report"

"1024"

"102400" 44286

"Random write report"

"1024"

"102400" 36476

In this command, -e means flush (fsync,fflush) is included when calculating the time, and -l means VxFS VX_DIRECT is used for all file operations. Tells the VxFS filesystem that all operations on files will be performed directly on disk across the cache. The -a flag is used to use fully automatic mode. Generate a report including all test operations, using block sizes from 4k to 16M and file sizes from 64k to 512M. , -s specifies the test file size, -r specifies the test block size, and -i specifies which test to run. (0=write/rewrite, 1=read/re-read, 2=random-read/write, 3=Read-backwards, 4=Re-write-record, 5=stride-read, 6=fwrite/re-fwrite, 7=fread/Re-fread, 8=random mix, 9=pwrite/Re-pwrite, 10=pread/Re-pread, 11=pwritev/Re-pwritev, 12=preadv/Re-preadv) Here we use read and write and random read/ write option tests, -b means create a file in binary format compatible with Excel.

The results of this command are all Kbyte/s, taking the above result as an example, the result is as follows:

Table 2-1.EMMC read and write speed results

Test items	Writer	Re-writer	Reader	Re-Reader	Random read	Random write
Test results (Kb/s)	35996	36536	44490	44549	44286	36476

4) Checking the Lifespan of eMMC

The lifespan of eMMC refers to the time period or total quantity during which an embedded MultiMediaCard (eMMC) storage device can reliably operate under normal usage conditions. The eMMC lifespan is typically influenced by various factors, including but not limited to usage frequency, write/read quantity

tities, ambient temperature, and flash memory type. When the eMMC storage approaches its end of life, issues such as data loss or decreased device performance may occur. Therefore, it is important to consider the lifespan of the storage when managing and backing up data. The eMMC lifespan refers to the time or total data write/read quantity that a storage device can reliably operate under certain conditions, making it one of the crucial factors to be monitored and managed during device design and usage.

To check the approximate remaining lifespan of eMMC using the following test command:

- **Check eMMC lifespan**

```
root@myd-am62x:~# mmc extcsd read /dev/mmcblk0 | grep Life
eMMC Life Time Estimation A [EXT_CSD_DEVICE_LIFE_TIME_EST_TYP_A]: 0x01
eMMC Life Time Estimation B [EXT_CSD_DEVICE_LIFE_TIME_EST_TYP_B]: 0x01
```

Here's an explanation of these parameters:

- eMMC Life Time Estimation A: Device lifespan estimation type A, estimating the lifespan for MLC user partitions' erase blocks.
- eMMC Life Time Estimation B: Device lifespan estimation type B, estimating the lifespan for SLC boot partitions' erase blocks.

The values for these parameters are referenced in increments of 10%. For example, 0x02 indicates that 10%-20% of the device's lifespan has been used.

Alternatively, you can use the following command to view:

```
root@myd-am62x:~# cat /sys/devices/platform/bus@f0000/fa10000.mmc/mmc_host/mmc0/mmc0:0001/life_time
0x01 0x01
```

The two displayed values represent the lifespan estimates for types A and B

- **Standard Values for Lifespan Estimation Type B Parameters**

The standard values are as shown in Table 2-2.

Table 2-2 Design Life Estimation Type B Standard

Value	Description
0x00	Not defined
0x01	0%-10% device life time used

0x02	10%-20% device life time used
0x03	20%-30% device life time used
0x04	30%-40% device life time used
0x05	40%-50% device life time used
0x06	50%-60% device life time used
0x07	60%-70% device life time used
0x08	70%-80% device life time used
0x09	80%-90% device life time used
0x0A	90%-100% device life time used
0x0B	Exceeded its maximum estimated device life time
Others	Reserved

2.6. RTC

RTC (real-time clock) itself is a clock that is used to record the Real time. When the software system is shut down, the system time is retained and the time is synchronized into the software system after the system is turned on again. The following test writes the system time to the RTC, reads the RTC time and sets it as the system time and carries out the test of time power down retention.

- **View the system RTC device**

```
root@myd-am62x:~# ls /dev/rtc* -al
lrwxrwxrwx    1 root    root          4 Jan  1  1970 /dev/rtc -> rtc0
crw-----    1 root    root       251,  0 Jan  1  1970 /dev/rtc0
```

rtc belongs to linux devices, and its device node rtc0 is available for user operation under /dev.

- **Setting the system time**

After setting the system time to Tue Sep 8 09:07:30 UTC 2020.

```
root@myd-am62x:~# date 092010002023.30
Wed Sep 20 10:00:30 UTC 2023
```

You may encounter the following problems while performing the steps above:

```
root@myd-am62x:~# date 092010002023.30
Wed Sep 20 10:00:30 UTC 2023
```

```
root@myd-am62x:~# [1055.511004] systemd-journald[153]: Time jumped back
wards, rotating.
```

```
root@myd-am62x:~# date
```

```
Wed Sep 20 03:01:47 UTC 2023
```

You can turn off network time synchronization at this point using the following command:

```
root@myd-am62x:~# timedatectl set-ntp false
```

Just re-use the date command to set the time.

- **Write the system time to RTC**

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

```
root@myd-am62x:~# hwclock -w
```

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

```
root@myd-am62x:~# hwclock -r
```

```
Wed Sep 20 03:03:35 2023 0.000000 seconds
```

- **Power down keep RTC time**

Turn the device off and disconnect the power, after about 2 minutes, power it back on. Check the RTC time and system time.

```
root@myd-am62x:~# hwclock -r
```

```
Wed Sep 20 03:11:27 2023 0.000000 seconds
```

After rebooting, the RTC time and system time viewed have increased by about 2 minutes compared to the previous setting, indicating that the RTC is working properly. If you need to test the accuracy of the RTC in detail, you can extend the power off time such as 24 hours to test the difference between the RTC time and the standard time.

- **Synchronize the system time with the RTC time**

```
root@myd-am62x:~# hwclock -s
```

```
root@myd-am62x:~# date
```

```
Wed Sep 20 03:11:52 UTC 2023
```

```
root@myd-am62x:~#
```

If you add the `hwclock-s` command to the startup script, you can ensure that the system time and RTC time can be synchronized every time you start.

2.7. Watchdog

Linux kernel contains Watchdog subsystem. In the process of hardware design, the Watchdog function can be realized by using internal watchdog timer or external watchdog chip to monitor the operation of the system. When the system fails to feed the dog in an abnormal situation, the system can be automatically reset.

- magic close feature: When the user wants to stop the watchdog, write the character "V" to the `wdt` node to stop the watchdog. MYD-YM62X Watchdog driver does not support closing watchdog in this way.
- 32-bit watchdog timer with programmable termination value
- Flexible watchdog clock with 16-bit predivider
- Powerful Watchdog refresh control that supports:
 - Window-based watchdog refresh
 - Enhanced refresh sequence
- Supports overflow interrupts and illegal operation interrupts
- Internal and external reset requests are supported

This section demonstrates the use of watchdog, simulates a kernel crash to test the watchdog system reset functionality, and provides examples of modifications to set watchdog timeouts.

1) Application Program Testing Watchdog

● Set Watchdog Timeout:

```
ioctl(wdt_fd, WDIOC_SETTIMEOUT, &time_out);
```

The above code is a reference for setting the current timeout of the watchdog. The variable "fd" represents the file handle of the watchdog device.

● Watchdog Application Testing:

```
root@myd-am62x:~# ./watchdog_test
```

Usage: wdt_driver_test <timeout> <sleep> <test>
 timeout: value in seconds to cause wdt timeout/reset
 sleep: value in seconds to service the wdt
 test: 0 - Service wdt with ioctl(), 1 - with write()

Note that when using this test program, the timeout value cannot be less than or equal to the default reset time of the watchdog set by the kernel, which is 10 seconds. Therefore, the timeout value must be set to a value greater than 10 seconds. The corresponding file and location in the kernel are as follows:

```
beste@system1:~/myir-ti-linux$ vi drivers/watchdog/rti_wdt.c
// SPDX-License-Identifier: GPL-2.0
/*
 * Watchdog driver for the K3 RTI module
 *
 * (c) Copyright 2019-2020 Texas Instruments Inc.
 * All rights reserved.
 */

#include <linux/clock.h>
#include <linux/device.h>
#include <linux/err.h>
#include <linux/io.h>
#include <linux/kernel.h>
#include <linux/mod_devicetable.h>
#include <linux/module.h>
#include <linux/moduleparam.h>
#include <linux/platform_device.h>
#include <linux/pm_runtime.h>
#include <linux/types.h>
#include <linux/watchdog.h>
//Here, the default reset time of the watchdog is set.
#define DEFAULT_HEARTBEAT 10
```



```
/* Max heartbeat is calculated at 32kHz source clock */
#define MAX_HEARTBEAT 1000
```

Then, run the watchdog application with a timeout of 11 seconds, and feed the watchdog every 1 second.

```
root@myd-am62x:~# ./watchdog_test 11 1
Starting wdt_driver (time_out: 11, sleep: 1)
options = 0x81a0,id = K3 RTI Watchdog
Trying to set time_out value=11 seconds
The actual time_out was set to 11 seconds
Now reading back -- The time_out is 11 seconds
food watchdog, count = 0
food watchdog, count = 1
food watchdog, count = 2
food watchdog, count = 3
food watchdog, count = 4
```

If the above 1 second is changed to approximately 4 seconds, which exceeds the required 4-second feeding time, the development board will restart.

```
root@myd-am62x:~# ./watchdog_test 11 12
Starting wdt_driver (time_out: 11, sleep: 12)
options = 0x81a0,id = K3 RTI Watchdog
Trying to set time_out value=11 seconds
The actual time_out was set to 11 seconds
Now reading back -- The time_out is 11 seconds
food watchdog, count = 0
U-Boot SPL 2023.04-gf4e7b03f15 (Sep 15 2023 - 01:45:12 +0000)
SYSFW ABI: 3.1 (firmware rev 0x0009 '9.0.5--v09.00.05 (Kool Koala)')
SPL initial stack usage: 13376 bytes
Trying to boot from MMC2.
```



3. Peripheral interface

3.1. GPIO

GPIO is tested through the filesystem sysfs interface. The following content will take the P10 pin 540 of the NCA9555 chip as an example to explain the use of GPIO.

1) Check the current usage of gpio

Use the following command to see which GPIOs are currently in use.

```
root@myd-am62x:~# cat /sys/kernel/debug/gpio
gpiochip3: GPIOs 362-413, parent: platform/601000.gpio, 601000.gpio:
gpio-412 (                |vbus                ) out lo
gpio-413 (                |rts                 ) out lo

gpiochip2: GPIOs 414-505, parent: platform/600000.gpio, 600000.gpio:
gpio-426 (                |rts                 ) out lo
gpio-450 (                |GPIO Key USER1     ) in  hi ACTIVE LOW
gpio-485 (                |int                ) in  hi

gpiochip1: GPIOs 506-529, parent: platform/4201000.gpio, 4201000.gpio:
gpio-509 (                |csi1-io            ) out lo
gpio-510 (                |regulator-6        ) out lo
gpio-521 (                |regulator-4        ) out lo
gpio-522 (                |am62-sk:d53        ) out lo ACTIVE LOW
gpiochip0: GPIOs 530-545, parent: i2c/2-0020, can sleep:
gpio-531 (                |reset              ) out hi ACTIVE LOW
gpio-534 (                |am62-sk:d54        ) out lo ACTIVE LOW
gpio-535 (                |PHY reset          ) out hi ACTIVE LOW
gpio-536 (                |PHY reset          ) out hi ACTIVE LOW
```

Gpiochip0: GPIOs as NCA9555 chip 530-545, P10 is 530 + 10 get corresponding gpio540 P10 pin value.

2) Export GPIO

```
root@myd-am62x:~# echo 540 > /sys/class/gpio/export
```

A directory named gpio500 will be generated in /sys/class/gpio/.

```
root@myd-am62x:~# ls /sys/class/gpio/
export      gpio540      gpiochip362  gpiochip414  gpiochip506  gpiochip5
30  unexport
```

3) Set/View GPIO orientation

- **Setting the input**

```
root@myd-am62x:~# echo "in" > /sys/class/gpio/gpio540/direction
```

- **Setting the output**

```
root@myd-am62x:~# echo "out" > /sys/class/gpio/gpio540/direction
```

- **View the gpio direction**

```
root@myd-am62x:~# cat /sys/class/gpio/gpio540/direction
out
```

Return in for input and return out for output.

4) Set/view the value of the GPIO

- **Set output low**

```
root@myd-am62x:~# echo "0" > /sys/class/gpio/gpio540/value
```

- **Set the output high**

```
root@myd-am62x:~# echo "1" > /sys/class/gpio/gpio540/value
```

- **View the value of gpio**

```
root@myd-am62x:~# cat /sys/class/gpio/gpio540/value
1
```

3.2. LED

The Linux system provides a separate subsystem to facilitate the operation of LED devices from the user space. The subsystem provides the operation interface for 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 device. Let's test the LED by reading and writing sysfs. The following commands are general commands, which are also the general method of controlling the LED.

1) The directory to operate LED is /sys/class/leds. The contents of the directory are:

```
root@myd-am62x:/sys/class/leds# ls
am62-sk:d53  am62-sk:d54  mmc0::      mmc1::      mmc2::
```

2) Test the LED with the D53 lamp as an example

- Read the LED status of the D53 lamp

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

```
root@myd-am62x:~# cat /sys/class/leds/am62-sk:d53/brightness
1
```

The blue LED is currently on.

- Turn off the LED

```
root@myd-am62x:~# echo 0 > /sys/class/leds/am62-sk:d53/brightness
```

- Light up the leds

```
root@myd-am62x:~# echo 1 > /sys/class/leds/am62-sk:d53/brightness
```

- Enable LED trigger mode

When the "timer" departure mode is enabled, the LED blinks at a 1Hz cycle by default with a duty cycle of 50%.

```
root@myd-am62x:~# echo "timer" > /sys/class/leds/am62-sk:d53/trigger
```

- Change the on/off duty cycle when the LED lights are flashing

The led blinking period and bright-off duty cycle can be adjusted by adjusting the delay_on and delay_off attributes under the LED, such as:

```
root@myd-am62x:~# echo "100" > /sys/class/leds/am62-sk\:d53/delay_on  
root@myd-am62x:~# echo "500" > /sys/class/leds/am62-sk\:d53/delay_off
```

3.3. Key(press the key)

Linux's /dev/input/eventx device can be used to easily debug input devices such as mouse, keyboard, touchpad, etc. This section is devoted to testing key.

Use the hexdump command as well as the dmesg command to see if the key is responsive. MYD-YM62X has three buttons, S1 is the MCU reset button; S2 is the Soc reset button; S3 is the user key, already configured in the device tree.

1) Key test

- **View the corresponding input device event information**

```
root@myd-am62x:~# ls /proc/bus/input/
devices  handlers
root@myd-am62x:~# cat /proc/bus/input/devices
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/input0
U: Uniq=
H: Handlers=event0
B: PROP=0
B: EV=100003
B: KEY=1 0 0 0 0
```

The device event for gpio-keys is event0.

- **dump key information**

Execute the following command and press S3. The serial terminal will print the following information:

```
root@myd-am62x:~# hexdump /dev/input/event0
00000000 6f8d 650a 0000 0000 a838 0007 0000 0000
00000010 0001 0100 0001 0000 6f8d 650a 0000 0000
00000020 a838 0007 0000 0000 0000 0000 0000 0000
```

```
0000030 6f8d 650a 0000 0000 ae56 0009 0000 0000
0000040 0001 0100 0000 0000 6f8d 650a 0000 0000
0000050 ae56 0009 0000 0000 0000 0000 0000 0000
```

Each time S3 is pressed, the current terminal will print the current event code value, that is, the key is normal.

3.4. RS485

This routine demonstrates how to use Linux API test development board RS485 send and receive data function, RS485_1 device for J13 8,9 port, for UART 6, device node for ttyS1; RS485_2 device for J13 port 11,12, for UART4, device node for ttyS0. The following will be tested with RS485_1 and RS485_2 connected to each other, and the hardware interface configuration is as follows:

Table 3-1.RS485 interface configuration

	MYD-YM62X	MYD-YM62X
Interface	RS485_1	RS485_2
Device node	ttyS1	ttyS0
Test software	tty_test	tty_test

Connect 485A1 and 485B2 of RS485_1 to 485A2 and 485B2 of RS485_2, respectively.

