

MYD-LD25X Linux Software Evaluation Guide



File status: [] Draft [✓] Release	FILE ID:	MYIR-MYD-LD25X-SW-EG-EN-L6.1.82
	VERSION:	V1.2[Doc]
	AUTHOR:	MSW0407
	RELEASE:	2024-09-15
	UPDATED:	2024-11-28

Revision History

VERSION	AUTHOR	PARTICIPANT	DATE	DESCRIPTION
V1.0[Doc]	MSW0407	MSW0041	2024-09-15	Official Release
V1.1[Doc]	MSW0407	MSW0041	2024-11-08	The adaptation for the MYD-LD257-8E1D model.
V1.2[Doc]	MSW0407	MSW0041	2024-11-28	Add Audio Recording Test

CONTENT

Revision History	- 1 -
CONTENT	- 2 -
1. Overview	- 4 -
1.1. Hardware Resources	- 4 -
1.2. Software Resources	- 5 -
1.3. Documentation Resources	- 5 -
1.4. Environment Preparation	- 5 -
1.5. Resource List	- 6 -
2. Core Resources Evaluation	- 8 -
2.1. CPU	- 8 -
2.2. GPU	- 14 -
2.3. Memory	- 17 -
2.4. eMMC	- 21 -
2.5. RTC	- 25 -
2.6. Watchdog	- 27 -
2.7. eeprom	- 30 -
2.8.	- 31 -
3. Basic Peripheral Interfaces	- 32 -
3.1. GPIO	- 32 -
3.2. LED	- 36 -
3.3. Key	- 37 -
3.4. USB	- 40 -
3.5. USB-OTG	- 43 -
3.6. Micro SD card	- 47 -
3.7. LVDS Display	- 50 -
3.8. Touch Panel	- 54 -
4. RS232、RS485、CAN	- 58 -

4.1. RS232	- 60 -
4.2. RS485	- 62 -
4.3. CAN	- 65 -
5. Network.....	- 68 -
5.1. Network Devices	- 68 -
5.2. Network applications	- 83 -
6. Multimedia application system	- 99 -
6.1. Camera	- 99 -
6.2. Audio	- 102 -
6.3. Video Playback	- 103 -
7. References	- 104 -
Appendix A.....	- 105 -

1. Overview

The Linux Software Evaluation Guide is designed to introduce the testing procedures and evaluation methods for core resources and peripheral resources running on the open-source Linux system on the MYiR development board. This document serves as an initial assessment guide and can also be used as a general testing manual for system development.

1.1. Hardware Resources

This document applies to the MYiR Electronics MYD-LD25X series boards, which are embedded development platforms based on high-performance ARM processors from ST. The core chip used in this series is the STM32MP257D. Table 1-1 outlines the chip models and their main features for the MYD-LD25X:

Table 1-1. Board Model Introduction

Core Chip Model	Processor Core	Core Board	Base Board
STM32MP257D	Quad- ARMCortex-A35	MYC-LD257-8E2D-150-C	MYD-LD257-8E2D-150-I
		MYC-LD257-8E2D-150-I	MYD-LD257-8E2D-150-C
		MYC-LD257-8E1D-150-C	MYD-LD257-8E1D-150-I
		MYC-LD257-8E1D-150-I	MYD-LD257-8E1D-150-C

For detailed hardware configuration parameters, please refer to the "MYD-LD25X Product Manual-V1.0" document. During the evaluation and testing process, users may utilize some optional modules, listed below in Table 1-2.

Table 1-2. Optional Modules

Accessories	Interface Type	Explanation and Link
Communication Interface Module	2x20PIN Interface	MY-WIREDCOM https://en.myir.cn/c/MY-WIREDCOM.html
LCD Screen	LVDS Interface	MY-LVDS070C https://en.myir.cn/c/MY-LVDS070C.html
Camera Module	CSI Interface	MY-CAM003M

1.2. Software Resources

The BSP (Board Support Package) of the MYD-LD25X series development board is derived from the porting and modification of the official ST open-source community version Linux BSP. The system image is built using the yocto project. All software resources for the Bootloader, Kernel, and file system are provided in source code form. For more details, please refer to the "MYD-LD25X SDK Release Note"

The development board has been flashed with the image when it leaves the factory. You can use it by simply powering it on.

1.3. Documentation Resources

According to the different stages of user use with the development board, the SDK provides various categories of documentation and manuals.

1.4. Environment Preparation

Before starting to evaluate the board's software, you need to make necessary preparations for the board and configure the environment, including correct hardware wiring, configuration debugging port, setting up booting, etc. Detailed steps can be found in the "MYD-LD25X Quick Start Guide".

The next focuses on how to evaluate and test system hardware resources, interfaces, and software functions. It mainly uses some commonly-used tools and commands in Linux, as well as applications you developed for the tests. The software evaluation guide is described in several parts including, core resources,

peripheral resources, network applications, multimedia applications, development support applications, system tools, etc. The later chapters will provide comprehensive explanations for each part, and describe the specific evaluation methods and steps for each resources in detail.

1.5. Resource List

Table1-3. Evaluation List

Sequence	Functions	Notes
1	CPU	Dual-core Arm® Cortex®-A35
2	eMMC	8GB eMMC
3	RTC	LK8563T chip
4	Watchdog	Internal watchdog testing
5	GPIO	General-purpose input/output (GPIO)
6	LED	LED lighting devices
7	KEY	Button control
8	RS232	Multiple communication methods, Requires MY-WIREDCOM module
9	RS485	
10	CAN	
11	USB	USB storage device
12	USB-OTG	Type-C to USB
13	TF	microSD card storage device
14	LVDS	Requires MY-LVDS070C display
15	Touch Panel	
16	Ethernet	YT8531SH

17	WIFI	AP6256 Wi-Fi and Bluetooth chip
18	Bluetooth	AP6256 Wi-Fi and Bluetooth chip
19	Camera	Requires MY-CAM003M camera
20	Audio	ES8388 audio chip

2. Core Resources Evaluation

In Linux system, the proc virtual file system is provided to query the parameters of various core resources and some common tools to evaluate the performance of resources. The parameters of core resources such as CPU, memory, eMMC, RTC, etc. will be read and tested in detail below.

2.1. CPU

There are many types of core boards in the MYD-LD25X series development boards. Please refer to Table 1-1. The -C at the end of the model number represents a commercial-grade development board, and the -I represents an industrial-grade development board. All of them are equipped with an Arm® Cortex®-A35*2 processor with a maximum main frequency of 1.5GHz.