1) Receive and send RS485-1 under the development board

First enter the following command to see the instructions for tty_test:

```
root@myd-am62x:~# ./tty_test -h
Usage: ./tty_test [options]

Version 1.0
Options:
-d | --device name    tty device name: /dev/ttyO0
-m | --mode mode      tty mode. 0: RS232, 1: RS485 default mode: 0
-f | --flow           flow control
-b | --baudrate baudrate  set baudrate, default baudrate:9600
-l | --loop           operate circularly, default not operate circularly!
```



```
-w | --write frame      frame string. such as: 0123456789
-h | --help             Print this message
root@myd-am62x:~#
```

Run the following command to put RS485_2 in the background waiting to be received:

```
root@myd-am62x:~# ./tty_test -m 1 -b 115200 -l -d /dev/ttyS0 &
[1] 694
```

Run the following command to have RS485_1 send the data, using the -l loop to send, you can see that if sent correctly and received correctly, RS485_2 will feed back the data received by RECV. Press Ctrl+C to stop when you confirm it is correct.

```
root@myd-am62x:~# ./tty_test -m 1 -b 115200 -l -d /dev/ttyS0 -w 1234567890
SEND:1234567890
RECV:1234567890, total:10
SEND:1234567890
RECV:1234567890, total:10
SEND:1234567890
RECV:1234567890, total:10 ^C
```

Execute fg command to return the RS485_2 received in the background to the foreground, and then press Ctrl+C to stop.

```
root@myd-am62x:~# fg
./tty_test -m 1 -l -d /dev/ttyS0 -b 2400
^C
```

3.5. CAN

In this section, the commonly used Linux system cansend and candump commands are used to conduct the communication test of SocketCAN. The development board itself has a standard CAN and an M core M_CAN. The following will introduce the test process of using the standard CAN interface of the same type of development board for docking test.

Firstly, the 4 and 5 ports of J13, that is, CANL2 and CANH2, are connected to the corresponding interfaces of the same type of development board.

1) Initialize the CAN network interface

- **Set the CAN baud rate**

Using canfd requires setting the baud rate first and enabling the CAN network interface. Refer to the following command to set the arbitration baud rate of the two boards to 1M and the data baud rate device to 4M, respectively, and enable the canfd function.

```
root@myd-am62x:~# ip link set can0 down
root@myd-am62x:~# ip link set can0 up type can bitrate 1000000 dbitrates 4000000 fd on
```

At this point, the canfd functionality is ready to be used.

2) Sending and Receiving Data

Set CAN0 on one of the development boards as the receiver and wait for data to be received using the candump command:

```
root@myd-am62x:~# candump can0 -L
```

Set CAN0 of another development board as the sender and use the cansend command to send data for testing:

```
root@myd-am62x:~# cansend can0 00 f # # 0. A1, a2, a3, a4, a5, a6, a7, a8.
A9.10.11.12.13.14.15.16.17.18.19.20.21.23.24.25.26.27.28.29.30.31.32.33.34.35.36.3
7.38.39 . 40.41.42.43.44.45.46.47.48.49.50.51.52.53.54.55.56.57.58.59.60.61.62.63.6
4
root@myd-am62x:~# cansend can0 00 f # # 0. A1, a2, a3, a4, a5, a6, a7, a8.
A9.10.11.12.13.14.15.16.17.18.19.20.21.23.24.25.26.27.28.29.30.31.32.33.34.35.36.3
7.38.39 . 40.41.42.43.44.45.46.47.48.49.50.51.52.53.54.55.56.57.58.59.60.61.62.63.6
4
root@myd-am62x:~# cansend can0 00 f # # 0. A1, a2, a3, a4, a5, a6, a7, a8.
A9.10.11.12.13.14.15.16.17.18.19.20.21.23.24.25.26.27.28.29.30.31.32.33.34.35.36.3
7.38.39 . 40.41.42.43.44.45.46.47.48.49.50.51.52.53.54.55.56.57.58.59.60.61.62.63.6
4
```

Observing the print of the development board at the receiving end, you can see the successfully received data:

```
root@myd-am62x:~# candump can0 -L
(0000000400.823368) can0 00F##4A1A2A3A4A5A6A7A8A9101112131415161718
1920212324252627282930313233343536373839404142434445464748495051525
35455565758 59606162636400
(0000000401.913890) can0 00F##4A1A2A3A4A5A6A7A8A9101112131415161718
1920212324252627282930313233343536373839404142434445464748495051525
35455565758 59606162636400
(0000000402.217901) can0 00F##4A1A2A3A4A5A6A7A8A9101112131415161718
1920212324252627282930313233343536373839404142434445464748495051525
35455565758 59606162636400
```

3) Configure can0 interface

After sending and receiving CAN data, the CAN device details and sending and receiving statistics are displayed, where the value of "clock" represents the clock of the can, the value of "drop" represents packet loss, the value of "overrun" represents overflow, and the value of "error" represents bus error.

```
root@myd-am62x:~# ip -details -statistics link show can0
4: can0: <NOARP,UP,LOWER_UP,ECHO> mtu 72 qdisc pfifo_fast state UP mode
  DEFAULT group default qlen 10
    link/can  promiscuity 0 minmtu 0 maxmtu 0
    can <FD> state ERROR-ACTIVE (berr-counter tx 0 rx 0) restart-ms 0
      bitrate 1000000 sample-point 0.750
      tq 12 prop-seg 29 phase-seg1 30 phase-seg2 20 sjw 1 brp 1
      m_can: tseg1 2.. 256 tseg2 2.. 128 sjw 1.. 128 brp 1.. 512 brp_inc 1
      dbitrte 4000000 dsample-point 0.750
      dtq 12 dprop-seg 7 dphase-seg1 7 dphase-seg2 5 dsjw 1 dbrp 1
      m_can: dtseg1 1.. 32 dtseg2 1.. 16 dsjw 1.. 16 dbrp 1.. 32 dbrp_inc
1
      clock 80000000
      re-started bus-errors arbit-lost error-warn error-pass bus-off
```

```

0          0          0          0          0          0          nu
mtxqueues 1 numrxqueues 1 gso_max_size 65536 gso_max_segs 65535 parent
bus platform parentdev 20701000.can
  RX:  bytes packets errors dropped missed  mcast
      201 6 0 0 0
  TX:  bytes packets errors dropped carrier collsns
      192 3 0 0 0 0
root@myd-am62x:~#

```

4) Test of M_CAN

The M_CAN test CAN connect the 1 and 2 ports of J13 to the 4 and 5 ports of standard CAN for communication test, that is, CANL1 to CANL2, CANH1 to CANH2.

For the specific test process, please read the relevant chapters of the application note "MYD-YM62X M4 Coprocessor Application Development Notes".

3.6. USB

This section verifies the feasibility of USB Host driver through relevant commands or hotplug, USB HUB, and realizes the function of reading and writing U disk and usb enumeration.

1) Check the print information of the inserted usb

- View the USB device information

Connect the USB key to the development board USB Host interface (J16), and the kernel prompts the following information:

```

root@myd-am62x:~# [6229.728605] usb 2-1.1: new high-speed USB device number 4 using xhci-hcd
[6229.941933] USB-Storage 2-1.1:1.0: USB Mass Storage device detected
[6229.949370] scsi host0: usb-storage 2-1.1:1.0
[6230.977538] scsi 0:0:0:0: Direct-Access aigo U330 2.00PQ :0 ANSI: 4

```

```
[6230.993399] sd 0:0:0:0: [sda] 61440000 512-byte logical blocks: (31.5 GB/29.3 GiB)
[6231.005361] sd 0:0:0:0: [sda] Write Protect is off
[6231.010560] sd 0:0:0:0: [sda] No Caching mode page found
[6231.016029] sd 0:0:0:0: [sda] Assuming drive cache: write through
[6231.026129] sda: sda1
[6231.029480] sd 0:0:0:0: [sda] Attached SCSI removable disk
```

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

2) U disk mount read and write

- **Mount the device**

```
root@myd-am62x:~# mount /dev/sda1 /mnt/
```

- **Read the file**

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

```
root@myd-am62x:~# ls /mnt
test.txt
root@myd-am62x:~# cat /mnt/test.txt
hello world!
```

- **Writing a file**

```
root@myd-am62x:~# touch test.txt
root@myd-am62x:~# echo "hello world !!!" > test.txt
root@myd-am62x:~# cp test.txt /mnt
root@myd-am62x:~# cat /mnt/test.txt
hello world !!!
```

After you have written the file, you need to run the sync command to ensure that the data is completely written to the USB key before you can unmount the device.

3) Uninstall the USB key

- **Uninstall device command**

```
root@myd-am62x:~# umount /mnt
```

3.7. typeC interface

In this section, the host function of typeC port is tested by connecting the U disk with the OTG connection of typeC to USB, and the device function of typeC port is tested by connecting the PC with the typeC to USB public port line.

1) Test the host function

- **Connect the OTG cable to the development board and to the USB key**

```
root@myd-am62x:~# [281.493596] sd 0:0:0:0: [sda] 61440000 512-byte logical
blocks: (31.5 GB/29.3 GiB)
[281.506449] sd 0:0:0:0: [sda] Write Protect is off
[281.519666] sd 0:0:0:0: [sda] No Caching mode page found
[281.525166] sd 0:0:0:0: [sda] Assuming drive cache: write through
[281.534353] sda: sda1
[281.537612] sd 0:0:0:0: [sda] Attached SCSI removable disk
root@myd-am62x:~# cat /sys/kernel/debug/usb/31000000.usb/mode
host
```

- **Mount the USB key and read and write the test**

```
root@myd-am62x:~# mount /dev/sda /mnt
root@myd-am62x:~# cd /mnt
root@MYD-YM62X:/mnt# echo "this is a test file" > test.txt
root@MYD-YM62X:/mnt# cat test.txt
this is a test file
root@MYD-YM62X:/mnt#
```

2) Test the devic function

First connect the development board to the host PC using a typeC cable to see that mode has indeed automatically switched to device mode:

```
root@myd-am62x:~# cat /sys/kernel/debug/usb/31000000.usb/mode
device
```

Load the `g_mass_storage` driver to simulate the development board as a USB device:

```
root@myd-am62x:~# modprobe g_mass_storage stall=0 file=/dev/mmcbk0p1
removable=1 iSerialNumber=1234
[127.144992] Mass Storage Function, version: 2009/09/11
[127.150337] LUN: removable file: (no medium)
[127.155077] LUN: removable file: /dev/mmcbk0p1
[127.159642] Number of LUNs=1
[127.162778] g_mass_storage gadget.0: Mass Storage Gadget, version: 2009/09/11
[127.169936] g_mass_storage gadget.0: g_mass_storage ready
root@myd-am62x:~#
```

As you can see, `/dev/mmcbk0p1` has been emulated as a USB flash drive, check the PC display:



Figure 3.1 Emulating a USB drive

3.8. Micro SD card

Micro SD Card, formerly known as Trans-flash Card (TF card), Micro SD card is a very small flash memory card. Compared with the standard SD card, the Micro SD card is more compact in appearance and is the smallest SD card in the SD card type. Although the external size and interface shape of the Micro SD card are different from the original SD card, the interface specification remains the same to ensure compatibility. If Micro SD is inserted into a specific adapter card, it can be used as a standard SD card. SD card has become the most widely used memory card in consumer digital devices. It is a multifunctional memory card with large capacity, high performance and security. Mi

cro SD card generally has 9 pins on the back, including 4 data lines, supporting 1bit/4bit two kinds of data transmission width.

This section will explain the steps and methods of viewing and operating TF card under Linux system.

1) Check the TF card capacity

The TF card partition information and capacity can be queried by the `fdisk -l` command.

```
root@myd-am62x:~# fdisk -l
slightly
Disk /dev/mmcb1k1: 29.72 GiB, 31914983424 bytes, 62333952 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x798ed1d5

Device            Boot  Start      End  Sectors  Size Id Type
/dev/mmcb1k1p1  *           2048   264191   262144   128M  c W95 FAT32 (LBA)
/dev/mmcb1k1p2  264192  5933789  5669598   2.7G  83 Linux
[1422.906481] EXT4-fs (mmcb1k1p2): mounted filesystem with ordered data mode. Quota mode: none.
```

➤ /dev/mmcb1k1: Device node for TF card

2) View TF card partition information

Through the `df` command, you can query TF card partition information, usage, mount directory and other information.

```
root@myd-am62x:~# df -h
Filesystem            Size      Used Available Use% Mounted on
slightly
/dev/mmcb1k1p1  127.7M  127.7M    0   100% /run/media/boot-mmcb1k1p1
/dev/mmcb1k1p2  2.5G   1.5G  926.8M   62% /run/media/root-mmcb1k1p2
```

➤ /dev/root : The root filesystem, mounted to the root directory

- tmpfs : in-memory virtual filesystem, mounted under different directories
- devtmpfs : Used to create dev for the system

3) Performance test for TF cards

The performance test mainly tests the TF card in the linux system to read and write the file speed, here uses the iotest command test. Mount the TF card partition that needs to be tested, here the last partition /dev/mmcblk1p2 is taken as an example, and the mount directory is /mnt.

● Read and write tests

The following command iotest is used to test only the read and write speed as well as the rereading and rewriting speed. Please refer to the performance testing section of eMMC in eMMC in Section 2.5 for the meaning of the parameters.

```
root@MYD-YM62X:/mnt# iotest -e -l -a -s 100M -r 1M -i 0 -i 1
```

```
iotest: Performance Test of File I/O
```

```
Version $Revision: $3.489
```

```
Compiled for 64 bit mode.
```

```
Build: linux
```

```
Contributors:William Norcott, Don Capps, Isom Crawford, Kirby Collins
```

```
Al Slater, Scott Rhine, Mike Wisner, Ken Goss
```

```
Steve Landherr, Brad Smith, Mark Kelly, Dr. Alain CYR,
```

```
Randy Dunlap, Mark Montague, Dan Million, Gavin Brebner,
```

```
er,
```

```
Jean-Marc Zucconi, Jeff Blomberg, Benny Halevy, Dave Boone,
```

```
oone,
```

```
Erik Habbinga, Kris Strecker, Walter Wong, Joshua Root,
```

```
Fabrice Bacchella, Zhenghua Xue, Qin Li, Darren Sawyer,
```

```
Vangel Bojaxhi, Ben England, Vikentsi Lapa,
```

```
Alexey Skidanov, Sudhir Kumar.
```

Run began: Fri Jan 1 00:03:17 2066

Include fsync in write timing

O_DIRECT feature enabled

Auto Mode

File size set to 102400 kB

Record Size 1024 kB

Command line used: iotest -e -l -a -s 100M -r 1M -i 0 -i 1

Output is in kBytes/sec

Time Resolution = 0.000001 seconds.

Processor cache size set to 1024 kBytes.

Processor cache line size set to 32 bytes.

File stride size set to 17 * record size.

kB	reclen	write	rewrite	read	reread
102400	1024	7188	7248	22411	22542

random random bkwd record stride

read	write	read	rewrite	read	fwrite	frewrite	fread	freread

iotest test complete.

According to the results, the read and write speed data of the Sd card was obtained, the read speed was 7188Kb/s, the read speed was 22411Kb/s, the rewrite speed was 7248Kb/s, and the reread speed was 22542Kb/s.

3.9. Display

This section will be testing the display section. MYD-YM62X has 1 hdmi display, 1 lvds output, the development board divides this 1 lvds output into two single LVDS J5 and J6, and dual LVDS J8.