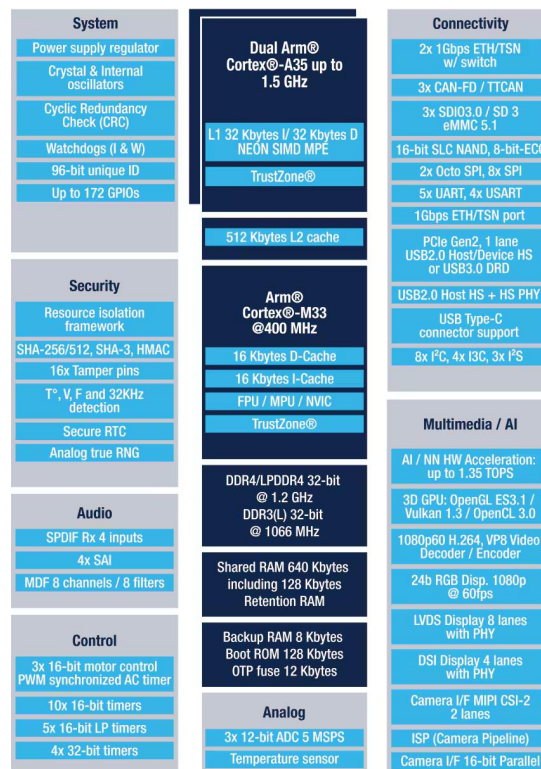


Figure 2-1. CPU Resource Diagram

The status or functions of the CPU will be tested by the following commands.

1) Command to view CPU information:

You can get the vendor and parameter information of the CPU in the system by reading the /proc/cpuinfo file.

```
root@myd-ld25x:~# cat /proc/cpuinfo
processor : 0
BogoMIPS : 80.00
Features : fp asimd evtstrm crc32 cpuid
CPU implementer : 0x41
CPU architecture: 8
CPU variant : 0x1
CPU part : 0xd04
CPU revision : 0

processor : 1
BogoMIPS : 80.00
Features : fp asimd evtstrm crc32 cpuid
CPU implementer : 0x41
CPU architecture: 8
CPU variant : 0x1
CPU part : 0xd04
CPU revision : 0
```

- Processor: The number of logical processing cores in the system can be the physical cores or the logical cores virtualized by using the Hyper-Threading technology for multi-core processors.
- BogoMIPS: Rough estimate of CPU's million instructions per second (Million Instructions Per Second) calculated when the system kernel is started.

2) View CPU utilization

```
root@myd-ld25x:~# top
```

```
top - 01:32:04 up 0 min,  2 users,  load average: 0.94, 0.27, 0.09
```

```
Tasks: 154 total,  1 running, 153 sleeping,  0 stopped,  0 zombie
```

```
%Cpu(s): 16.7 us, 33.3 sy,  0.0 ni, 50.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
```

```
MiB Mem :  1832.6 total,  1368.5 free,   302.6 used,   243.8 buff/cache
```

```
MiB Swap:    0.0 total,    0.0 free,    0.0 used. 1529.9 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME	COMMAND
1755	root	20	0	13184	4644	2628	R	11.8	0.2	0:00.04	top
1	root	20	0	17368	9820	6968	S	0.0	0.5	0:02.43	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.01	kthrea
3	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_par
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	slub_flu
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	netns
7	root	20	0	0	0	0	I	0.0	0.0	0:00.00	kworke
8	root	0	-20	0	0	0	I	0.0	0.0	0:00.11	worker
9	root	20	0	0	0	0	I	0.0	0.0	0:00.15	kworke
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_pe
11	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tas
12	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tas

13	root	20	0	0	0	0 S	0.0	0.0	0:00.06	ksoftir
qd/0										
14	root	20	0	0	0	0 I	0.0	0.0	0:00.03	rcu_pre
empt										

- %usr: Represents the CPU usage of user space programs (not scheduled through nice)
- %sys: Represents the CPU usage of the system space, mainly the kernel program
- %nic: nic: Represents the CPU usage of user space programs that have been scheduled through nice
- %idle: Idle CPU
- %irq: Number of hard interrupts handled by the CPU
- %sirq: Number of soft interrupts handled by the CPU

3) Acquire CPU Temperature Information

The CPU has an integrated temperature sensor for CPU temperature collection, which can conveniently capture the internal temperature of the CPU.

```
root@myd-ld25x:~# cat /sys/class/thermal/thermal_zone0/temp
39790
```

The obtained value divided by one thousand is the actual temperature.

4) CPU Stress Test

There are many ways to stress test the CPU, and stability during calculations can be tested by using the bc command to calculate pi.

```
root@myd-ld25x:~# echo "scale=5000; 4*a(1)" | bc -l -q &
```

The above command generates Pi in the background, accurate to 5000 decimal spots. The calculation process can take some time. In the meantime, we can check the change in CPU utilization with the top command, as shown below:

```
root@myd-ld25x:~# top
Mem: 128012K used, 16424K free, 8948K shrd, 3904K buff, 43580K cached
CPU: 50% usr 4% sys 0% nic 45% idle 0% io 0% irq 0% sirq
Load average: 0.15 0.03 0.02 2/77 416
```

PID	PPID	USER	STAT	VSZ	%VSZ	%CPU	COMMAND
415	328	root	R	2744	2%	50%	bc -l -q
186	1	systemd-	S	80920	56%	0%	/lib/systemd/systemd-timesyncd
154	1	root	S	17096	12%	0%	/lib/systemd/systemd-journald
168	1	root	S	12580	9%	0%	/lib/systemd/systemd-udev
1	0	root	S	10052	7%	0%	{systemd} /sbin/init
349	1	systemd-	S	7540	5%	0%	/lib/systemd/systemd-networkd
351	1	systemd-	S	6972	5%	0%	/lib/systemd/systemd-resolved
350	1	root	S	6728	5%	0%	/lib/systemd/systemd-logind
352	1	avahi	S	4852	3%	0%	avahi-daemon: running [myd-ld25x.local]
354	352	avahi	S	4720	3%	0%	avahi-daemon: chroot

After about 3 minutes, the result of PI is calculated. The CPU usage on this device is high, with no exceptions occurring, indicating that the CPU stress test can pass. By increasing the precision requirements, the test pressure can be further increased.

```
root@myd-ld25x:~# 3.141592653589793238462643383279502884197169399375
105820974944592307\
8164062862089986280348253421170679821480865132823066470938446095505
8\
2231725359408128481117450284102701938521105559644622948954930381964
4\
```

```

2881097566593344612847564823378678316527120190914564856692346034861
0\
4543266482133936072602491412737245870066063155881748815209209628292
5\
4091715364367892590360011330530548820466521384146951941511609433057
2\
7036575959195309218611738193261179310511854807446237996274956735188
5\
7527248912279381830119491298336733624406566430860213949463952247371
9\
0702179860943702770539217176293176752384674818467669405132000568127
1\
4526356082778577134275778960917363717872146844090122495343014654958
5\
3710507922796892589235420199561121290219608640344181598136297747713
0\
9960518707211349999998372978049951059731732816096318595024459455346
9\
0830264252230825334468503526193118817101000313783875288658753320838
1\
4206171776691473035982534904287554687311595628638823537875937519577
8\
1857780532171226806613001927876611195909216420198938095257201065485
8\
6327886593615338182796823030195203530185296899577362259941389124972
1\
7752834791315155748572424541506959508295331168617278558890750983817
5\
.....
[1]+  Done

```

2.2. GPU

The STM32MP257D chip includes a GPU: an OpenGL ES 3.1 module, which extends the OpenGL framework and supports advanced graphics features such as compute shaders and enhanced texture processing, thereby improving graphics performance and effects. It supports 3D acceleration. To connect an LVDS display or an HDMI display as described in section 3.7, execute the following command:

```
root@myd-ld25x:~# killall mxapp2
root@myd-ld25x:~# glmark2-es2-wayland

=====
==
    glmark2 2023.01
=====
==
    OpenGL Information
    GL_VENDOR:      Vivante Corporation
    GL_RENDERER:    Vivante GCNANOULTRA31_VIP2
    GL_VERSION:     OpenGL ES 3.1 V6.4.15.6.691815
    Surface Config: buf=32 r=8 g=8 b=8 a=8 depth=24 stencil=0 samples=0
    Surface Size:   800x600 windowed
=====
==
[build] use-vbo=false: FPS: 647 FrameTime: 1.546 ms
[build] use-vbo=true: FPS: 1025 FrameTime: 0.976 ms
[texture] texture-filter=nearest: FPS: 784 FrameTime: 1.277 ms
[texture] texture-filter=linear: FPS: 767 FrameTime: 1.305 ms
[texture] texture-filter=mipmap: FPS: 743 FrameTime: 1.346 ms
[shading] shading=gouraud: FPS: 680 FrameTime: 1.472 ms
[shading] shading=blinn-phong-inf: FPS: 642 FrameTime: 1.559 ms
[shading] shading=phong: FPS: 463 FrameTime: 2.162 ms
```

[shading] shading=cel: FPS: 369 FrameTime: 2.716 ms
 [bump] bump-render=high-poly: FPS: 442 FrameTime: 2.263 ms
 [bump] bump-render=normals: FPS: 871 FrameTime: 1.149 ms
 [bump] bump-render=height: FPS: 572 FrameTime: 1.751 ms
 [effect2d] kernel=0,1,0;1,-4,1;0,1,0;: FPS: 234 FrameTime: 4.289 ms
 [effect2d] kernel=1,1,1,1;1,1,1,1;1,1,1,1;: FPS: 97 FrameTime: 10.394 ms
 [pulsar] light=false:quads=5:texture=false: FPS: 575 FrameTime: 1.740 ms
 [desktop] blur-radius=5:effect=blur:passes=1:separable=true:windows=4: FPS: 13 FrameTime: 8.917 ms
 [desktop] effect=shadow:windows=4: FPS: 275 FrameTime: 3.639 ms
 [buffer] columns=200:interleave=false:update-dispersion=0.9:update-fraction=0.5:update-method=map: FPS: 144 FrameTime: 6.981 ms
 [buffer] columns=200:interleave=false:update-dispersion=0.9:update-fraction=0.5:update-method=subdata: FPS: 142 FrameTime: 7.086 ms
 [buffer] columns=200:interleave=true:update-dispersion=0.9:update-fraction=0.5:update-method=map: FPS: 204 FrameTime: 4.925 ms
 [ideas] speed=duration: FPS: 181 FrameTime: 5.550 ms
 [jellyfish] <default>: FPS: 191 FrameTime: 5.237 ms
 [terrain] <default>: FPS: 6 FrameTime: 186.377 ms
 [shadow] <default>: FPS: 412 FrameTime: 2.432 ms
 [refract] <default>: FPS: 46 FrameTime: 22.070 ms
 [conditionals] fragment-steps=0:vertex-steps=0: FPS: 635 FrameTime: 1.576 ms
 [conditionals] fragment-steps=5:vertex-steps=0: FPS: 162 FrameTime: 6.189 ms
 [conditionals] fragment-steps=0:vertex-steps=5: FPS: 609 FrameTime: 1.643 ms
 [function] fragment-complexity=low:fragment-steps=5: FPS: 313 FrameTime: 3.197 ms
 [function] fragment-complexity=medium:fragment-steps=5: FPS: 176 FrameTime: 5.686 ms
 [loop] fragment-loop=false:fragment-steps=5:vertex-steps=5: FPS: 306 FrameTime: 3.274 ms
 [loop] fragment-steps=5:fragment-uniform=false:vertex-steps=5: FPS: 306 FrameTime: 3.274 ms

[loop] fragment-steps=5:fragment-uniform=true:vertex-steps=5: FPS: 185 Frame
Time: 5.432 ms

=====

==

glmark2 Score: 402

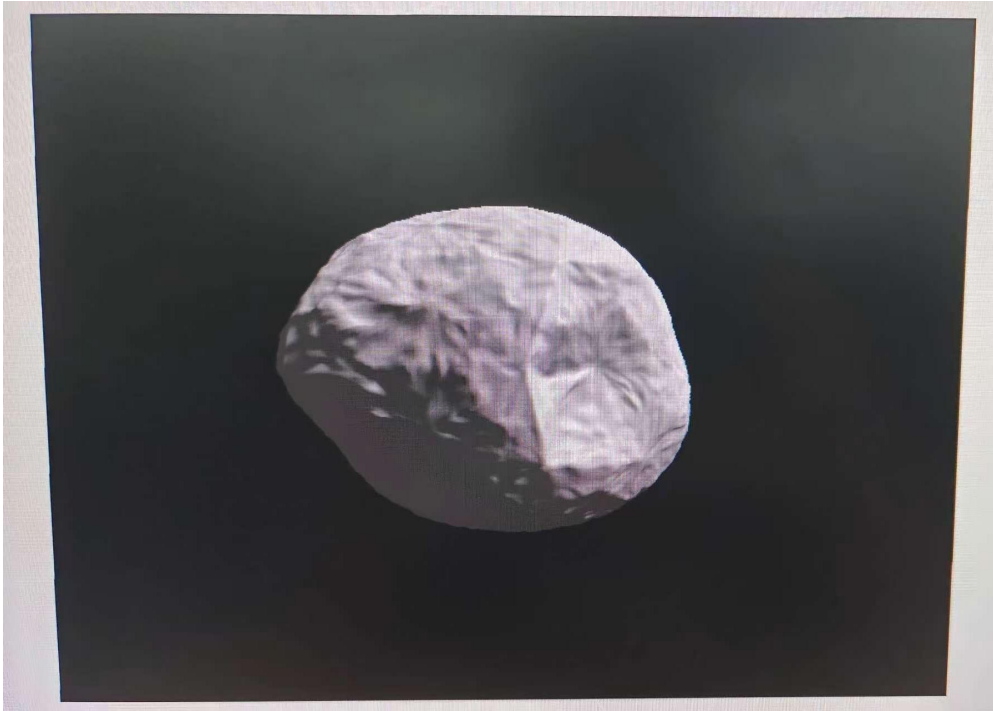


Figure 2-2. GPU Test 3D Acceleration

2.3. Memory

The MYC-LD25X core board is equipped with 2GB and 1GB DDR4 memory, in the models MYD-LD257-8E2D and MYD-LD257-8E1D, respectively.

1) To check memory information

Retrieve parameter details from the system, you can access the /proc/meminfo file.