1) HDMI Display

First, use HDMI to connect the J29 interface of the bottom board and the monitor, and observe whether the HDMI can output normally after starting, wh

either there is any abnormal phenomenon such as color difference, jitter, offset and so on. When the display is normal, plug and unplug the HDMI cable many times quickly, and check whether there is a normal output each time. Check the resolution of the current display, such as the current default display resolution of 1920*1080.

```
root@myd-am62x:~# cat /sys/kernel/debug/dri/0/state
plane[31]: plane-0
    crtc=crtc-0
    fb=55
        allocated by = weston
        refcount=2
        format=XR24 little-endian (0x34325258)
        modifier=0x0
        size=1920x1080
        layers:
            size[0]=1920x1080
            pitch[0]=7680
            offset[0]=0
            obj[0]:
                name=0
                refcount=3
                start=001017bb
                size=8294400
                imported=no
                dma_addr=0x00000000b0700000
                vaddr=0000000032003f42
        crtc-pos=1920x1080+0+0
        SRC - x1080.000000 pos = 1920.000000 + 0.000000 + 0.000000
        rotation=1
        normalized-zpos=0
        color-encoding=ITU-R BT.601 YCbCr
        color-range=YCbCr full range
plane[41]: plane-1
```

```

crtc=(null)
fb=0
crtc-pos=0x0+0+0
SRC - x0.000000 pos = 0.000000 + 0.000000 + 0.000000
rotation=1
normalized-zpos=0
color-encoding=ITU-R BT.601 YCbCr
color-range=YCbCr full range
crtc[38]: crtc-0
enable=1
active=1
self_refresh_active=0
planes_changed=1
mode_changed=0
active_changed=0
connectors_changed=0
color_mgmt_changed=0
plane_mask=1
connector_mask=1
encoder_mask=1
mode: "1920x1080": 60 148500 1920 2008 2052 2200 1080 1082 1087
1125 0x40 0x5
connector[40]: HDMI-A-1
crtc=crtc-0
self_refresh_aware=0
max_requested_bpc=0
root@myd-am62x:~#

```

2) LVDS display

The LVDS display is suitable for the MY-LVDS070C screen, which is a 7-inch capacitive touch screen with a resolution of 1024*600. The screen is first flexibly connected through 40pin. The following single-channel LVDS display will take J6 interface as an example.

Must first be /run/media/mmcblk0p1 uEnv. TXT file, modified to the following content.

```
root@myd-am62x:~# vi /run/media/mmcblk0p1/uEnv.txt

# This uEnv.txt file can contain additional environment settings that you
# want to set in U-Boot at boot time. This can be simple variables such
# as the serverip or custom variables. The format of this file is:
#   variable=value
# NOTE: This file will be evaluated after the bootcmd is run and the
#       bootcmd must be set to load this file if it exists (this is the
#       default on all newer U-Boot images. This also means that some
#       variables such as bootdelay cannot be changed by this file since
#       it is not evaluated until the bootcmd is run.
#To enable the corresponding feature, please remove the comment on the li
ne below the description of the functionality for "name_overlays".
#Attention: Only one of the following features can be enabled at a time.
#HDMI audio playback.
#name_overlays=myir/myd-y62x-hdmi-audio.dtbo
#Dual-channel LVDS display
#name_overlays=myir/myd-y62x-lvds-dual.dtbo
#Single-channel LVDS display with J5 interface
#name_overlays=myir/myd-y62x-lvds.dtbo
#Single-channel LVDS display with J6 interface
name_overlays=myir/myd-y62x-lvds-j6.dtbo
```

Uncomment the line name_overlays=myir/myd-y62x-lvds-j6.dtbo to enable the use of J6's single-way LVDS interface, then type sync to save and restart the development board.

After starting, you can see the screen display normally.

Other interface displays refer to the above operation, and unannotate other functions, but only one can be opened at the same time.

3.10. Touch Panel

MYD-YM62X series development board supports capacitated touch, MYIR Electronics provides 7 inch touch LCD accessories, see Table 1-1. You can purchase accessories according to your actual needs. The capacitive screen is more sensitive in use and rarely has problems. If the touch screen can be tapped, it is fine. Plus, capacitive screens don't need to be precise. Because according to the principle of the capacitive screen, the capacitive screen can accurately identify the position of the finger and the screen in use, with high sensitivity. If we click the phenomenon that the software is not selected in use, there is generally only one situation: there is a problem with the screen. Let's do a simple test through the touch function of the evtest command screen.

1) Connect the touch LCD monitor of MY LVDS070C to the development board J5 or J6 according to Section 3.9, and pay attention to modify the uEnv file according to Section 3.9, and connect the jumper cap on the base board. If it is J5 interface, connect JP3, and J6 interface, connect JP2.

2) The terminal executes "evtest" to enter the test interface.

Select the test device as a touch screen device, which means that the input is interrupted. Select '1' in the test interface and press Enter to start the test:

```
root@myd-am62x:~# evtest
No device specified, trying to scan all of /dev/input/event*
Available devices:
/dev/input/event0:      gpio-keys
/dev/input/event1:      generic ft5x06 (79)
Select the device event number [0-1]: 1
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      50
Min        0
Max        1023
Event code 1 (ABS_Y)
Value      22
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)

It is recognized as an input interrupt device when connected to the LVDS.

3) Click the touch screen, and the terminal will print the corresponding information

```
Event: time 1695266775.747141, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID), value 17
```

```
Event: time 1695266775.747141, type 3 (EV_ABS), code 53 (ABS_MT_POSITION_X), value 252
```

```

Event: time 1695266775.747141, type 3 (EV_ABS), code 54 (ABS_MT_POSITION
_Y), value 337
Event: time 1695266775.747141, type 1 (EV_KEY), code 330 (BTN_TOUCH), val
ue 1
Event: time 1695266775.747141, type 3 (EV_ABS), code 0 (ABS_X), value 252
Event: time 1695266775.747141, type 3 (EV_ABS), code 1 (ABS_Y), value 337
Event: time, 1695266775.747141 -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- SYN_RE
PORT -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
Event: time 1695266775.805983, type 3 (EV_ABS), code 53 (ABS_MT_POSITION
_X), value 249
Event: time 1695266775.805983, type 3 (EV_ABS), code 54 (ABS_MT_POSITION
_Y), value 339
Event: time 1695266775.805983, type 3 (EV_ABS), code 0 (ABS_X), value 249
Event: time 1695266775.805983, type 3 (EV_ABS), code 1 (ABS_Y), value 339
Event: time, 1695266775.805983 -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- SYN_RE
PORT -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
Event: time 1695266775.823965, type 3 (EV_ABS), code 54 (ABS_MT_POSITION
_Y), value 338
Event: time 1695266775.823965, type 3 (EV_ABS), code 1 (ABS_Y), value 338
Event: time, 1695266775.823965 -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- SYN_RE
PORT -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
Event: time 1695266775.860706, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING
_ID), value-1
Event: time 1695266775.860706, type 1 (EV_KEY), code 330 (BTN_TOUCH), val
ue 0
Event: time, 1695266775.860706 -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- SYN_RE
PORT -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

```

It can be seen from the above that it mainly displays coordinate values and key values, and the specific information is as follows:

- EV_SYN: The synchronization event
- EV_KEY: The key press event, such as BTN_TOUCH indicates a touch button

- EV_ABS: Absolute coordinates, such as those reported by the touch screen
- BTN_TOUCH: Touch the button
- ABS_MT_TRACKING_ID indicates the beginning of collecting information, followed by an ABS_MT_TRACKING_ID indicates the end of collecting information

Single touch information is carried in ABS and sent in a certain order, such as:

- ABS_X: is the absolute coordinate X relative to the screen
- ABS_Y: is the absolute coordinate Y relative to the screen

Multi-touch messages are carried in ABS_MT and sent in a certain order, such as:

- ABS_MT_POSITION_X: This indicates the x-coordinate position of the center of the screen's contact surface.
- ABS_MT_POSITION_Y: Indicates the x-coordinate position of the center point of the screen contact

4. Networking and Communication

This chapter is designed to test and evaluate the connection, configuration, and use of network devices such as Ethernet, Wi-Fi, and mobile networks (optional). In many cases, Wi-Fi and Bluetooth are two modules in one, so this chapter will introduce the configuration and use of Bluetooth together.

4.1. Network Devices

The MYD_YM62X development board contains two Gigabit Ethernet interfaces and a Wi-Fi and Bluetooth 2-in-1 module called L297B. The configuration of these two network devices is introduced in the following.

4.1.1. Ethernet

There are many tools for network configuration in Linux, the common ones are net-tools, iproute2, systemd-networkd, network manager and connman, etc. These can be selected according to the actual needs when the system is customized. Here introduces several commonly used Ethernet manual temporary configuration and automatic permanent configuration.

1) Manually and temporarily configure Ethernet IP addresses

- Use ifconfig in the net-tools toolkit to manually configure the network

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

```
root@myd-am62x:~# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 1C:63:49:1A:6A:1F
          inet addr:192.168.30.216 Bcast:192.168.30.255 Mask:255.255.255.0
          inet6 addr: fe80::1e63:49ff:fe1a:6a1f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1111 errors:0 dropped:204 overruns:0 frame:0
          TX packets:78 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

RX bytes:109508 (106.9 KiB) TX bytes:8031 (7.8 KiB)

eth0 is the actual Ethernet device, which uses a random hardware MAC address by default

Here's how to manually configure the IP address 192.168.0.100 for eth0 with the following command:

```
root@myd-am62x:~# ifconfig eth0 192.168.0.100 netmask 255.255.255.0up
```

The above command manually config eth0 with an IP address of 192.168.0.100, a subnet mask of 255.255.255.0, and a default configured broadcast address of 192.168.0.255, which is activated with the up parameter, as follows:

```
root@myd-am62x:~# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 1C:63:49:1A:6A:1F
          inet addr:192.168.0.100 Bcast:192.168.0.255 Mask:255.255.255.0
          inet6 addr: fe80::1e63:49ff:fe1a:6a1f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1123 errors:0 dropped:204 overruns:0 frame:0
          TX packets:90 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:113598 (110.9 KiB) TX bytes:9840 (9.6 KiB)
```

- **Use the ip command in the iproute2 toolkit to manually configure the network**

The ifconfig command to manually set the IP address of the method can also use the IP addr and IP link for alternative, more information please view the instructions in the <https://wiki.linuxfoundation.org/networking/iproute2>.

```
root@myd-am62x:~# ip addr flush dev eth0
root@myd-am62x:~# ip addr add 192.168.0.101/24 brd + dev eth0
root@myd-am62x:~# ip link set eth0 up
```

If the IP address has been configured before, the ip address configured by using ip addr add will become the Secondary address, so first use ip addr flush to clear the previous address before configuration and activation. After the configuration is complete, the ip addr show command is used to view the eth0 information as follows:

```
root@myd-am62x:~# ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state U
P group default qlen 1000
    link/ether 1c:63:49:1a:6a:1f 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
```

2) Automatically permanently config Ethernet IP address

ip addresses configured with the ifconfig and IP commands are lost when the power is off. If you want to make the IP address permanent, you will need to modify the corresponding configuration file of the network management tool.

When the embedded Linux system uses busybox or ifupdown tools for network management, the network is automatically configured according to the system file /etc/network/interfaces. So in this case, you need to write the /etc/network/interfaces file in advance or modify the /etc/network/interfaces file to configure the default static IP address for eth0, or configure the IP address to be dynamically obtained through DHCP.

- **Use the ifupdown management tool to automatically and permanently configure the static IP address**

The /etc/network/interfaces file to automatically permanently configure static IP addresses using ifupdown or busybox is as follows:

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address 192.168.0.100
    netmask 255.255.255.0
    gateway 192.168.0.1
    dns-domain example.com
    dns-nameservers 192.168.0.11
```

In this way, every time eth0 is started, the default configuration IP address is 192.168.0.100, the subnet mask is 255.255.255.0, the gateway address is 192.168.0.1, and the dns server is 192.168.0.11.

- **Use ifupdown management tool configuration to dynamically obtain IP address**

Use ifupdown or busybox to configure the /etc/network/interfaces file for dynamically obtaining IP addresses as follows:

```
auto lo
iface lo inet loopback

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

In this way, eth0 will apply for the dynamic IP address to the DHCP server in the local area network and configure the IP address, gateway and DNS.

4.1.2. Wi-Fi

This section mainly introduces the configuration and use of Wi-Fi under Linux. Usually, Wi-Fi module can support two working modes, namely STA mode and AP mode. Some devices also support STA and AP mode working at the same time. The STA mode allows the device to connect to external Wi-Fi hotspots, and the AP mode turns the device into a Wi-Fi hotspot for other devices to connect to.

The L297B Wi-Fi and Bluetooth two-in-one module on the development board currently does not support STA and AP working at the same time. The corresponding drivers of L297B Wi-Fi module are moal and mlan:

```
root@myd-am62x:~# lsmod
Module                Size  Used by    Tainted: G
...
moal                  753664  0
```

```
mlan          528384  1  moal
```

The process of driver loading will load the Wi-Fi firmware located in /lib/firmware/nxp inside the module. After the Wi-Fi module driver is loaded successfully, the network node mlan0 of the Wi-Fi device is generated as follows:

```
root@myd-am62x:~# ifconfig mlan0
mlan0      Link encap:Ethernet  HWaddr 5C:C5:63:7D:3B:8C
           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 (0.0B) TX bytes:0 (0.0B)
```

1) Manually connect to WiFi hotspot in STA mode

Let's try manually connecting to the nearby Wi-Fi hotspot "MYIR", which is a Wi-Fi hotspot using WPA2 encryption with the password myir@1601.

- **Make sure the mlan0 network device is active**

```
root@myd-am62x:~# ifconfig mlan0 up
```

- **Scan for nearby WiFi hotspots**

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

```
root@myd-am62x:~# iw dev mlan0 scan | grep SSID
SSID: 360WiFi-4FF7C3
SSID: HP-Print-3C-LaserJet Pro MFP
SSID: MYIR
SSID: HUAWEI_B316_CA1C
SSID: MYIR
```

- **wpa_passphrase Set wifi name and password**

```
root@myd-am62x:~# wpa_passphrase MYIR myir@1601 >> /etc/wpa_supplicant.conf
root@myd-am62x:~# cat /etc/wpa_supplicant.conf
ctrl_interface=/var/run/wpa_supplicant
```

```
ctrl_interface_group=0
update_config=1

network={
    key_mgmt=NONE
}
network={
    ssid="MYIR"
    #psk="myir@1601"
    psk=6a85bda077eebd8473c9b585b74c823199c6cbb66a4f1d9e40c9b94
e9605aa0
}
```

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

- **Turn off the wpa_supplicant process**

Before using wpa_supplicant to connect and configure WIFI, you need to shut down the wpa_supplicant process.

```
root@myd-am62x:~# killall wpa_supplicant
```

- **Initialize wpa_supplicant**

wpa_supplicant is a tool to connect and configure WIFI, its main job is to interact with the driver through the socket and report data to the user layer, and the user layer can also send commands to wpa_supplicant through the socket to mobilize the driver to operate the WiFi chip. It usually runs in the background as follows:

```
root@myd-am62x:~# wpa_supplicant -B -i wlan0 -c /etc/wpa_supplicant.conf
```

- -B: Run daemons in the background
- -c: Path to config information
- -i: wifi interface to listen to

- **Get the WiFi IP address**

```
root@myd-am62x:~# udhcpc -i wlan0
udhcpc: started, v1.31.1
```

```
udhcpc: sending discover
udhcpc: sending select for 192.168.40.107
udhcpc: lease of 192.168.40.107 obtained, lease time 7200
/etc/udhcpc.d/50default: Adding DNS 223.5.5.5
/etc/udhcpc.d/50default: Adding DNS 201.104.111.114

root@myd-am62x:~# ping www.baidu.com
PING www.baidu.com (112.80.248.75) 56(84) bytes of data.
64 bytes from 112.80.248.75 (112.80.248.75): icmp_seq=1 ttl=56 time=28.8 ms
64 bytes from 112.80.248.75 (112.80.248.75): icmp_seq=2 ttl=56 time=37.6 ms
64 bytes from 112.80.248.75 (112.80.248.75): icmp_seq=3 ttl=56 time=38.9 ms
```