```
root@myd-ld25x:~# cat /proc/meminfo
MemTotal:        1876544 kB
MemFree:         1388924 kB
MemAvailable:    1554624 kB
Buffers:         17496 kB
Cached:          208608 kB
SwapCached:      0 kB
Active:          54288 kB
Inactive:        314672 kB
Active(anon):    872 kB
Inactive(anon):  152608 kB
Active(file):    53416 kB
Inactive(file):  162064 kB
Unevictable:     0 kB
Mlocked:         0 kB
SwapTotal:       0 kB
SwapFree:        0 kB
Dirty:           0 kB
Writeback:       0 kB
AnonPages:       142860 kB
Mapped:          91108 kB
Shmem:           10624 kB
KReclaimable:    24340 kB
Slab:            59688 kB
```

```

SReclaimable:      24340 kB
SUnreclaim:        35348 kB
KernelStack:       2832 kB
PageTables:         3056 kB
SecPageTables:      0 kB
NFS_Unstable:       0 kB
Bounce:             0 kB
WritebackTmp:       0 kB
CommitLimit:        938272 kB
Committed_AS:       575860 kB
VmallocTotal: 133143592960 kB
VmallocUsed:         15592 kB
VmallocChunk:        0 kB
Percpu:             576 kB
HardwareCorrupted:  0 kB
AnonHugePages:      51200 kB
ShmemHugePages:      0 kB
ShmemPmdMapped:      0 kB
FileHugePages:       0 kB
FilePmdMapped:       0 kB
CmaTotal:           131072 kB
CmaFree:            122680 kB
HugePages_Total:     0
HugePages_Free:      0
HugePages_Rsvd:      0
HugePages_Surp:      0
Hugepagesize:        2048 kB
Hugetlb:             0 kB

```

- MemTotal: The total system memory amount, measured in kilobytes (KB).
- MemFree: Amount of free memory available, indicating currently unused memory.

- **MemAvailable:** Estimated amount of available memory, taking into account the kernel's memory management strategy.
- **Buffers:** Memory used for file system buffers, i.e., buffers used for file read/write operations. In this example, buffers occupy 9,500 KB.
- **Cached:** Memory used for file system cache, including cached files and directory data.
- **SwapCached:** Amount of memory that has been cached to swap space. Typically 0 indicates that swap space is not being used.
- **Active:** Active memory, the amount currently in use. This includes active anonymous memory (Active(anon)) and active file memory (Active(file)).
- **Inactive:** Inactive memory, previously used but not currently in use. This includes inactive anonymous memory (Inactive(anon)) and inactive file memory (Inactive(file)).

2) Obtain memory usage information

You can use the free command with the -m parameter to display memory usage in megabytes.

```
root@myd-ld25x:~# free -m
```

	total	used	free	shared	buff/cache	available
Mem:	1832	316	1354	10	244	1516
Swap:	0	0	0			

- **Total:**Total memory
- **used:** Used memory
- **free:** Free memory

3) Memory stress testing

By specifying the size and number of tests for memory, you can stress test the current system memory. You can use the system tool memtester for this purpose. For example, to specify a memory size of 500MB and conduct 1 test, use the command:

This command will perform a single test using 500MB of memory space.

```
root@myd-ld25x:~# memtester 500M 1
memtester version 4.6.0 (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 500MB (524288000 bytes)
got 500MB (524288000 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.
```

2.4. eMMC

eMMC is a data storage device that includes a MultiMediaCard (MMC) interface and a NAND Flash component. Its cost-effectiveness, compact size, Flash technology independence, and high data throughput make it an ideal choice for embedded products. The MYC-LD25X-8E2D-150 core board comes with a standard 8GB eMMC. This section will explain the steps and methods for viewing and managing eMMC under a Linux system. The maximum recognizable capacity of eMMC is 7.28GB.

1) Viewing eMMC Capacity

You can use the `fdisk -l` command to query eMMC partition information and capacity under Linux.

```
root@myd-ld25x:~# fdisk /dev/mmcblk1 -l
Disk /dev/mmcblk1: 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: gpt
Disk identifier: 027A60C3-52DE-4B48-A273-D14FB4650D54
```

Device	Start	End	Sectors	Size	Type
/dev/mmcblk1p1	1024	2047	1024	512K	unknown
/dev/mmcblk1p2	2048	3071	1024	512K	unknown
/dev/mmcblk1p3	3072	11263	8192	4M	unknown
/dev/mmcblk1p4	11264	19455	8192	4M	unknown
/dev/mmcblk1p5	19456	20479	1024	512K	unknown
/dev/mmcblk1p6	20480	151551	131072	64M	Linux filesystem
/dev/mmcblk1p7	151552	526335	374784	183M	Linux filesystem
/dev/mmcblk1p8	526336	6817791	6291456	3G	Linux filesystem
/dev/mmcblk1p9	6817792	15268863	8451072	4G	Linux filesystem

2) Viewing eMMC Partition Information

You can use the `df` command to query eMMC partition information, usage, mount points, and other details under Linux. Here's an example output:

```
root@myd-ld25x:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        848M  4.0K  848M   1% /dev
/dev/mmcblk1p8  2.8G  914M  1.8G  34% /
tmpfs           917M    0  917M   0% /dev/shm
tmpfs           367M   8.9M  358M   3% /run
tmpfs           4.0M    0   4.0M   0% /sys/fs/cgroup
tmpfs           917M   20K  917M   1% /tmp
/dev/mmcblk1p6   55M   20M   32M  39% /boot
/dev/mmcblk1p7  166M   27M  127M  18% /vendor
tmpfs           917M   84K  917M   1% /var/volatile
/dev/mmcblk1p9  3.8G   5.4M  3.6G   1% /usr/local
tmpfs           184M   4.0K  184M   1% /run/user/1000
tmpfs           184M    0  184M   0% /run/user/0
```

- `/dev/root`: Root filesystem mounted at the root directory.
- `Tmpfs` : Virtual memory filesystems mounted at various directories.
- `Devtmpfs` : Used by the system to create devices (`/dev`).

3) eMMC Performance Testing

Performance testing primarily measures the read and write speeds of eMMC for files under the Linux system. The following will introduce the use of the `time dd` command for conducting read and write performance tests.

● Read/Write Testing

To conduct read and write tests, use the following command:

```
root@myd-ld25x:~# dd if=/dev/zero of=test_file bs=50M count=10 conv=fsy
nc
10+0 records in
10+0 records out
524288000 bytes (524 MB, 500 MiB) copied, 8.89445 s, 58.9 MB/s
root@myd-ld25x:~# time dd if=test_file of=/dev/null bs=50M count=10 ifl
ag=direct,nonblock
10+0 records in
10+0 records out
524288000 bytes (524 MB, 500 MiB) copied, 5.61148 s, 93.4 MB/s
real    0m 5.61s
user    0m 0.00s
sys     0m 0.03s
```

4) Viewing eMMC Lifespan

The lifespan of eMMC refers to the reliable operational time or total data write/read volume under normal usage conditions for embedded multimedia card storage devices (eMMC). The lifespan of eMMC is influenced by various factors including usage frequency, number of write/read operations, environmental temperature, and the type of flash memory. As eMMC approaches its end of life, issues such as data loss or reduced device performance may occur. Therefore, it's important to consider the lifespan of the storage device when managing and backing up data.

To roughly estimate the remaining lifespan of eMMC, you can use the following command to view specific parameters:

- **Checking eMMC Lifespan**

```
root@myd-ld25x:~# mmc extcsd read /dev/mmcblk1 | 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]: 0x03
```

Explanation of these parameters:

- eMMC Life Time Estimation A: Device lifespan estimation type A, applicable for SLC user partition erase blocks.
- eMMC Life Time Estimation B: Device lifespan estimation type B, applicable for MLC boot partition erase blocks.

The values of these parameters are typically referenced in steps of 10%. For example, 0x02 would indicate that 10-20% of the device lifespan has been used.

● **The standard values for the Life Time Estimation Type B**

parameters are as shown in Table 2-1:

Table 2-1. Design Life Time Estimation Type B Standards

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

2.5. RTC

The RTC (Real-time Clock) itself is a clock used to keep track of the actual time. After connecting the RTC battery, it retains the system time and continues to keep time even when the software system is powered off. When the system is turned back on, it synchronizes the time into the software system. The following tests involve writing the system time to the external RTC, reading the time from the external RTC, and setting it as the system time, as well as testing the time retention during power loss.

- **Checking the system RTC device:**

```
root@myd-ld25x:~# ls /dev/rtc* -al
lrwxrwxrwx 1 root root      4  4 01:31 /dev/rtc -> rtc0
crw----- 1 root root 252, 0  4 01:31 /dev/rtc0
```

- **Setting the system time:**

Set the system time to "2024-09-05 15:04:00".

```
root@myd-ld25x:~# date -s "2024-09-05 15:04:00"
Thursday, September 5, 2024 15:04:00 UTC
```

If the time appears inconsistent due to automatic network time synchronization, you can disable it temporarily:

```
root@myd-ld25x:~# timedatectl set-ntp false
```

Then set the time again using the date command.

- **Writing system time to RTC:**

After setting the system time with the date command, the next step would be to write this time to the RTC device.

```
root@myd-ld25x:~# hwclock -w -f /dev/rtc0
```

```
root@myd-ld25x:~# hwclock -r -f /dev/rtc0
2024-09-05 15:06:21.359969+00:00
```

- **Retaining RTC Time during Power-off**

Install the RTC battery, ensure it is properly installed on the development board, and after completing the above steps, disconnect the power. After about 2 minutes, reconnect the power and boot up. Check the RTC time and system time.

```
myd-ld25x login: root
root@myd-ld25x:~# hwclock -r -f /dev/rtc0
2024-09-05 15:06:14.794092+00:00
```

After rebooting, if the RTC time is approximately 2 minutes later than the time set previously, it indicates that the RTC is functioning correctly. For detailed accuracy testing of the RTC, you can extend the power-off duration, such as to 24 hours, and compare the RTC time with standard time.

- **Synchronizing System Time with RTC Time:**

```
root@myd-ld25x:~# hwclock -w -f /dev/rtc0
root@myd-ld25x:~# hwclock -r -f /dev/rtc0
2024-09-05 15:08:21.359969+00:00
```

If you add the command `hwclock -s -f /dev/rtc0` to the startup script, it will ensure that the system time and RTC time are synchronized each time the system boots up.

2.6. Watchdog

The Linux kernel includes the Watchdog subsystem. This section tests the external watchdog.

1) Testing the Watchdog

- **Watchdog Application Testing:**

Before testing, you need to change the watchdog configuration to disable it. Modify the settings to the following:

```
root@myd-ld25x:~# vi /lib/systemd/system.conf.d/01-watchdog.conf
[Manager]
#RuntimeWatchdogSec=32
#ShutdownWatchdogSec=32
RebootWatchdogSec=32
```

To view the usage of the watchdog:

```
root@myd-ld25x:~# watchdog_test
Usage: wdt_driver_test <timeout> <sleep>
    timeout: value in seconds to cause wdt timeout/reset
    sleep: value in seconds to service the wdt
```

The command will start the watchdog with the specified timeout period. ``<timeout>`` sets the watchdog timeout duration, and ``<sleep>`` sets the watchdog timeout warning period. If the watchdog does not receive a system response within this period, it will force a system reboot. First, set the timeout to 10 seconds and the watchdog feed interval to 5 seconds. You can observe that the system is running continuously.

```
root@myd-ld25x:~# watchdog_test 10 5
Starting wdt_driver (time_out: 10, sleep: 5)
options = 0x8180,id = arm_smc_wdt
```

```
Trying to set time_out value=10 seconds
The actual time_out was set to 10 seconds
Now reading back -- The time_out is 10 seconds
food watchdog, count = 0
food watchdog, count = 1
.....
```

Change the timeout to 5 seconds and the watchdog feed interval to 10 seconds. In this case, if the system does not feed the watchdog within 5 seconds, it will directly restart due to the timeout.