- **Configure Wi-Fi STA mode using a shell script**

We will organize the process of manually configuring Wi-Fi into a script `ifup_wifi_sta` for user reference. Users only need to prepare the `/etc/wpa_supplicant.conf` configuration file, and then execute this script to configure the STA mode, the script is as follows:

```
#!/usr/bin/env sh

SSID=
PASSWD=
WLAN=mlan0
WPA_FILE=/etc/wpa_supplicant/wpa_supplicant-${WLAN}.conf
DRIVER_NAME=nl80211

usage(){
    echo "Usage: ./ifup_wifi_sta [-ssid wifi_sta_name] [-passwd wifi_sta_passwd]
    [-driver nl80211 or wext]"
}

clean_stage(){
    killall udhcpc
```



```

killall wpa_supplicant

killall hostapd
killall udhcpd
sleep 1
}

enable_wifi(){
    T_HCI="phy0"
    RFKILL_SYS_PATH="/sys/class/rfkill/"
    dir=`ls ${RFKILL_SYS_PATH}`
    for i in ${dir}
    do
        if [ ${T_HCI} == `cat ${RFKILL_SYS_PATH}${i}/name` ]; then
            echo 0 > ${RFKILL_SYS_PATH}${i}/state
            echo "find ${T_HCI} enable it"
            sleep 1
            echo 1 > ${RFKILL_SYS_PATH}${i}/state
        fi
    done
}

parse_input_info(){
    while [ $# -gt 0 ]; do
        case $1 in
            -ssid)
                SSID="$2"
                shift
                ;;
            -passwd)
                PASSWD="$2"
                if [ ${#PASSWD} -lt 8 ]; then

```



```

        echo "passwd should be 8... 64"
        exit

    fi
    shift
    ;;
    -driver)
        DRIVER_NAME="$2"
        shift
        ;;
    -h)
        usage
        exit
        ;;
esac
shift $(( $# > 0 ? 1 - 0 ))
done

echo "SSID:${SSID} PASSWD:${PASSWD} DRIVER:${DRIVER_NAME}"
}
connect_wifi(){
    if [ -n "${SSID}" ]; then
        if [ ! -d ${WPA_FILE%/*} ]; then
            mkdir -p ${WPA_FILE%/*}
        fi
        echo "ctrl_interface=/var/run/wpa_supplicant" > ${WPA_FILE}
        echo "ctrl_interface_group=0" >> ${WPA_FILE}
        echo "update_config=1" >> ${WPA_FILE}
        echo "" >> ${WPA_FILE}

        wpa_passphrase ${SSID} ${PASSWD} >> ${WPA_FILE}

    fi
}

```

```

    wpa_supplicant -B -i ${WLAN} -c ${WPA_FILE} -D ${DRIVER_NAME} >/dev/
null 2>&1
}

obtain_dns(){
    time=10
    while [ $time -gt 0 ]; do
        state=`wpa_cli -i${WLAN} -p/var/run/wpa_supplicant status | grep wpa_
state | awk -F[=] '{print $2}'`
        if [ "${state}" = "COMPLETED" ]; then
            udhcpc -i ${WLAN}
            exit
        fi
        let time-=1
        sleep 1
    done
    echo "connect wifi error"
}

parse_input_info $@
clean_stage
enable_wifi
connect_wifi
obtain_dns

```

Using the above script will generate the /etc/wpa_supplicant.conf configuration file based on the ssid and passwd parameters entered, connect to the specified WiFi hotspot, and configure the IP address information of wlan0 via DHCP, for example:

```
root@myd-am62x:~# /usr/bin/ifup_wifi_sta -ssid MYIR -passwd myir@1601
```

After executing the above command, wlan0 will connect to the external MYIR hotspot in STA mode.

2) AP mode manually configures the hotspot

Hostapd is a wireless access point with encryption program, is on the Linux operating system component of the wireless access point is a more convenient tool to support the IEEE 802.11 protocol and IEEE 802.1X/WPA/connected/EAP/RADIUS of encryption. As a WiFi hotspot, the board needs to assign IP, routing and other network parameters for each terminal accessing the hotspot (such as mobile phones). For example, SSID is MYIR_TEST, PASSWD is myir2020 wireless wifi hotspot. Let's configure it manually first:

- **Configure the IP address of wlan0**

When using AP mode, you need to configure a static IP address for wlan0 to be activated. Here you configure a default IP address: 192.168.10.1.

```
root@myd-am62x:~# ifconfig wlan0 192.168.10.1up
```

In the previous section we explained that the current Wi-Fi module does not support STA and AP modes to work at the same time, so we need to clear the STA mode configuration here, as follows:

```
root@myd-am62x:~# killall udhcpc
root@myd-am62x:~# killall udhcpd
root@myd-am62x:~# killall wpa_supplicant
root@myd-am62x:~# killall hostapd
```

- **Run the DHCP service using the wlan0 device**

wlan0 works in AP mode, when other devices connect to this AP hotspot, it needs to dynamically allocate IP address for other devices through wlan0, so it needs to use wlan0 to run DHCP service program udhcpd. The corresponding configuration file for udhcpd is /etc/udhcpd.conf, and modify the contents as follows:

```
# the start and end of the IP lease block
start 192.168.10.10
end 192.168.10.254
# the interface that udhcpd will use
```

```
interface        uap0
opt dns 8.8.8.8
option subnet 255.255.255.0
opt router 192.168.10.1
option domain local
option lease 864000
```

When the AP mode is configured and other devices connect to the hotspot, they will obtain the IP address in the above address pool through mlan0, the address range is 192.168.10.10 to 192.168.10.254, the subnet mask is 255.255.255.0, the default gateway is 192.168.10.1, and the IP address is 192.168.10.1. DNS is 8.8.8.8.

Once the udhcpd configuration file is ready, run the following command to start the udhcpd service.

```
root@myd-am62x:~# udhcpd /etc/udhcpd.conf
```

- **Run the hostapd service using mlan0**

The most critical step in configuring AP mode is, of course, starting the hostapd service. Before starting the service, you need to configure the ssid, password, encryption algorithm, driver type, working mode, etc. of AP mode through /etc/hostapd.conf. For a complete parameter configuration description, please refer to: <http://w1.fi/cgit/hostap/plain/hostapd/hostapd.conf>, the following is in view of the current hardware/etc/hostapd. Conf configuration, content is as follows:

```
# File: /etc/hostapd.conf
interface=uap0
driver=nl80211
# mode Wi-Fi (a = IEEE 802.11a, b = IEEE 802.11b, g = IEEE 802.11g)
hw_mode=g
ssid=MYIR_TEST
channel=7
wmm_enabled=0
macaddr_acl=0
# Wi-Fi closed, need an authentication
```

```
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=myir2020
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

After the configuration file is ready, execute the following command to start the hostapd service, and you can normally use the above configured hotspot.

```
root@myd-am62x:~# hostapd -B /etc/hostapd.conf
```

If the Ethernet card eth0 is already connected to the Internet, then the device connected to MYIR_TEST can also be connected to the Internet by using the following configuration for IP forwarding.

```
root@myd-am62x:~# echo "1" > /proc/sys/net/ipv4/ip_forward
root@myd-am62x:~# iptables -t nat-A POSTROUTING -s 192.168.10.1/24-o eth0-j MASQUERADE
```

● **Configure Wi-Fi AP mode using a shell script**

We organize the process of manual Wi-Fi configuration into a script ifup_wifi_ap for user reference. Users only need to prepare two configuration files /etc/dhcp/dhcpd.conf and /etc/hostapd.conf, and then execute this script to configure the AP mode. The script is as follows:

```
#!/usr/bin/env sh
ETH=eth0
WLAN=uap0
WLAN_IP = 192.168.10.1
DHCP_FILE=/etc/myir_udhcpd.conf
HOSTAPD_FILE=/etc/myir_hostapd.conf
clean_stage(){
    killall udhcpd
    killall wpa_supplicant

    killall hostapd
```

```

killall udhcpd
sleep 1
}
enable_wifi(){
    T_HCI="phy0"
    RFKILL_SYS_PATH="/sys/class/rfkill/"
    dir=`ls ${RFKILL_SYS_PATH}`
    for i in ${dir}
    do
        if [ ${T_HCI} == `cat ${RFKILL_SYS_PATH}${i}/name` ]; then
            echo 0 > ${RFKILL_SYS_PATH}${i}/state
            echo "find ${T_HCI} enable it"
            sleep 1
            echo 1 > ${RFKILL_SYS_PATH}${i}/state
        fi
    done
}
enable_sta_mode(){

    ifconfig ${WLAN} up ${WLAN_IP}

    echo "1" > /proc/sys/net/ipv4/ip_forward
    iptables -t nat -A POSTROUTING -s ${WLAN_IP}/24 -o ${ETH} -j MASQUE
RADE

    sleep 1

    udhcpd ${DHCP_FILE}
    hostapd -B ${HOSTAPD_FILE}
}
clean_stage
enable_wifi
enable_sta_mode

```

To configure the AP mode, run `/usr/bin/ifup_wifi_ap` in the Linux system console. After successful execution, the user can use other devices to connect to the MYIR_TEST hotspot for testing

4.1.3. Bluetooth

BlueZ (<http://www.bluez.org/>) is usually used to configure and manage Bluetooth devices on Linux platform. BlueZ is a set of relatively complete Bluetooth configuration and management tool set and protocol stack, the following use of these tools to configure and use Bluetooth.

Here, the Bluetooth function in the L297b 2-in-1 module is tested, and its corresponding drivers are `hci_uart` module and `btbcm` module. Firstly, `hciattach` is used to establish the data connection channel between the serial port and the Bluetooth protocol layer:

```
root@myd-am62x:~# hciattach /dev/ttyS3 any 115200 flow
[4074.747618] Bluetooth: HCI UART driver ver 2.3
[4074.752231] Bluetooth: HCI UART protocol H4 registered
[4074.758096] Bluetooth: HCI UART protocol LL registered
[4074.764054] Bluetooth: HCI UART protocol Broadcom registered
[4074.770479] Bluetooth: HCI UART protocol QCA registered
[4074.776194] Bluetooth: HCI UART protocol Marvell registered
Device setup complete
```

The `lsmod` command shows the following result of the driver:

```
root@myd-am62x:~# lsmod | grep hci
hci_uart          77824  1
btqca             20480  1 hci_uart
btbcm             24576  1 hci_uart
bluetooth         507904  24 btqca,hci_uart,btbcm
```

The `hci0` device node is generated after the driver is loaded successfully. Here's how to configure and connect a nearby Bluetooth device using the BlueZ toolset:

1) Activate the Bluetooth device `hci0`

```
root@myd-am62x:~# hciconfig hci0 up
```

If the following prompt appears:

```
root@myd-am62x:~# hciconfig hci0 up
Can't init device hci0: Operation not possible due to RF-kill (132)
```

It means that the Bluetooth switch is not open, you can check the current Bluetooth switch status through rfkill, and do the corresponding treatment.

- **Check the switch status of the wireless device**

```
root@myd-am62x:~# rfkill list
0: phy0: wlan
    Soft blocked: no
    Hard blocked: no
1: hci0: bluetooth
    Soft blocked: yes
    Hard blocked: no
```

The above results show that the rfkill ID of the Bluetooth device is 0, and it is currently in the state of soft blocked, so it cannot wake up the activation and subsequent operation.

- **Turn on Bluetooth switch**

```
root@myd-am62x:~# echo 1 > /sys/class/rfkill/rfkill0/state
```

After changing the state of Bluetooth with the above command, it is no problem to run hciconfig hci0 up again.

2) Scan for nearby Bluetooth devices that are available

```
root@myd-am62x:~# hcitool scan
Scanning ...
    54:14:F3:CA:70:37      BESTE
    C0:3C:59:46:38:49      USER-WU
    84:5C:F3:24:9C:89      DESKTOP-LJRC636
```

3) Manage your Bluetooth connection via the bluetoothctl command

bluetoothctl is a set of Bluetooth management tools provided by BlueZ, enabling Bluetooth controller power, proxy, scan, pair and connect through this command.

Let's use bluetoothctl for specific test Bluetooth scanning, pairing and connection, the steps are as follows:

- **Terminal input bluetoothctl to enter the Bluetooth control interface**

```
root@myd-am62x:~# bluetoothctl
Agent registered
[CHG] Controller 5C:C5:63:01:CC:0E Pairable: yes
[bluetooth]#
```

- **Enable Bluetooth to control power**

```
[bluetooth]# power on
Changing power on succeeded
```

- **Enable Bluetooth proxy**

Manage and see if agent management was successful, this is enabled by default.

```
[bluetooth]# agent on
Agent is already registered
[bluetooth]# default-agent
Default agent request successful
```

- **Scan for nearby Bluetooth devices that can connect**

```
[bluetooth]# scan on
Discovery started
[CHG] Controller 5C:C5:63:01:CC:0E Discovering: yes
[NEW] Device F0:A6:54:99:AA:F6 QDXCXY
[NEW] Device 54:14:F3:CA:70:37 BESTE
[bluetooth]# scan off
[CHG] Device B0:FC:36:3B:CF:0E TxPower is nil
[CHG] Device B0:FC:36:3B:CF:0E RSSI is nil
[CHG] Device 54:14:F3:CA:70:37 RSSI is nil
```

```
[CHG] Controller C0:84:7D:B6:34:69 Discovering: no
Discovery stopped
```

Follow up tests as follows through the device scanned above **54:14:F3:CA:70:37** **BESTE**.

- **Do device pairing**

```
[bluetooth]# pair 54:14:F3:CA:70:37
Attempting to pair with 54:14:F3:CA:70:37
[CHG] Device 54:14:F3:CA:70:37 Connected: yes
Request confirmation
[agent] Confirm passkey 711813 (yes/no): yes
[CHG] Device 54:14:F3:CA:70:37 Bonded: yes
[CHG] Device 54:14:F3:CA:70:37 Modalias: bluetooth:v0006p0001d0A00
[CHG] Device 54:14:F3:CA:70:37 UUIDs: 00001000-0000-1000-8000-00805f9b34fb
[CHG] Device 54:14:F3:CA:70:37 UUIDs: 0000110a-0000-1000-8000-00805f9b34fb
[CHG] Device 54:14:F3:CA:70:37 UUIDs: 0000110b-0000-1000-8000-00805f9b34fb
[CHG] Device 54:14:F3:CA:70:37 ServicesResolved: yes
[CHG] Device 54:14:F3:CA:70:37 Paired: yes
Pairing successful
[CHG] Device 54:14:F3:CA:70:37 ServicesResolved: no
[CHG] Device 54:14:F3:CA:70:37 Connected: no
```

At this point, it means that it has been successfully paired with the Xiaomi mobile phone Bluetooth

- **Connecting the device**

```
# [11501.839856] input millet, millet (AVRCP) support the as/devices/virtual/input/input2
[CHG] Device 54:14:F3:CA:70:37 ServicesResolved: yes
[DEL] Device B0:FC:36:3B:CF:0E MYIR-BT
[Mi]#
```

At this point, you will be able to see the phone with Bluetooth on it as well as the connection.

4.1.4. 4G/5G

LINUX devices can also be connected to an external 4G or 5G module for dial-up Internet access. The 4G module uses EM05-CE, and the 5G module includes RM500Q and RG801H.

MYD-YM62X provides an M.2 Key B (J21) interface to connect to the external 5G module. The 5G module model is RM500Q, which can be directly used after being inserted into the board, and hot plugging is not supported.

1) Check the VID and PID

Install the RM500Q module to connect to the development board through the M.2 interface, and then start the development board. Use lsusb to view the RM500Q module information.