```
root@myd-ld25x:~# watchdog_test 5 10
Starting wdt_driver (time_out: 5, sleep: 10)
options = 0x8180,id = arm_smc_wdt
Trying to set time_out value=5 seconds
The actual time_out was set to 5 seconds
Now reading back -- The time_out is 5 seconds
E/TC:1   Panic 'Watchdog' at /usr/src/debug/optee-os-stm32mp/3.19.0-stm32
mp-r2-r0/core/drivers/stm32_iwdg.c:193 <stm32_iwdg_it_handler>
E/TC:1   TEE load address @ 0x82000000
E/TC:1   Call stack:
E/TC:1   0x8200831c
E/TC:1   0x82030254
E/TC:1   0x82019eb0
E/TC:1   0x8202f198
E/TC:1   0x82013fd4
E/TC:1   0x820017dc
NOTICE:  CPU: STM32MP257DAL Rev.B
NOTICE:  Model: STMicroelectronics STM32MP257F-EV1 Evaluation Board
INFO:    Reset reason (0x2134):
INFO:    IWDG1 system reset (rst_iwdg1)
INFO:    PMIC2 version = 0x11
INFO:    PMIC2 product ID = 0x21
```

```
INFO: FCONF: Reading TB_FW firmware configuration file from: 0xe011000
INFO: FCONF: Reading firmware configuration information for: stm32mp_io
INFO: FCONF: Reading firmware configuration information for: stm32mp_fu
se
INFO: Using EMMC
INFO: Instance 2
.....
```

2.7. eeprom

To view the mounting position of the EEPROM chip, use the following command:

```
root@myd-ld25x:~# i2cdetect -y 2
   0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                -- -- -- -- -- -- --
10: UU -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- UU -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- --
50: UU -- -- -- -- -- -- -- 58 -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- -- -- -- --
```

In the output above, the EEPROM chip is located at position 0x50. The I2C address is defined by the device tree, which is not covered here. The UU notation indicates that the EEPROM chip driver is loaded.

To test the EEPROM write functionality, use the following command:

```
root@myd-ld25x:~# i2ctransfer -y -f 2 w4@0x50 0x00 0x10 0xa 0xb
```

This command writes two bytes, 0x0a and 0x0b, to the address 0x0010. For writing a single byte, use:

```
root@myd-ld25x:~# i2ctransfer -y -f 2 w3@0x50 0x00 0x10 0xa
```

Here, change w4 to w3 for writing one byte. To write more bytes, adjust accordingly. The first two bytes 0x00 and 0x10 are the 16-bit register address, which is 0x0010.

To test the EEPROM read functionality, use the following command to read the data written earlier:

```
root@myd-ld25x:~# i2ctransfer -y -f 2 w2@0x50 0x00 0x10 r1
```

0x0a

You only need to use `w2` and add the register address, then specify the number of bytes to read at the end. For example, `r1` represents reading 1 byte, so change it as follows:

```
root@myd-ld25x:~# i2ctransfer -y -f 2 w2@0x50 0x00 0x10 r2
0x0a 0x0b
```

This will read the two bytes previously written.

Another way to view the EEPROM is by using the following command to directly inspect its contents:

```
root@myd-ld25x:~# hexdump -C /sys/bus/i2c/devices/2-0050/eeprom
00000000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
00000010  0a 0b 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
*
00008000
root@myd-ld25x:~#
```

This method is more intuitive and is recommended for directly inspecting the EEPROM.

To erase the EEPROM, use the following command:

```
root@myd-ld25x:~# dd if=/dev/zero of=/sys/bus/i2c/devices/2-0050/eeprom b
s=1 count=512
```

This command can erase 512 bytes of the EEPROM's content. To erase more, adjust the `count` value.

3. Basic Peripheral Interfaces

3.1. GPIO

GPIO testing is implemented through the sysfs filesystem interface.

1) Check the current GPIO usage

You can check the current GPIO usage with the following command.

```
root@myd-ld25x:~# cat /sys/kernel/debug/gpio
gpiochip0: GPIOs 0-15, parent: platform/soc@0:pinctrl@44240000, GPIOA:
gpio-0  (PA0          )
gpio-1  (PA1          )
gpio-2  (PA2          )
gpio-3  (PA3          )
gpio-4  (PA4          )
gpio-5  (PA5          )
gpio-6  (PA6          )
gpio-7  (PA7          )
gpio-8  (PA8          )
gpio-9  (PA9          )
gpio-10 (PA10         )
gpio-11 (PA11         )
gpio-12 (PA12         )
gpio-13 (PA13         )
gpio-14 (PA14         )
gpio-15 (PA15         )

gpiochip1: GPIOs 16-31, parent: platform/soc@0:pinctrl@44240000, GPIOB:
gpio-16 (PB0          )
gpio-17 (PB1          ).....
```

In this section, we will attempt to control GPIO pins under the file system. According to the schematic, you can control pin 12 of the J13 connector (PI9), which has a pin number of $(73-65)*16+9=137$. The formula is $PIN_NO('I',9)=(0x49-0x41)*0x10+9=(73-65)*16+9=137$.

Note: 73 is the ASCII value of 'I', and 65 is the ASCII value of 'A'.

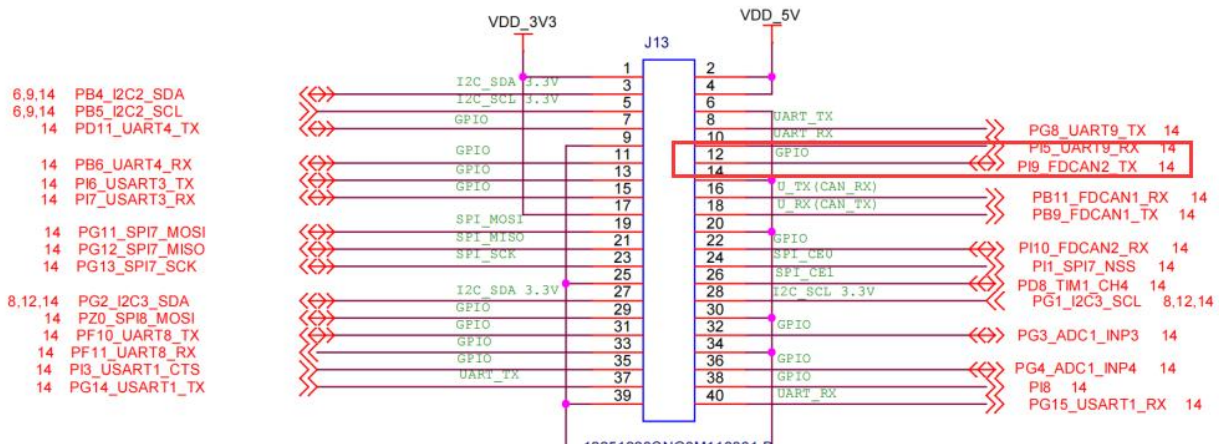


Figure 3-1. PI9 Position

Determine the actual location on the development board based on the position shown in the image, then proceed with the subsequent GPIO validation.

2) Export GPIO

```
root@myd-ld25x:/sys/class/gpio# echo 137 > /sys/class/gpio/export
```

After successful export, a directory named PI9 will be created under /sys/class/gpio/.