```
root@myd-am62x:~# lsusb
Bus 004 Device 002: ID 2c7c:0800
...
```

➤ 2 c7c: 0800: RM500Q VID and PID information.

2) View the kernel identification module

If kernel has increased the VID and PID configuration for this module, then a node for /dev/ttyUSB* is generated:

```
root@myd-am62x:~# ls -l /dev/ttyUSB*
crw-rw---- 1 root dialout 188, 0 Dec  9 10:18 /dev/ttyUSB0
crw-rw---- 1 root dialout 188, 1 Dec  9 10:18 /dev/ttyUSB1
crw-rw---- 1 root dialout 188, 2 Dec  9 10:18 /dev/ttyUSB2
crw-rw---- 1 root dialout 188, 3 Dec  9 10:18 /dev/ttyUSB3
```

3) Use the AT command for a preliminary test

Use the AT command to easily check the signal strength, whether the SIM card is inserted, whether the SIM card is currently searching for a carrier, and also use the AT command to make a phone call to test the current card function. Here AT communication also need to know which device is the communi

cation port, here need to query the module file, EC20 uses ttyUSB2 AT communication. Here use microcom for example, you can also use minicom. For example: microcom/dev/ttyUSB2 Enter mode ctrl+x to exit.

- **Query signal quality**

```
root@myd-am62x:~# microcom /dev/ttyUSB2
at+csq
+CSQ: 22,99
OK
```

➤ 22, 99:22 is the signal quality, with a higher number indicating a stronger signal.

- **Check if it can be operated**

```
at+cpin?
+CPIN: READY
OK
```

➤ +CPIN:READY :READY stands for ready.

Since the MYD-YM62X has two SIM card slots, if you find that the above test step feedback fails, you can try to switch the SIM card slot selection of the module:

```
AT+QUIMSLLOT=1/2 // Switch to SIM1/2
AT+QUIMSLLOT? // View the currently used SIM card
2
```

- **Check the carrier**

```
at+cops?
+COPS: 0,0,"CHN-UNICOM",2
OK
```

➤ "CHN-UNICOM",2: CHN-UNICOM stands for China Unicom, 100 stands for 2G, 3G, 4G, or 5G according to the module manual.

If the above 3 steps can be normal, you can dial up the Internet, here also introduces how to call and send text messages, further verification.

- **Making a call**

```
ATD134xxxx5673;
OK
```

- **Texting**

```
at+cmgf=1
OK
at+cscs="GSM"
OK
at+cmgs="134xxxx5673"
> hello[63.568240] audit: type=1006 audit(1607509801.160:3): pid=741 uid=0
old-auid=4294967295 auid=0 tty=(none) old-ses=4294967295 ses=2 res=1

+CMGS: 76
OK
```

- At+cmgf=1: Set text message mode
- At+cscs="GSM" : Set TE to use GSM characters
- At+cmgs: CTRL+Z send after "phone" writes message, ESC quit send

4) ppp dialing test

The pppd dialing command that comes with the development board is used here:

```
root@myd-am62x:~# pppd call quectel-dial
root@myd-am62x:~# [33.774572] [dhd] CFG80211-ERROR) wl_cfg80211_netdev
_notifier_call: wdev null. Do nothing
[33.783789] [dhd] CFG80211-ERROR) wl_cfg80211_netdev_notifier_call: wdev nu
ll.Do nothing
[37.259047] [dhd] CFG80211-ERROR) wl_cfg80211_netdev_notifier_call: wdev nu
ll. Do nothing
[37.267814] [dhd] CFG80211-ERROR) wl_cfg80211_netdev_notifier_call: wdev nu
ll.Do nothing
```

This call will take a while. The call log is hidden, so you can view the log:

```
root@myd-am62x:~# cat /var/log/quectel-dial.log
pppd options in effect:
debug          # (from /etc/ppp/peers/quectel-dial)
slightly
noauth         # (from /etc/ppp/peers/quectel-dial)
user card      # (from /etc/ppp/peers/quectel-dial)
password ?????? # (from /etc/ppp/peers/quectel-dial)
slightly
not replacing default route to eth0 [192.168.40.1]
local IP address 10.241.79.143
remote IP address 10.64.64.64
primary DNS address 218.104.111.114
secondary DNS address 218.104.111.122
Script /etc/ppp/ip-up started (pid 749)
Script /etc/ppp/ip-up finished (pid 749), status = 0x0
```

You should see that you are connected normally and can get the IP

Enter the following command to view "ppp0" and find that IP has been normally obtained, and can ping Baidu normally:

```
root@myd-am62x:~# ifconfig ppp0
ppp0      Link encap:Point-to-Point Protocol
          inet addr: 10.241.79.143P-t-P :10.64.64.64 Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:
1
          RX packets:7 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:130 (130.0B) TX bytes:414 (414.0B)

root@MYD-JYM62X:~# ping www.baidu.com
PING www.baidu.com (14.215.177.39) 56(84) bytes of data.
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=1 ttl=54 time=19.6 ms
```

64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq= 2ttl =54 time=19.4 ms

64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=3 ttl=54 time=19.5 ms

64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=4 ttl=54 time=19.6 ms

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

5) Use qmi_wwan dialing

Use quectel-CM dialing, using method quectel-CM&, where "-s ctnet" telecom; "-s cmnet" mobile; "-s 3gnet" connectivity:

```
root@myd-am62x:~# quectel-CM &
854
root@myd-am62x:~# [09-07_09:59:16:978] Quectel_QConnectManager_Linux_V1.6.0.16
[09-07_09:59:16:980] Find /sys/bus/usb/devices/2-1.3 idVendor=0x2c7c idProduct=0x125, bus=0x002, dev=0x003
[09-07_09:59:16:981] Auto find qmichannel = /dev/cdc-wdm0
[09-07_09:59:16:981] Auto find usbnet_adapter = wwan0
[09-07_09:59:16:981] netcard driver = qmi_wwan_q, driver version = 6.1.46-g6b301669c273-dirty
[09-07_09:59:16:981] ioctl(0x89f3, qmap_settings) failed: No such file or directory, rc=-1
[09-07_09:59:16:982] Modem works in QMI mode
[09-07_09:59:17:000] cdc_wdm_fd = 7
[09-07_09:59:17:106] Get clientWDS = 7
[09-07_09:59:17:137] Get clientDMS = 1
[09-07_09:59:17:170] Get clientNAS = 2
[09-07_09:59:17:201] Get clientUIM = 1
[09-07_09:59:17:234] Get clientWDA = 1
[09-07_09:59:17:266] requestBaseBandVersion EM05CEFCR06A04M1G_ND
[09-07_09:59:17:395] requestGetSIMStatus SIMStatus: SIM_READY
[09-07_09:59:17:427] requestGetProfile[1] 3gnet///0
[09-07_09:59:17:458] requestRegistrationState2 MCC: 460, MNC: 11, PS: Attached, DataCap: LTE
```



```
[09-07_09:59:17:491] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED
[09-07_09:59:17:491] ifconfig wwan0 0.0.0.0
[09-07_09:59:17:508] ifconfig wwan0 down
[09-07_09:59:17:586] requestSetupDataCall WdsConnectionIPv4Handle: 0x8719e440
[09-07_09:59:17:715] ifconfig wwan0 up
[36.505425] systemd-journald[177]: Time jumped backwards, rotating.
[09-07_09:59:17:224] busybox udhcpc -f -n -q -t 5 -i wwan0
udhcpc: started, v1.35.0
udhcpc: broadcasting discover
udhcpc: broadcasting select for 10.73.143.43, server 10.73.143.44
udhcpc: lease of 10.73.143.43 obtained from 10.73.143.44, lease time 7200
[09-07_09:59:17:515] /etc/udhcpc.d/50default: Adding DNS 202.103.44.150
[09-07_09:59:17:515] /etc/udhcpc.d/50default: Adding DNS 202.103.24.68
```

ifconfig will see that the IP is retrieved as expected:

```
root@myd-am62x:~# ifconfig wwan0
wwan0      Link encap:Ethernet  HWaddr 0e:84:fc:87:fd:cb
            inet addr:10.42.46.46 Bcast:10.42.46.47 Mask:255.255.255.252
            UP BROADCAST RUNNING NOARP 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 (0.0B) TX bytes:0 (0.0B)
```

Ping external network:

```
root@myd-am62x:~# ping www.baidu.com
PING www.baidu.com (14.215.177.39) 56(84) bytes of data.
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=1 ttl=54 time=20.1 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq= 2ttl =54 time=19.7 m
s
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=3 ttl=54 time=19.5 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=4 ttl=54 time=19.4 ms
```

4.2. Network applications

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

4.2.1. PING

PING is mainly used to test network connectivity, but it can also test network latency and packet loss rate. Once you have configured your Ethernet connection as described in 4.1.1, you can use PING for a simple test of your network connection.

1) Wiring and message output

Connect the device to a switch or router through a network cable, such as the eth0 network port of the development board.

```
root@myd-am62x:~# udhcpc -i eth0
udhcpc: started, v1.33.0
udhcpc: sending discover
udhcpc: sending select for 192.168.30.131
udhcpc: lease of 192.168.30.131 obtained, lease time 59353
deleting routers
adding dns 192.168.30.1
root@myd-am62x:~#
```

2) Test external web addresses

```
root@myd-am62x:~# ping www.baidu.com -I eth0
PING www.baidu.com (112.80.248.76): 56 data bytes
64 bytes from 112.80.248.76: seq=0 ttl=55 time=13.029 ms
64 bytes from 112.80.248.76: seq=1 ttl=55 time=12.909 ms
64 bytes from 112.80.248.76: seq=2 ttl=55 time=13.239 ms
64 bytes from 112.80.248.76: seq=3 ttl=55 time=12.884 ms
64 bytes from 112.80.248.76: seq=4 ttl=55 time=12.812 ms
^C
--- www.baidu.com ping statistics ---
```

5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 12.812/12.974/13.239 ms

Note: ping public network needs to make sure DNS is working properly.

The above result shows that the IP address of www.baidu.com after domain name resolution is 112.80.248.76. icmp_seq represents the number of icmp packets. time stands for the latency of the response, the shorter the better. In addition to testing Ethernet, the ping command can also be used to test Wi-Fi.

4.2.2. SSH

SSH stands for Secure Shell and was developed by the IETF's Network Working Group. SSH is a security protocol based on the application layer, which is more reliable and designed to provide security for remote login sessions and other network services. Usually under the Linux platform, dropbear or OpenSSH is used to realize the SSH server and client. Let's test the SSH client and server over an Ethernet connection. The current factory default includes the client and server provided by openssh (<http://www.openssh.com/>).

First, configure the connection between the Ethernet interface and the SSH server according to Section 4.1.1. The configured Ethernet card address is as follows:

```
root@myd-am62x:~# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 1C:63:49:1A:6A:1F
          inet addr:192.168.40.216 Bcast:192.168.30.255 Mask:255.255.255.0
          inet6 addr: fe80::1e63:49ff:fe1a:6a1f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:13590 errors:0 dropped:2181 overruns:0 frame:0
          TX packets:364 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1630491 (1.5 MiB) TX bytes:27886 (27.2 KiB)
```

The IP address of the SSH server is 192.168.40.21. After testing that the connection between the device and the SSH server is working with the ping command, the following tests can be performed.

- **SSH client test**

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

```
root@myd-am62x:~# ssh beste@192.168.40.21

Host '192.168.1.13' is not in the trusted hosts file.
(ecdsa-sha2-nistp256 fingerprint sha1!! 99:7b:56:89:e4:26:b5:2a:9a:80:27:85:a3:92:14:02:f0:ff:13:c1)
Do you want to continue connecting? (y/n) y
beste@192.168.40.21's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-79-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage
New release '22.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2025.
*** System restart required ***
Last login: Wed Sep 20 10:52:24 2023 from 192.168.40.41
beste@system1:~$
```

After the successful login, the console console on the SSH server is automatically entered, and the user can perform the control within the my-linux user rights on the remote server from the client side. If you need to log out, you can directly execute the "exit" command in the current console.

● SSH server-side testing

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

Since the SSH server is also enabled by default on the device side, we can also use the SSH command to log in to the current device on other devices that have ssh clients. The command and results are as follows:



```
beste@system1:~$ssh root@192.168.40.216
```

In the example above, we are logging into the device remotely as root and accessing the console, which allows us to perform root user privileges on the device. If you need to log out, you can simply execute the "exit" command directly in the console.

OpenSSH is the main connection tool for logging in remotely using the SSH protocol. It encrypts all traffic to eliminate eavesdropping, connection hijacking, and other attacks. In addition, OpenSSH offers a range of large secure tunnel capabilities, multiple authentication methods, and sophisticated and flexible configuration options. Users can modify the configuration files `ssh_config` and `sshd_config` located in the `/etc/ssh/directory` according to their needs. For example, if you want the SSH server to allow root to log in remotely without a password, you can modify `/etc/ssh/sshd_config` on the SSH server machine by adding the following two lines:

```
PermitRootLogin yes
PermitEmptyPasswords yes
```

The above configuration has a relatively large security risk, generally used for debug phase remote deployment. In actual production, it is usually turned off for security reasons.

4.2.3. SCP

SCP is the abbreviation of Secure Copy, it is the linux system based on SSH protocol secure remote file copy command, in the system debugging phase it is very useful.

In 4.2.2, we have already introduced an example of using SSH protocol and SSH client and server for remote login, here is an example of using SCP command for remote file copy:

1) Copy files from remote to local

```
PC $scp test.txt root@192.168.40.216:~
```

```
The authenticity of host '192.168.40.216 (192.168.40.216)' can't be established.
ECDSA key fingerprint is SHA256:C6fXFGctxoRu2eBuCPe04fEcfsRzI82WJ1Rbbqj
kp4.
```

```
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.40.216' (ECDSA) to the list of known ho
sts.
test.txt 100% 2303KB 1.1MB/s 00:02
```

This file can be seen by going to the development board home directory, as follows:

```
root@myd-am62x:~# ls
test.txt
```

2) Copy the file from local to remote

```
# scp test.txt beste@192.168.40.216:~/
The authenticity of host '192.168.40.216 (192.168.40.216)' can't be established.
ECDSA key fingerprint is SHA256:KCTJPoHzafew1fRJmTx2hV4BNymgaZ1WDg2o
vdtqtCw.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.40.216' (ECDSA) to the list of known ho
sts.
myir@192.168.40.216's password:
test.txt
```

After successful verification, the file is copied from the device to the \$HOME directory of the specified account on the server.

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

4.2.4. TFTP

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

The default image file includes the tftp client program provided by busybox. The command syntax is as follows:

```
root@myd-am62x:~# tftp --help
BusyBox v1.29.3 () multi-call binary.
```

Usage: tftp [OPTIONS] HOST [PORT]

The detailed parameter description is as follows:

- -g: Get the file
- -p: Upload the file
- -l: Local file
- -r: Remote file
- HOST: Remote host IP address
- [PORT]: Ignored, defaults to 69

TFTP server can choose Linux platform TFTP - hpa, also can choose Windows platform tftpd32/64 (http://tftpd32.jounin.net/tftpd32_download.html). The following ubuntu platform as an example to explain the configuration of tftp server.

1) Install the TFTP server

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

2) Configure TFTP service

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

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

Modify or add the following fields:

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

3) Restart the TFTP service

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

After configuring the tftp server, place a test file test.txt in the <WORKDIR>/tftpboot/ directory configured above, and you can use the tftp client to download and upload files on the target device.

```
root@myd-am62x:~# tftp-g-r test.txt -l test.txt 192.168.40.21
```

The above command will download test.txt from the tftp server's <WORKDIR>/tftpboot directory to the current directory of the target device.

```
root@myd-am62x:~# tftp-p-l Ym62x-test -r ym62x-test.txt 192.168.0.2
```

The above command will upload the Ym62x-test file from the current directory on the target device to the <WORKDIR>/tftpboot directory configured earlier on the tftp server, and rename it to ym62x-test.txt.

4.2.5. DHCP

DHCP (Dynamic Host Configuration Protocol) is a network protocol for local area networks. It refers to a range of IP addresses controlled by the server. When the client logs in to the server, it can automatically obtain the IP addresses and subnet mask assigned by the server.

DHCP also has both server and client roles, and in 4.1.1 we tested the use of DHCP client mode to automatically obtain IP addresses; When configuring WiFi AP mode in 4.1.2, we also tested DHCP server mode for assigning IP addresses to connected WiFi devices. Here is to introduce the use of udhcpc command to manually obtain the IP address method, convenient for users to use when debugging the network.

Connect the development board and router with a network cable, use the command to manually assign an IP address to the eth0 network card, and observe the process of dhcp obtaining an ip address.

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

```
root@myd-am62x:~# udhcpc -i eth0
udhcpc: started, v1.31.1
udhcpc: sending discover
udhcpc: sending select for 192.168.30.199
[3038.408887] IPv4: martian source 255.255.255.255 from 192.168.8.1, on dev eth0
```

```
[3038.414720] ll header: 00000000: ff ff ff ff ff ff 20 da 22 fc ca 1b 08 00
udhcpc: lease of 192.168.30.199 obtained, lease time 3600
/etc/udhcpc.d/50default: Adding DNS 223.5.5.5
/etc/udhcpc.d/50default: Adding DNS 201.104.111.114
```

You can then configure the IP address for eth0, along with the gateway, sub net mask, DNS, and other information as follows:

```
root@myd-am62x:~# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 26:13:52:CE:15:40
          inet addr:192.168.30.199 Bcast:192.168.30.255 Mask:255.255.255.0
          inet6 addr: fe80::2413:52ff:fece:1540/64 Scope:Link
          inet6 addr: fd20:da22:fcca:1b00:2413:52ff:fece:1540/64 Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:20745 errors:0 dropped:752 overruns:0 frame:0
          TX packets:633 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1529726 (1.4 MiB) TX bytes:93845 (91.6 KiB)
          Interrupt:51 Base address:0x8000

root@myd-am62x:~# cat /etc/resolv.conf
nameserver 223.5.5.5
nameserver 201.104.111.114
```

4.2.6. IPTables

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

Devices using Linux systems usually use the iptables tool to configure firewalls. iptables handles various packets according to the methods defined by the packet filtering rules, such as accept, reject, and drop. Let's use iptables to test how we can intercept icmp packets and prevent other devices on the network.

ork from ping them. Specific commands to use see: <https://linux.die.net/man/8/iptables>

1) Configure the target device iptables

To use the iptables configuration on the target device to drop incoming icmp packets and not respond to ping probes from other hosts, the command is as follows:

```
root@myd-am62x:~# iptables -A INPUT -p icmp --icmp-type 8 -j DROP
root@myd-am62x:~# iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
-A INPUT -p icmp -m icmp --icmp-type 8 -j DROP
```

2) Test ping the target device

ping the target device on the development host with a deadline of 10 gives the following result:

```
PC$ping 192.168.40.216 -w 10
PING 192.168.40.216 (192.168.40.216) 56(84) bytes of data.

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

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

3) Delete the corresponding firewall rules

```
root@myd-am62x:~# iptables -F
root@myd-am62x:~# iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
```

4) Test ping the target device again

```
PC$ping 192.168.40.216 -w 5
```

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

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

After clearing the iptables rules, ping the target device again from the development host, and you're ready to ping. The preceding example is just a simple demonstration. In fact, iptables combined with various rules can be very powerful, so I won't cover it in detail here.

4.2.7. Ethtool

ethtool is a tool to view and modify the parameters of Ethernet devices, which is useful in the network debugging phase. Let's use this command to check the information of the Ethernet card and try to modify its parameters.

First, let's check the help information of the command through `ethtool -h`

```
root@myd-am62x:~# ethtool --help
ethtool version 4.5
Usage:
    ethtool DEVNAME Display standard information about device
    ethtool -s|--change DEVNAME      Change generic options
        [ speed %d ]
        [ duplex half|full ]
        [ port tp|aui|bnc|mii|fibre ]
        [ mdix auto|on|off ]
        [ autoneg on|off ]
        [ advertise %x ]
        [ phyad %d ]
```



```
[ xcvr internal|external ]
[ wol p|u|m|b|a|g|s|f|d... ]
[ sopass %x:%x:%x:%x:%x:%x ]
[ msglvl %d | msglvl type on|off ... ]
ethtool -a|--show-pause DEVNAME Show pause options
```

...

View the current equipment basic information of the Ethernet card

```
root@myd-am62x:~# ethtool eth0
```

Settings for eth0:

```
Supported ports: [ TP      MII ]
Supported link modes:  10baseT/Half 10baseT/Full
                      100baseT/Half 100baseT/Full
                      1000baseT/Full
Supported pause frame use: Symmetric
Supports auto-negotiation: Yes
Supported FEC modes: Not reported
Advertised link modes: 10baseT/Half 10baseT/Full
                      100baseT/Half 100baseT/Full
                      1000baseT/Full
Advertised pause frame use: Symmetric
Advertised auto-negotiation: Yes
Advertised FEC modes: Not reported
Speed: 1000Mb/s
Duplex: Full
Auto-negotiation: on
Port: Twisted Pair
PHYAD: 5
Transceiver: internal
MDI-X: Unknown
Supports Wake-on: d
Wake-on: d
Current message level: 0x000020f7 (8439)
drv probe link ifdown ifup rx_err tx_err hw
```

Link detected: yes

Through the ethtool command, you can see the current Ethernet card supports six connection modes: 10 megabit, 100 megabit and Gigabit half duplex and full duplex. The current connection state is negotiated gigabit, full duplex mode, using MII interface, PHY address is 6 and so on.

We can also use the ethtool tool to set the parameters of Ethernet, which plays a certain role in Ethernet debugging and diagnosis, for example, we force the Ethernet to be set to 100 megabit full duplex, and turn off self-negotiation, the command is as follows:

```
root@myd-am62x:~# ethtool -s eth0 speed 100 duplex full autoneg off
root@myd-am62x:~# [66620.951859] am65-cpsw-nuss 8000000.ethernet eth0:
Link is Down
[66625.041328] am65-cpsw-nuss 8000000.ethernet eth0: Link is Up - 100Mbps
/ Full-flow control off
```

More instructions about ethtool refer to: <http://man7.org/linux/man-pages/man8/ethtool.8.html>.

4.2.8. iPerf3

iPerf3 is a tool to actively measure the maximum achievable bandwidth on IP networks. It supports tuning various parameters such as test time, buffer size and protocol (TCP, UDP, SCTP under IPV4 and IPV6). iPerf3 can be divided into server-side mode or client-side mode according to its role. We can use it to test and view network bandwidth, TCP window value, retransmission probability, etc. in TCP mode, and also test packet loss rate, delay and jitter under specified UDP bandwidth.

We use an ubuntu 16.04 server with a gigabit NIC as the server of iperf3, and the tested device as the client to test the TCP and UDP performance of the device Ethernet card, respectively.

First, we install iperf3 on the server as follows:

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

Connect the server and the device directly through CAT6 network cable, and configure their respective IP addresses. For example, let's set the server ip to

192.168.0.2 and the device IP to 192.168.0.60 and use the ping command to test to make sure they are connected.

Note: Try not to connect routers or switches, so as not to affect the test results by intermediate devices.

1) Testing TCP performance

● Server-side (192.168.0.2)

iperf3 on the server uses the -s parameter to indicate working in server-side mode.

```
PC $ iperf3 -s -i 2
```

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

● The client (192.168.0.60)

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

- -c 192.168.0.2: works on the client side, connects to the server side 192.168.0.2
- -i 2: Test results are reported at a time interval of 2 seconds
- -t 10: The total test duration is 10 seconds

```
root@myd-am62x:~# iperf3-c 192.168.0.2-i 2-t 10
```

```
Connecting to host 192.168.0.2, port 5201
```

```
[5] local 192.168.0.60 port 38994 connected to 192.168.0.2 port 5201
```

```
[ ID] Interval          Transfer      Bitrate          Retr  Cwnd
```

```
[5] 0.00-2.00 sec 225 MBytes 940 Mbits/sec 0 529 KBytes [5] 2.00-4.00 sec 224 MBytes 940 Mbits/sec 0 529 KBytes [5] 4.00-6.00 sec 224 MBytes 941 Mbits/sec 0 560 KBytes [5] 6.00-8.00 sec 224 MBytes 940 Mbits/sec 0 560 KBytes [5] 8.00-10.00 sec 225 MBytes 940 Mbits/sec 0 560 KBytes - - - - -
```

```
-----  
[ ID] Interval          Transfer      Bitrate          Retr
```

```
[5] 0.00-10.00 sec 1.10 GBytes 940 Mbits/sec 0 sender
```

```
[5] 0.00-10.00 sec 1.09GBytes 939 Mbits/sec receiver
```

iperf Done.

The client ends the test after 10 seconds and shows the above test result, which indicates that the TCP bandwidth is around 940Mbps, no retransmissions, and the TCP window value is 560KBytes at the time of the test.

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

```
$ iperf3 -s -i 2
-----
Server listening on 5201
-----
Accepted connection from 192.168.0.60, port 38992
[5] local 192.168.0.2 port 5201 connected to 192.168.0.60 port 38994
[ ID] Interval          Transfer      Bandwidth
[5] 0.00-2.00 sec 218 MBytes 916 Mbits/sec
[5] 2.00-4.00 sec 224 MBytes 941 Mbits/sec
[5] 4.00-6.00 sec 224 MBytes 941 Mbits/sec
[5] 6.00-8.00 sec 224 MBytes 940 Mbits/sec
[5] 8.00-10.00 sec 224 MBytes 940 Mbits/sec
[5] 10.00-10.04 sec 5.02 MBytes 938 Mbits/sec
-----
[ ID] Interval          Transfer      Bandwidth      Retr
[5] 0.00-10.04 sec 1.10 GBytes 937 Mbits/sec 0 sender
[5] 0.00-10.04 sec 1.09 GBytes 936 Mbits/sec receiver
-----
Server listening on 5201
-----
```

2) UDP test performance

● Server-side (192.168.0.2)

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

```
PC $ iperf3 -s -i 2
```

```
Server listening on 5201
```

- **The client (192.168.0.60)**

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

- -u: Works in UDP mode
- -c 192.168.0.2: Works on client side, connects to server side 192.168.0.2
- -i 2: Test results are reported at a time interval of 2 seconds
- -t 10: The total test duration is 10 seconds
- -b 100M: Set UDP transmission bandwidth to 100Mbps.

```
root@myd-am62x:~# iperf3-u-c 192.168.0.2-i 2-t 10-b 100M
```

```
Connecting to host 192.168.0.2, port 5201
```

```
[5] local 192.168.0.60 port 42492 connected to 192.168.0.2 port 5201
```

[ID]	Interval	Transfer	Bitrate	Total Datagrams
-------	----------	----------	---------	-----------------

[5]	0.00-2.00 sec	23.8 MBytes	100 Mbites/sec	17263
-----	---------------	-------------	----------------	-------

[5]	2.00-4.00 sec	23.8 MBytes	100 Mbites/sec	17266
-----	---------------	-------------	----------------	-------

[5]	4.00-6.00 sec	23.8 MBytes	100 Mbites/sec	17266
-----	---------------	-------------	----------------	-------

[5]	sec	23.8 MBytes	100 Mbites/sec	17263
-----	-----	-------------	----------------	-------

[5]	8.00-10.00 sec	23.8 MBytes	100 Mbites/sec	17268
-----	----------------	-------------	----------------	-------

[ID]	Interval	Transfer	Bitrate	Jitter	Lost/Total Datagrams
-------	----------	----------	---------	--------	----------------------

[5]	0.00-10.00 sec	119 MBytes	100 Mbites/sec	0.000 ms	0/86326 (0%) sender
-----	----------------	------------	----------------	----------	---------------------

[5]	0.00-10.00 sec	119 MBytes	100 Mbites/sec	0.068 ms	0/86326 (0%) receiver
-----	----------------	------------	----------------	----------	-----------------------

```
iperf Done.
```

After 10 seconds, the client ends the test and displays the above test result, indicating that UDP did not lose packets at the specified bandwidth of 100 MBPS.

The server also displays the following test result and continues listening on port 5201 for the client to connect:

```
$ iperf3 -s -i 2
-----
Server listening on 5201
-----
Accepted connection from 192.168.30.206, port 38372
[5] local 192.168.30.2 port 5201 connected to 192.168.30.206 port 42492
[ ID] Interval      Transfer    Bandwidth      Jitter    Lost/Total Datagrams
[5] 0.00-2.00sec 23.4MBytes 98.0Mbits /sec 0.079ms 0/16918 (0%)
[5] 2.00-4.00sec 23.8MBytes 100 Mbits/sec 0.077ms 0/17264 (0%)
[5] 4.00-6.00sec 23.8MBytes 100 Mbits/sec 0.065ms 0/17268 (0%)
[5] 6.00-8.00sec 23.8 MBytes 100 Mbits/sec 0.084 ms 0/17265 (0%)
[5] 8.00-10.00 sec 23.8 MBytes 100 Mbits/sec 0.074 ms 0/17267 (0%)
[5] 10.00-10.04 sec 486 KBytes 100 Mbits/sec 0.068 ms 0/344 (0%) - - - - -
-----
[ ID] Interval      Transfer    Bandwidth      Jitter    Lost/Total Datagrams
[5] 0.00-10.04 sec 119 MBytes 99.6 Mbits/sec 0.068 ms 0/86326 (0%) -----
-----
Server listening on 5201
-----
```

Client - b parameters, continue to increase the specified UDP bandwidth, when began to appear lost package to the UDP bandwidth is the actual UDP bandwidth. If the specified bandwidth reaches the top 1Gbps of the Ethernet card and there is still no packet loss, the bandwidth returned by the iperf3 test is the actual UDP bandwidth. For example, the following example specifies a bandwidth of 1000Mbps, and there is still no packet loss, but the actual bandwidth is actually around 600Mbps.

```
root@myd-am62x:~# iperf3-u-c 192.168.0.2-i 2-t 10-b 1000M
Connecting to host 192.168.0.2, port 5201
[4] local 192.168.0.60 port 58904 connected to 192.168.0.2 port 5201
[ ID] Interval      Transfer    Bandwidth      Total Datagrams
[4] 0.00-2.00 sec 129 MBytes 539 Mbits/sec 16462
```

```
[4] 2.00-4.00 sec 143 MBytes 599 Mbits/sec 18293
[4] 4.00-6.00 sec 143 MBytes 599 Mbits/sec 18295
[4] 6.00-8.00 sec 143 MBytes 600 Mbits/sec 18300
[4] 8.00-10.00 sec 143 MBytes 599 Mbits/sec 18294
-----
[ ID] Interval      Transfer      Bandwidth      Jitter    Lost/Total Datagrams
[4] 0.00-10.00 sec 700 MBytes 587 Mbits/sec 0.080 ms 0/89643 (0%)[4] Sent
89643 datagrams
iperf Done.
```

iperf3 also has many parameters that can be configured in the process of testing. Users can adjust the test according to the actual application needs. For example, the value of -t parameter can be increased for long-term stress testing, or the -P parameter can be specified for concurrent stress testing of multiple connections. More information about the iperf3 test is available at <https://iperf.fr/iperf-doc.php#3doc>

5. Graphics system

Linux graphics system is a more complex subsystem of Linux system, which has been in constant change. It is located between the device driver related to display and the user interface application. It shields all kinds of hardware differences and provides a uniform API to the upper user interface.

The earliest graphics system under Linux/Unix environment is Xorg graphics system. With the development of software and hardware, especially the development of embedded systems, Xorg becomes too bulky, and then there are some more concise graphics systems that are easy to develop and maintain. Such as Wayland (<https://wayland.freedesktop.org/>). Below is the structure diagram of a graphics system for a typical embedded system.

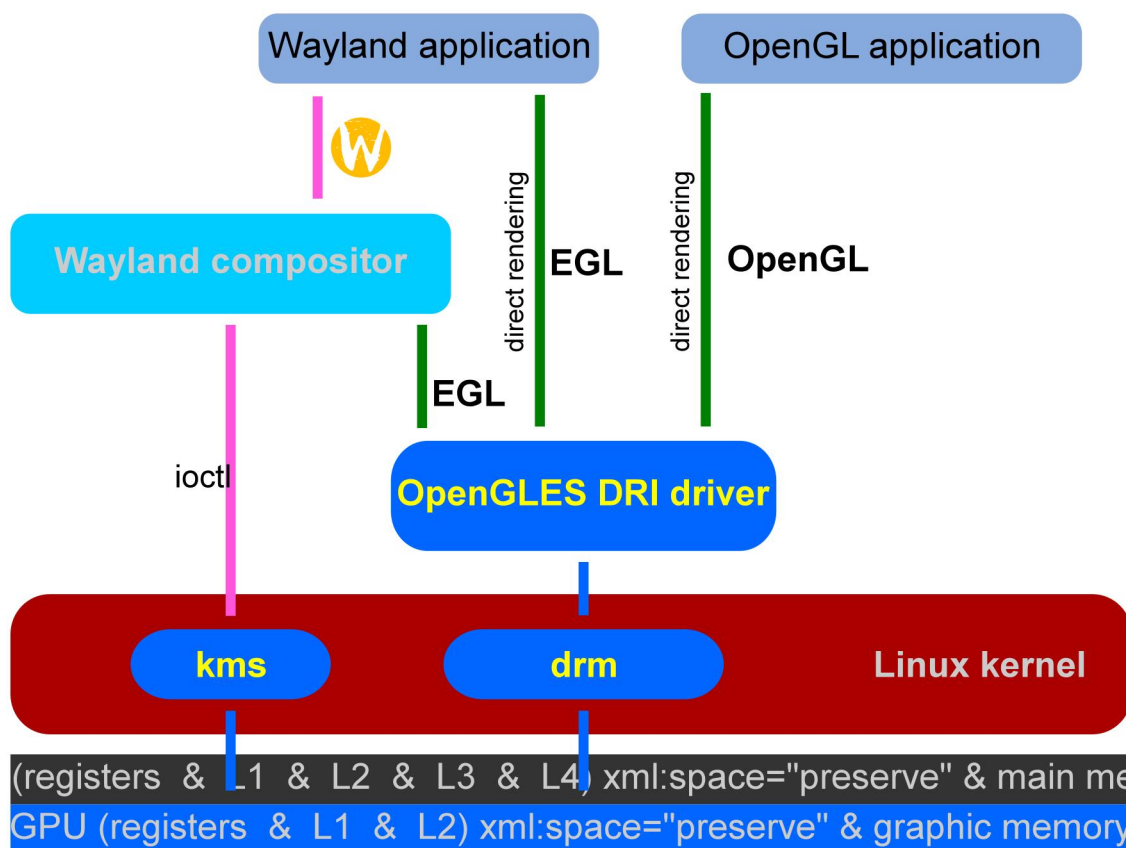


Figure 5-1. Embedded Linux Graphics Stack Overview

(by Shmuel Csaba Otto Traian; GFDL 1.2+ and CC-BY-SA 3.0+; created 2013-11-04)

As you can see above, Wayland is not required for GUI development. For example, the MEasy HMI V2.0 included in our factory image can run Qt5 directly through a platform-specific plugin. The following sections test and illustrate some of the key parts of the graphics system.

5.1. GPU

In MYD-YM62X series development board, YM6254 and YM6252 contain a GPU core, support OpenGL 3.x/2.0/1.1, support 2D and 3D graphics.

The following use development board is located in the /user/share/example/QT under opengl example test:

```
root@myd-am62x:/usr/share/examples/opengl# ls
2dpainting          computegles31      cube               hellogl
es3                 legacy             paintedwindow      qopenglwindo
w                   threadedqopenglwidget
README              contextinfo        hellogl2           hellowi
ndow                opengl.pro         qopenglwidget      textures
```

Test with helloworld:

```
root@myd-am62x:/usr/share/examples/opengl# ./helloworld/helloworld
```

5.2. Wayland+Weston+QT

QT is a cross-platform C++ graphical user interface application development framework. It can be used to develop GUI programs as well as non-GUI programs, such as console tools and servers. Qt is an object-oriented framework that uses special code generation extensions as well as some macros. Qt is easy to extend and allows for true component programming.

Wayland uses Weston graphics desktop, can run QT programs on weston desktop, development board will burn at the factory to provide a rich HMI demonstration system.

1) Run QT routines

QT examples are all in the /user/share/example directory, run a routine:

```
root@myd-am62x:~# /usr/share/examples/widgets/touch/pinchzoom/pinchzoom
```

You can see QT displayed on the screen.

If you want to boot up and run QT programs, add the following to your script:

```
root@myd-am62x:~# cat qt_demo.sh
#!/bin/sh
export QT_WAYLAND_SHELL_INTEGRATION=xdg-shell
export QTWEBENGINE_DISABLE_SANDBOX=1
export QT_QPA_EGLFS_ALWAYS_SET_MODE=1
export WAYLAND_DISPLAY=/run/wayland-0
export XDG_RUNTIME_DIR=/run/user/0
/usr/share/examples/widgets/touch/pinchzoom/pinchzoom
```

2) Run HMI

The MYD-YM62X development board comes with a qt routine for demonstration use. The default is boot, and the running script is /home/root/hmi.sh

```
root@myd-am62x:~# cat hmi.sh
#!/bin/sh
export QT_WAYLAND_SHELL_INTEGRATION=xdg-shell
export QTWEBENGINE_DISABLE_SANDBOX=1
export QT_QPA_EGLFS_ALWAYS_SET_MODE=1
export WAYLAND_DISPLAY=/run/wayland-0
export XDG_RUNTIME_DIR=/run/user/0
flag=`cat /proc/cpuinfo | grep processor | wc -l`
echo "flag ${flag}"

AutoDisabled()
{
file="/etc/systemd/system/graphical.target.wants/weston.service"
pam_name="PAMName=weston-autologin"

if [ ! -e "$file" ]; then
touch "$file"
```



```

fi

if grep -q "$pam_name" "$file"; then
    if ! grep -q "#$pam_name" "$file"; then
        sed -i "s/$pam_name/#$pam_name/" "$file"
    fi
else
    echo "#$pam_name" >> "$file"
fi
}

if [ ${flag} -gt 1 ];then
    /home/root/mxapp2
else
    time=1
    while [ $time -lt 10 ];
    do
        if [ -L /dev/input/touchscreen0 ];then
            break
        else
            sleep 3
            echo ${time}
            time=$((time + 1))
        fi
    done
    systemctl stop weston
    AutoDisabled
    /home/root/lvgl_ethercat_demo
fi

root@myd-am62x:~#

```

Due to its insufficient performance, MYD-YM6231 experiences severe lag when running Weston. When booting with the full image, a script will be used to



o run the LVGL example and automatically disable the startup parameter for Weston, resulting in no display of Weston and HMI.



6. Multimedia applications

6.1. Camera

In this section, the system's own gstreamer, v4l2-utils, and media-ctl are used to evaluate and test the csi camera. Preview and take photos of the main test camera.

1) Camera window preview, take photo test

- **View the basic information of the device**

Plug in the usb camera and the character device /dev/video0 will be generated

```
root@myd-am62x:~# ls -l /dev/video0
crw-rw---- 1 root video 81, 0 Jan 1 1970 /dev/video0
```

- **Use the gstream tool to get a preview**

For a preview of the camera, use the gst-launch command in gstream and execute the following command in the terminal:

```
root@myd-am62x:~# gst-launch-1.0 v4l2src device=/dev/video0 io-mode=2! video/x-raw,format=YUY2,width=640,height=480,framerate=30/1 ! waylandsink
Setting pipeline to PAUSED ...
Pipeline is live and does not need PREROLL ...
Pipeline is PREROLLED ...
Setting pipeline to PLAYING ...
New clock: GstSystemClock
Redistribute latency...
0:00:00.0/99:99:99.
```

When done, the screen generates a preview window with a width of 640, a height of 480, and a frame rate of 30 frames per second

- **Use the gstream tool to do the preview**

Use the `gst-launch` command in `gststream` to take a picture of the camera and save it as a jpeg.

```
root@myd-am62x:~# gst-launch-1.0v4l2src device=/dev/video0 io-mode=2 num-buffers=10! video/x-raw,format=YUY2,width=640,height=480,framerate=30/1! jpegenc ! filesink location=pic.jpeg
Setting pipeline to PAUSED ...
Pipeline is live and does not need PREROLL ...
Pipeline is PREROLLED ...
Setting pipeline to PLAYING ...
New clock: GstSystemClock
Redistribute latency...
Got EOS from element "pipeline0".
Execution ended after 0:00:01.480332314
Setting pipeline to NULL ...
Freeing pipeline ...
Can be pic.jpeg
```

6.2. Audio

This section is test play audio. There are two ways to do this, one is to play the audio directly from the HMI2.0 application, and the other is to play the audio manually through the `gststream` tool.

1) HMI2.0 music playback

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

2) Play music manually

- **Play music in wav format**

```
root@MYD-YM62X:~/audio# aplay att16k.wav
Press 'k' to see a list of keyboard shortcuts.
Now playing /home/root/audio/att16k.wav
```

Redistribute latency...

0:00:14.9/0:00:14.9

Reached end of play list.

- **Play music in mp3 format**

```
root@myd-am62x:~# ls
```

```
Born a Stranger.mp3  The promise.mp3
```

In the terminal, run the following command to play music in MP3 format:

```
root@myd-am62x:~# aplay The\ promise.mp3
```

```
Press 'k' to see a list of keyboard shortcuts.
```

```
Now playing /usr/share/myir/Music/The promise.mp3
```

```
Redistribute latency...
```

```
0:02:54.0/0:03:15.8
```

7. System Tools

Some common system tools are included in the default image, which is convenient for users to view and manage the various resources of the system in system debugging or actual deployment of the product, and can also be invoked in SHELL scripts or other applications. These tools may not fully meet the user's system customization needs, at this time, the system developer needs to make appropriate adjustments according to the actual situation.

7.1. Compression and decompression tools

This section mainly tests the decompression tool of the system. Compression is the ability to compress multiple files into a compressed package can be compressed multiple files into a compressed package, convenient for file transmission. Decompression can be compressed after the compression of the compressed file back to the original size for easy use. In this section, we will use tar, gzip, gunzip, and other filesystems as examples.

1) The tar tool

Now we use the tar tool in Linux, it can not only package a file, but also compress, view, add and decompress it. Here is the package operation.

- **Syntax Formatting**

Enter the following to see the tar syntax format:

```
root@myd-am62x:~# tar --help
```

```
BusyBox v1.33.0 (2023-05-19 10:43:45 CST) multi-call binary.
```

```
Usage: tar c|x|t [-ahvokO] [-f TARFILE] [-C DIR] [-T FILE] [-X FILE] [OPTION]...  
[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
-a      (De)compress based on extension
-h      Follow symlinks
-T FILE File with names to include
-X FILE File with glob patterns to exclude
--exclude PATTERN      Glob pattern to exclude
--overwrite            Replace existing files
--strip-components NUM  NUM of leading components to strip
--no-recursion          Don't descend in directories
--numeric-owner         Use numeric user:group
--no-same-permissions   Don't restore access permissions
--to-command COMMAND   Pipe files to COMMAND 567.001398] delay
                        timeout.

```

The detailed parameter description is as follows:

- -c: Set up a compressed file parameter directive (meaning create);
- -x: unpack a compressed file parameter directive!
- -t: View the files in the tarfile! Note that only one c/x/t can exist in the parameter setting! Can not exist at the same time! Because it is impossible to compress and decompress at the same time.
- -j: Does it have the property of bzip2 at the same time? That is, does it need to be compressed with bzip2?
- -v: Display the file during compression! This is often used, but not recommended for background running processes!
- -f: Use the filename, please note that after the f should be immediately followed by the filename, do not add parameters! For example, "tar-z

cvfP tfile sfile" is the wrong way to write it, "tar-zcvPf tfile sfile" is the right way.

- -p: Use the original attributes of the original file (the attributes do not change depending on the user)
- -P: Can use absolute paths to compress!
- -N: Dates newer than yyyy/mm/dd will be packed into the new file!
- --exclude FILE: Do not bundle files during compression!

● Use tar compression

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

```
root@myd-am62x:~# tar -cvf test.tar.gz test.txt
root@myd-am62x:~# ls
OpenAMP_TTY_echo.elf  rs485_write  test.tar.gz  uart_test
rs485_read            test.c       test.txt    wifi.conf
```

So with the z parameter above, tar.gz or.tgz represents a gzipped tar file.

● Extract the tar file using tar

Unzip the tar.gz file

```
root@myd-am62x:~# tar -xvf test.tar.gz
test.txt
root@myd-am62x:~# ls
OpenAMP_TTY_echo.elf  rs485_write  test.tar.gz  uart_test
rs485_read            test.c       test.txt    wifi.conf
```

2) gzip compression tool

gzip is a command that is often used to compress and decompress files in Linux systems, which is convenient and easy to use.

● Syntax Format

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

```
root@myd-am62x:~# gzip --help
BusyBox v1.31.1 () multi-call binary.
```

Usage: gzip [-cfkdt] [FILE]...

- **gzip the file**

```
root@myd-am62x:~# gzip test.txt
root@myd-am62x:~# ls
OpenAMP_TTY_echo.elf  rs485_write  test.tar.gz  uart_test
rs485_read            test.c       test.txt.gz  wifi.conf
```

- **Unzip it with gunzip tool**

Unzip the file with gunzip, as shown in the following example:

```
root@myd-am62x:~# gunzip test.txt.gz
root@myd-am62x:~# ls
OpenAMP_TTY_echo.elf  rs485_write  test.tar.gz  uart_test
rs485_read            test.c       test.txt     wifi.conf
```

7.2. Filesystem Tools

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

1) mount the mount tool

mount is a Linux command that connects a partition to a folder in Linux, so that if you access the folder, you are accessing the partition. The syntax is as follows:

```
root@myd-am62x:~# mount -h
mount: invalid option -- 'h'
BusyBox v1.33.0 (2023-05-19 10:43:45 CST) multi-call binary.
```

Usage: mount [OPTIONS] [-o OPT] DEVICE NODE

Mount a filesystem. Filesystem autodetection requires /proc.

An example of mounting the fourth partition of an sd card is as follows:

```
root@myd-am62x:~# mount /dev/mmcb1k1p4 /mnt/
[10677.124115] EXT4-fs (mmcb1k1p4): mounted filesystem with ordered data
mode.Opts: (null)
[10677.130984] ext4 filesystem being mounted at /mnt supports timestamps
until 2038 (0x7fffffff)
```

2) mkfs format tool

After the hard drive is partitioned, the next job is the setup of the Linux file system. Similar to formatting hard drives under Windows. Setting up a file system on a hard drive partition will erase the data on the partition and is not recoverable, so make sure the data on the partition is no longer in use before setting up a file system. The mkfs command to create a filesystem is in the following format:

```
root@myd-am62x:~# mkfs.
mkfs.ext2  mkfs.ext3  mkfs.ext4  mkfs.fat  mkfs.msdo  mkfs.vfat
```

An example of formatting the 7th partition of the sd card is as follows:

```
root@myd-am62x:~# mkfs.ext3 -c /dev/mmcb1k1p7
mkfs from util-linux 2.32.1
mkfs.ext3 -c /dev/mmcb1k1p7
mke2fs 1.44.3 (10-July-2018)
/dev/mmcb1k1p7 contains a ext4 file system labelled 'userfs'
      created on Thu Aug 13 04:32:02 2020
Proceed anyway? (y,N) y
Creating filesystem with 330532 1k blocks and 82656 inodes
Filesystem UUID: 46bd5bb8-1882-4a68-901a-e11b4e71924b
Superblock backups stored on blocks:
      8193, 24577, 40961, 57345, 73729, 204801, 221185
Checking for bad blocks (read-only test): done
Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
```

3) fsck File Repair Tool

The fsck command is mainly used to check the correctness of the file system.

When a file system error occurs, the fsck command can be used to try to repair it. And repair the Linux disk. For example:

```
root@myd-am62x:~# fsck -a /dev/mmcblk1p7
fsck from util-linux 2.32.1
/dev/mmcblk1p7: clean, 11/82656 files, 20726/330532 blocks
```

4) dumpe2fs

Print information about the existing super blocks and blocks groups of the file system on a particular device. To see the application syntax, enter the following command on the development board:

```
root@myd-am62x:~# dumpe2fs -h
dumpe2fs 1.45.4 (23-Sep-2019)
Usage: dumpe2fs [-bfghimxV] [-o superblock=<num>] [-o blocksize=<num>]
device
```

To view the detailed properties of the filesystem after formatting, for example, enter the command to view the details of a certain disk:

```
root@myd-am62x:~# dumpe2fs /dev/mmcblk0p50
dumpe2fs 1.45.6 (20-Mar-2020)
Filesystem volume name:   <none>
Last mounted on:         /mnt
Filesystem UUID:         d4815f27-a633-453f-9b2a-14f8d14aab9d
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:      has_journal ext_attr resize_inode dir_index filetype ext
ent 64bit flex_bg sparse_super large_file huge_file dir_nlink extra_isize metadat
a_csum
Filesystem flags:         unsigned_directory_hash
```

Check the number of inodes on a disk; inodes also consume hard disk space, so when a hard disk is formatted, the operating system automatically splits the hard disk into two regions. One is the data area, where file data is stored;

The other is the inode table, which holds the information contained in the i node.

```
root@myd-am62x:~# dumpe2fs /dev/mmcb1k0p50| grep -i "inode size"
Dumpe2fs 1.45.6 (20 - Mar - 2020)
Inode size:                256
```

Looking at the number of blocks on a disk, the operating system does not read the hard disk sector by sector, which would be inefficient, but reads multiple sectors in succession at once, that is, one "block" at a time. This "block" composed of multiple sectors is the smallest unit of file access.

```
root@myd-am62x:~# dumpe2fs /dev/mmcb1k0p50| grep -i "block size"
Dumpe2fs 1.45.6 (20 - Mar - 2020)
Block size:                4096
```

7.3. Disk Management Tools

The main test system disk management tools, this section will introduce several common disk management tools. The system comes with disk management tools fdisk, dd, mkfs, du, df, cfdisk, fsck. Through these commands, you can monitor the usual disk usage.

1) fdisk disk partitioning tool

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

```
root@myd-am62x:~# fdisk -h
Usage:
fdisk [options] <disk>      change partition table
fdisk [options] -l [<disk>] list partition table(s)
```

Partition the eMMC:

```
root@myd-am62x:~# fdisk /dev/mmcb1k2p2
Welcome to fdisk (util-linux 2.32.1).
```

Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
The old ext4 signature will be removed by a write command.
Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xd2dcd6ef.
Command (m for help):

2) dd copy command

The dd command copies a specified input file to a specified output file. And you can do format conversion during the copying process. The difference between the dd command and the cp command is that the dd command can be performed on a floppy disk without the creation of a file system, and the data copied to the floppy disk is actually an image file. Similar to the role of the DOS diskcopy command. The format of the dd command is: dd [<if= input filename/device name >] [<of= output filename/device name >] [bs= block byte size] [count = number of blocks]

An example of creating a file of size 2M is as follows:

```
root@myd-am62x:~# time dd if=/dev/zero of=ffmpeg1 bs=2M count=1 conv=fsync
1+0 records in
1+0 records out
2097152 bytes (2.1 MB, 2.0 MiB) copied, 0.195515 s, 10.7 MB/s
real 0m 0.20s
user 0m 0.00s
sys 0m 0.05s
```

3) du Disk Usage Statistics Tool:

The du command is used to display disk space usage. This command shows, level by level, how each level of subdirectories of a specified directory occupies filesystem data blocks. The general syntax for du is as follows:

```
root@myd-am62x:~# du --help
Usage: du [OPTION]... [FILE]...
or: du [OPTION]... --files0-from=F
```

Summarize disk usage of the set of FILEs, recursively for directories.

Some of the parameters:

- -a: Shows the size of all directories or files
- -h: In K,M,G to improve the readability of information
- -k: Output in kilobytes
- -m: Output in megabytes

Count the size of the file generated by the dd command:

```
root@myd-am62x:~# du ffmpeg1
2048    ffmpeg1
root@myd-am62x:~# du -h ffmpeg1
2.0M ffmpeg1
```

4) df Disk Statistics Tool:

It is used to display the disk usage statistics of the current file system on the Linux system. The general usage is as follows:

```
root@myd-am62x:~# df --help
Usage: df [OPTION]... [FILE]...
Show information about the file system on which each FILE resides,
or all file systems by default.
```

Some parameter notes:

- -h: Can be displayed using appropriate units depending on the size used
- -i: View the number of inodes under the partition and the usage of inodes
- -T: Prints the filesystem type

7.4. Process management tools

Process is also an important concept in the operating system, it is an execution process of a program, program is a static description of the process, every program running in the system is run in its process. All processes in Linux system are related to each other, in addition to the initialization process, all processes have a parent process. New processes are not created, but are copied.

ied or copied from previous processes. All processes in Linux are derived from an init process with process number 1. The Linux system includes three different types of processes, each with its own characteristics and properties:

- Interactive processes: Processes that are started by a Shell and can run in both the foreground and background.
- Batch processes: These processes are not associated with the terminal and are a sequence of processes. A process that is submitted to a waiting queue to be executed sequentially.
- Monitor processes (daemons) : Daemons are always active and usually run in the background. Daemons are usually started by the system at the beginning by automatically activating scripts or root
- For the linux system, the process management is an important link, for the process management is usually through the process management tool, the Linux system more commonly used process management command has the following: ps top vmstat kill.

1) ps shows the current process tool

Display the current system process running status, the general syntax is as follows:

```
root@myd-am62x:~# ps --help
Usage:
ps [options]

Try 'ps --help <simple|list|output|threads|misc|all>'
or 'ps --help <s|l|o|t|m|a>'
for additional help text.
For more details see ps(1).
```

Instructions for some parameter combinations:

- -u: user-centered organization process status information display;
- -a: terminal independent process;
- -x: processes related to the terminal; (threads, which are lightweight processes;)

Usually the above commands are combined with: aux.

- --e: Show all processes; Equivalent to ax;
- -f: Display complete format program information;
Usually: ef is used in combination with the above commands
- -H: This displays the number of processes at the process level
- -F: Displays more program information

The command: eHF is usually used in combination

An example of showing information about all processes is as follows:

```
root@myd-am62x:~# ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root    1  0.6  1.1 23804 4820? Ss   06:57   0:07 /sbin/init
root    2  0.0  0.0  0  0? S    06:57   0:00 [kthreadd]
root    3  0.0  0.0  0  0? I<   06:57   0:00 [rcu_gp]
root    4  0.0  0.0  0  0? I<   06:57   0:00 [rcu_par_gp]
root    8  0.0  0.0  0  0? I<   06:57   0:00 [mm_percpu_wq]
root    9  0.0  0.0  0  0? S    06:57   0:00 [ksoftirqd/0]
root   10  0.0  0.0  0  0? I    06:57   0:00 [rcu_preempt]
root   11  0.0  0.0  0  0? S    06:57   0:00 [migration/0]
root   12  0.0  0.0  0  0? S    06:57   0:00 [cpuhp/0]
root   13  0.0  0.0  0  0? S    06:57   0:00 [cpuhp/1]
```

A brief explanation of some of its taskbars:

- VSZ: vittual memory size
- RSS: resident size, resident memory set
- STAT: There are several process states
 - R: runing
 - S: interruptable sleeping to interrupt sleep
 - D: uninterruptable sleeping
 - T: stopped
 - Z: zombie Zombie process
 - +: Foreground process
 - N: Low priority process
 - I: Multi-threaded process

2) top shows the linux process

The top command puts quite a bit of overall system performance information on one screen. The display can also be changed interactively. Dynamically and continuously monitor the running state of the process, the top syntax is generally as follows:

```
root@myd-am62x:~# top -help
procps-ng 3.3.16
Usage:
top -hv | -bcEHiOSs1 -d secs -n max -u|U user -p pid(s) -o field -w [cols]
```

Dynamic view system process sample is as follows:

```
root@myd-am62x:~# top
top-07:19:53 up 22 min, 1 user, load average: 0.00, 0.03, 0.05
Tasks: 109 total, 1 running, 108 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0us, 1.3sy, 0.0ni, 98.7id, 0.0wa, 0.0hi, 0.0si, 0.0st
MiB Mem: 425.8 total, 255.1 free, 56.6 used, 114.1 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 351.6 avail Mem
  PID USER  PR  NI  VIRT    RES  SHR S  %CPU  %MEM  TIME+  COMMAND
  593 root  20   0  2348  1596  1056 S   1.3   0.4  0:25.27 apps.plugin+
  553 root  20   0 129684 14196 2076 S   0.7   3.3  0:13.86 netdata
  594 root  20   0  2692  2140  1608 S   0.7   0.5  0:07.95 bash
```

3) kill process termination tool

Send the specified signal to the corresponding process. Failing to specify a model will send SIGTERM (15) to terminate the specified process. If the program cannot be terminated, use the "-KILL" parameter, which sends a signal of SIGKILL(9), which will force the termination of the process. Use the ps command or the jobs command to see the process number. The root user will affect the user's process, non-root users can only affect their own process. The general syntax of the kill command is as follows:

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

Description of some parameter combinations:

- -s: Specifies the signal to send
- -p: Simulate the signal being sent

- -l: Specify a list of names for the signal
- pid: The ID number of the process to be aborted
- Signal: Denotes a signal

Start by using `ps -ef` with the pipe command to determine the PID of the process to kill.

```
root@myd-am62x:~# ps -ef | grep mxapp2
root      1045      1 15 15:51 ?        00:00:02 /home/mxapp2 -platform eglfs
root      1174    1117  0 15:51 ttySTM0  00:00:00 grep mxapp2
```

Then enter the following command to terminate the process:

```
root@myd-am62x:~# kill 1045
```

The `killall` command kills all processes within the same process group, allowing the name of the process to be killed to be specified instead of the PID process number.

```
root@myd-am62x:~# killall mxapp2
root@myd-am62x:~#
```

8. Development support

This chapter mainly introduces some basic information about the secondary development of the current SDK. For related information of the SDK, please refer to the MYD-YM62X SDK Release Notes.

8.1. Development Language

8.1.1. SHELL

A Shell is a program written in C that is the user's bridge to Linux. Shell is not only a command language, but also a programming language. There are many kinds of common Linux Shell, common 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)

The highlight here is bash, which is supported by default by the current SDK.

The following example shows the image of MEasy HMI 2.0 starting from /etc/init.d/S99hmi with the following contents:

```
! /bin/sh
Export LD_LIBRARY_PATH = /lib: /usr/lib/usr/local/lib/out_qt_5.15.2 /lib
export QT_QPA_PLATFORM=eglfs
export QT_QPA_EGLFS_INTEGRATION=none
export QT_QPA_FONTDIR=/usr/share/fonts
export QT_QPA_PLATFORM_PLUGIN_PATH=/out_qt_5.15.2/plugins
export QML_IMPORT_PATH=/out_qt_5.15.2/qml/
export QML2_IMPORT_PATH=/out_qt_5.15.2/qml/
export QML_IMPORT_PATH=/out_qt_5.15.2/qml
export QMLSCENE_DEVICE=softwarecontext
sleep 1
```



15.2 / / out_qt_5 mxapp2 / mxapp/mxapp2 - platform eglfs &

This script first sets the MEasy HMI environment variables, then calls the /home/mxapp2 executable to start MEasy HMI based on the utility parameters, with the "&" at the end of the parameter indicating that the program will run in the background.

8.1.2. C/C++

C/C++ is the most common programming language for low-level application development on Linux platform, and it is also the most efficient language after assembly. Using C/C++ for development usually adopts the way of cross development, that is, the development is carried out on the development host side, compiled to generate the binary execution file running on the target machine, and then deployed to the target machine for running.

In this way, you first need to install the SDK. Please refer to the "MYD-YM62 X_Linux Software Development Guide" for installation steps. After installation, you need to configure the SDK environment, as follows:

```
PC $:source /home/my-linux/Yocto/SDK/environment-setup-cortexa7t2hf-neon-vfpv4-openstlinux_eglfs-linux-gnueabi
PC $:arm-openstlinux_eglfs-linux-gnueabi-gcc -march=armv7ve -mthumb -mfp
u=neon-vfpv4 -mfloat-abi=hard -mcpu=cortex-a7 --sysroot=/home/my-linux/Y
octo/SDK/sysroots/cortexa7t2hf-neon-vfpv4-openstlinux_eglfs-linux-gnueabi
```

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

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

Then compile the application, here using \$CC, because the compiler needs the corresponding header file and link, \$CC contains the corresponding system library and configuration information, if you directly compile with arm-openstlinux_eglfs-linux-gnueabi-gcc, it will not find the header file. At this time, you can add the parameter -v to view the detailed link process.

```
$CC -v hello.c -o hello
Using built-in specs.
COLLECT_GCC=arm-openstlinux_eglfs-linux-gnueabi-gcc
```

```
COLLECT_LTO_WRAPPER=/home/my-linux/Yocto/SDK/sysroots/x86_64-openstlin
ux_eglfs_sdk-linux/usr/libexec/arm-openstlinux_egl Fs - Linux - gnueabi/GCC/ar
m - openstlinux_eglfs - Linux - gnueabi 8.2.0 / lto - wrapper
Target: arm-openstlinux_eglfs-linux-gnueabi
...
```

Then the generated executable through the SCP command to the copy to the target machine to perform, the results are as follows:

```
root@myd-am62x:~# ./hello
hello world!
```

For more complex examples and development methods, refer to the instructions in the application migration section of the MYD-YM62X_Linux Software Development Guide.

8.1.3. Python

Python is an interpreted, object-oriented, dynamically typed high-level programming language. Python was invented by Guido van Rossum in late 1989 and the first public distribution was released in 1991. Like Perl, Python source code is available under the GNU General Public License (GPL). This section focuses on testing the use of python, both from the python command line and scripting.

1) See which versions of Thon are supported on your system

```
root@myd-am62x:~# python3.10-V
Python 3.10.9
```

2) python command-line tests

Start python and type the following text message into the python prompt, then press Enter to see it run:

```
root@myd-am62x:~# python3
Python 3.10.9 (main, Dec 6 2022, 18:44:57) [GCC 11.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

In Python 3.10.9, the output from this example is as follows:

```
Hello, Python!
```

```
>>>
```

To exit Python, execute `exit()` :

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

```
root@myd-am62x:~#
```

3) Write a script to test Python

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

```
root@myd-am62x:~# vi test.py
```

```
root@myd-am62x:~# cat test.py
```

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

```
print ("Hello, Python!" )
```

To execute the script file, Python3 interpreter in /usr/bin directory, use the following command to execute the script.

```
root@myd-am62x:~# chmod +x test.py
```

```
root@myd-am62x:~# ./test.py
```

```
Hello, Python!
```

Calling the Python3 interpreter with the script argument starts executing the script until it is finished executing. When the script has finished executing, the interpreter is no longer valid. The current system does not support pip command and pip to install python package, if the customer needs pip tools need to be ported.



9. Resources

- **Yocto official website**

<https://Yocto.org/>

- **The Linux kernel watchdog is introduced**

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

- **The Qt embedded Linux**

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

- **TI AM62X Wiki**

https://software-dl.ti.com/processor-sdk-linux-rt/esd/AM62X/09_00_00_03/exports/docs/devices/AM62X/index.html



Appendix A

Warranty & Technical Support Services

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

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

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

Service Guarantee

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

Price

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

Delivery Time

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

Technical Support

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

After-sale Service

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

The service covers:

Technical support service

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

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

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

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

After-sales maintenance service

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

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

- Due to unauthorized installation of the software, system or incorrect configuration or computer virus which has caused the damage of products.

Warm tips

1. MYIR does not supply maintenance service to LCD. We suggest the customer first check the LCD when receiving the goods. In case the LCD cannot run or no display, customer should contact MYIR within 7 business days from the moment get the goods.
2. Please do not use finger nails or hard sharp object to touch the surface of the LCD.
3. MYIR suggests user purchasing a piece of special wiper to wipe the LCD after long time use, please avoid clean the surface with fingers or hands to leave fingerprint.
4. Do not clean the surface of the screen with chemicals.
5. Please read through the product user manual before you using MYIR's products.
6. For any maintenance service, customers should communicate with MYIR to confirm the issue first. MYIR's support team will judge the failure to see if the goods need to be returned for repair service, we will issue you RMA number for return maintenance service after confirmation.

Maintenance period and charges

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

Shipping cost

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

Products Life Cycle

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

Value-added Services

1. MYIR provides services of driver development base on MYIR's products, like serial port, USB, Ethernet, LCD, etc.
2. MYIR provides the services of OS porting, BSP drivers' development, API software development, etc.
3. MYIR provides other products supporting services like power adapter, LCD panel, etc.
4. ODM/OEM services.

MYIR Electronics Limited

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

Yunli Intelligent Park, Bantian, Longgang District.

Support Email: support@myirtech.com

Sales Email: sales.en@myirtech.com

Phone: +86-755-22984836

Fax: +86-755-25532724

Website: www.myirtech.com