```
root@myd-ld25x:~# ls /sys/class/gpio/
export gpiochip0 gpiochip112 gpiochip128 gpiochip16 gpiochip32 gpiochip400
gpiochip48 gpiochip64 gpiochip80 gpiochip96 PI10 PI9 unexport
```

3) Set/Check GPIO Direction

- Set as input:

```
root@myd-ld25x:~# echo in > /sys/class/gpio/PI9/direction
```

- Set as output:

```
root@myd-ld25x:~# echo out > /sys/class/gpio/PI9/direction
```

- **Check GPIO direction:**

```
root@myd-ld25x:~# cat /sys/class/gpio/PI9/direction
out
```

If it returns "in", it indicates the GPIO is set as input; if "out", it indicates output.

4) Set/Check GPIO Value

- **Set output low**

Shown in Figure 3-1:

```
root@myd-ld25x:~# echo 0 > /sys/class/gpio/PI9/value
```

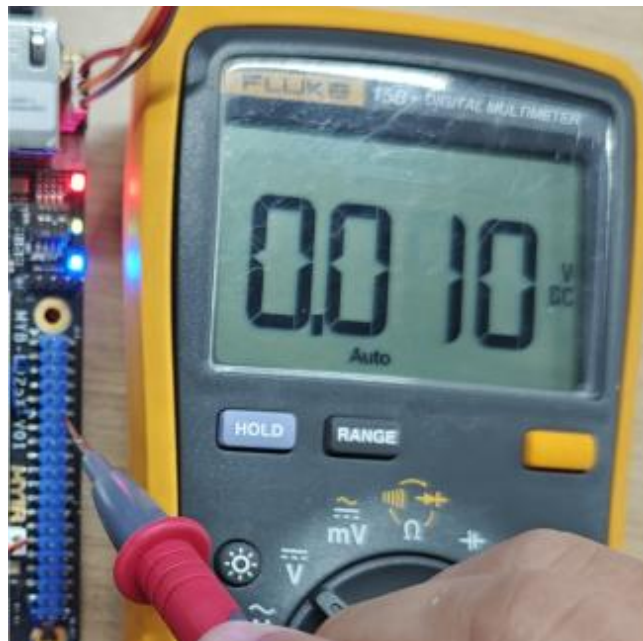


Figure 3-1. Set low output

- **Set high output**

As shown in Figure 3-2

```
root@myd-ld25x:~# echo 1 > /sys/class/gpio/PI9/value
```

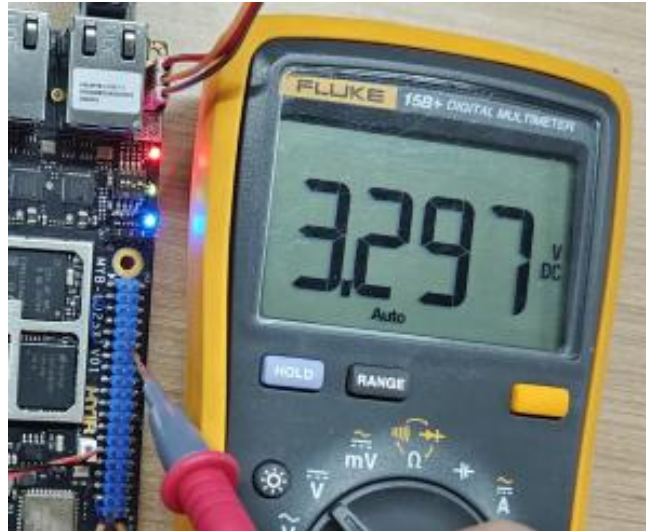


Figure 3-2. Set high output

- **Check the value of gpio**

```
root@myd-ld25x:~# cat /sys/class/gpio/PI9/value  
1
```

3.2. LED

Linux system provides an independent subsystem for easy user-space operations of LED devices, accessible through file interfaces in the `/sys/class/leds` directory. The following commands demonstrate how to test LEDs using sysfs read and write operations. These commands are universal and apply to general LED manipulation methods.

1) The LED directory `/sys/class/leds` contains:

```
root@myd-ld25x:~# ls /sys/class/leds
blue:heartbeat green:heartbeat
```

2) Testing the LED green as an example:

- Reading the state of the green LED:

The green LED is configured in the device tree to blink after the development board starts.

```
root@myd-ld25x:/sys/class/leds/green:heartbeat# cat trigger
none kbd-scrolllock kbd-numlock kbd-capslock kbd-kanalock kbd-shiftlock kbd-
-altgrlock kbd-ctrllock kbd-altlock kbd-shiftllock kbd-shiftrlock kbd-ctrlrlock kbd-
-ctrlrlock timer disk-activity disk-read disk-write ide-disk [heartbeat] cpu cpu0
cpu1 default-on panic mmc0 mmc1 mmc2 rfkill-any rfkill-none rfkill0 rfkill1
```

To turn off the green LED, write 'none' to the trigger file:

```
root@myd-ld25x:/sys/class/leds/green:heartbeat# echo none > trigger
```

The green LED will turn off.

- Turning on the LED:

```
root@myd-ld25x:/sys/class/leds/green:heartbeat# echo 1 > brightness
```

- Turning off the LED:

```
root@myd-ld25x:/sys/class/leds/green:heartbeat# echo 0 > brightness
```

- **Enabling LED trigger mode:**

After writing 'heartbeat' to the trigger file, the green LED will resume blinking.

```
root@myd-ld25x:/sys/class/leds/green:heartbeat# echo heartbeat > trigger
```

3.3. Key

The `/dev/input/eventx` (where x=1, 2, 3, ...) devices in Linux can be used to conveniently debug input devices such as mice, keyboards, and touchpads. This section primarily focuses on testing keys. You can use the `hexdump` and `evtest` commands to check if the keys are responding. The MYD-LD25X has three buttons: the Reset button, the User button, and the Wake Up button.

1) Key test

- **View corresponding input device event information**

```
root@myd-ld25x:~# 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=3
B: KEY=2 0 0 0 0
```

From the above, it can be known that the corresponding device event for gpio-keys is event0.

- **evtest key information**

Execute the following command, operate the User key, and the serial terminal will print out the following information:

```
root@myd-ld25x:~# evtest
No device specified, trying to scan all of /dev/input/event*
Available devices:
/dev/input/event0:  gpio-keys
Select the device event number [0-0]: 0
Input driver version is 1.0.1
Input device ID: bus 0x19 vendor 0x1 product 0x1 version 0x100
Input device name: "gpio-keys"
Supported events:
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 257 (BTN_1)
Properties:
Testing ... (interrupt to exit)
Event: time 1725557190.099869, type 1 (EV_KEY), code 257 (BTN_1), value 0
Event: time 1725557190.099869, ----- SYN_REPORT -----
Event: time 1725557190.280330, type 1 (EV_KEY), code 257 (BTN_1), value 1
Event: time 1725557190.280330, ----- SYN_REPORT -----
Event: time 1725557190.913019, type 1 (EV_KEY), code 257 (BTN_1), value 0
Event: time 1725557190.913019, ----- SYN_REPORT -----
Event: time 1725557191.047790, type 1 (EV_KEY), code 257 (BTN_1), value 1
Event: time 1725557191.047790, ----- SYN_REPORT -----
Event: time 1725557191.468046, type 1 (EV_KEY), code 257 (BTN_1), value 0
Event: time 1725557191.468046, ----- SYN_REPORT -----
Event: time 1725557191.619216, type 1 (EV_KEY), code 257 (BTN_1), value 1
Event: time 1725557191.619216, ----- SYN_REPORT -----
```

Each time the User is pressed, the current terminal will print out the current event code value, indicating that the key is functioning normally.

2) Test with Rest key

The MYD-LD25X can be rebooted using the Reset button.

```
root@myd-ld25x:~# NOTICE: CPU: STM32MP257DAL Rev.B
NOTICE: Model: STMicroelectronics STM32MP257F-EV1 Evaluation Board
INFO: Reset reason (0x2034):
INFO: Pad Reset from NRST
INFO: PMIC2 version = 0x11
INFO: PMIC2 product ID = 0x21
INFO: FCONF: Reading TB_FW firmware configuration file from: 0xe011000
INFO: FCONF: Reading firmware configuration information for: stm32mp_io
INFO: FCONF: Reading firmware configuration information for: stm32mp_fuse
INFO: Using EMMC
INFO: Instance 2
INFO: Boot used partition fsbl1
NOTICE: BL2: v2.8-stm32mp2-r2.0(debug):b1d29739d(b1d29739)
NOTICE: BL2: Built : 11:54:03, Jul 20 2024
INFO: BL2: Loading image id 26
INFO: Loading image id=26 at address 0xe041000
INFO: Image id=26 loaded: 0xe041000 - 0xe049650
INFO: BL2: Doing platform setup
INFO: RAM: LPDDR4 1x16Gbits 1x32bits 1200MHz
INFO:
.....
```


3.4. USB

This section verifies the feasibility of the USB Host driver through the relevant commands or hot plugging, USB HUB, and achieves the functions of reading and writing USB flash drives and USB enumeration. The test USB flash drive used in this section is a 32GB size.

1) View the print information when the usb is inserted

- View USB device information

Connect the USB flash drive to the USB Host interface (J4) of the development board. After booting up, view the kernel prompt information as follows:

```
root@myd-ld25x:~# dmesg | grep sda
[ 2.440825] sd 0:0:0:0: [sda] 61440000 512-byte logical blocks: (31.5 GB/29.3 GiB)
[ 2.441561] sd 0:0:0:0: [sda] Write Protect is off
[ 2.441577] sd 0:0:0:0: [sda] Mode Sense: 03 00 00 00
[ 2.442302] sd 0:0:0:0: [sda] No Caching mode page found
[ 2.442315] sd 0:0:0:0: [sda] Assuming drive cache: write through
[ 2.460596] sda: sda1
[ 2.461363] 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 sda1, and the maximum USB flash drive capacity recognized according to the cat /proc/partitions command is 30GB.

```
root@myd-ld25x:~# cat /proc/partitions
major minor #blocks name
179      0   7634944 mmcblk1
179      1      512 mmcblk1p1
179      2      512 mmcblk1p2
179      3     4096 mmcblk1p3
```

```
179      4      4096 mmcblk1p4
179      5      512 mmcblk1p5
179      6     65536 mmcblk1p6
179      7    187392 mmcblk1p7
179      8   3145728 mmcblk1p8
179      9   4225536 mmcblk1p9
179     32      4096 mmcblk1boot0
179     64      4096 mmcblk1boot1
  8      0  30720000 sda
  8      1  30715255 sda1
```

2) USB flash drive mount read and write

- Mount the device

Translation: By using the command `df -h`, you can see that the USB drive has been automatically mounted.

```
root@myd-ld25x:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        848M  4.0K  848M   1% /dev
/dev/mmcblk1p8  2.8G  1.4G  1.3G  53% /
tmpfs           917M    0  917M   0% /dev/shm
tmpfs           367M  9.0M  358M   3% /run
tmpfs           4.0M    0   4.0M   0% /sys/fs/cgroup
tmpfs           917M  24K  917M   1% /tmp
/dev/mmcblk1p6   55M   20M   32M  39% /boot
/dev/mmcblk1p7  166M   27M  127M  18% /vendor
/dev/mmcblk1p9  3.8G  5.4M  3.6G   1% /usr/local
tmpfs           917M  92K  917M   1% /var/volatile
tmpfs           184M  4.0K  184M   1% /run/user/1000
tmpfs           184M    0  184M   0% /run/user/0
/dev/sda1       30G   64K   30G   1% /run/media/sda1
```

- Read file

A test.txt file needs to be created on the USB flash drive in advance.

```
root@myd-ld25x:/run/media/sda1# ls test.txt
test.txt
root@myd-ld25x:/run/media/sda1# cat test.txt
hello world !!!
```

- **Write file**

```
root@myd-ld25x:~# touch test.txt
root@myd-ld25x:~# echo "MYiR" > test.txt
root@myd-ld25x:~# cp test.txt /run/media/sda1
root@myd-ld25x:~# cat /run/media/sda1/test.txt
MYiR
```

After writing the file, you need to execute the "sync" command to ensure that the data is completely written into the USB flash drive, and then you can unmount the device.

3) Unmount the USB flash drive

- **Unmount device command**

```
root@myd-ld25x:~# umount /run/media/sda1
```

or

```
root@myd-ld25x:~# umount /dev/sda1
```

3.5. USB-OTG

In this section, we will test the host function of the development board's Type-C (J3) port by using a Type-C to USB OTG cable to connect a USB flash drive. We will also test the device function of the Type-C port by connecting the development board to a PC using a Type-C to USB-A cable to simulate a USB flash drive.

1) Test host function

- **Connect the OTG cable to the development board and plug in a USB drive**

Here' s how to proceed using sda1 as an example. If multiple USB drives are connected to the development board, additional sd* device nodes like sdb may appear. After connecting a USB drive, use the following command to check the status of all partitions in the system to determine which device is the newly connected USB drive:

```
root@myd-ld25x:~# cat /proc/partitions
major minor #blocks name
179      0   7634944 mmcblk1
....
179      32     4096 mmcblk1boot0
179      64     4096 mmcblk1boot1
8        0  31166976 sda
8        1  31162880 sda1
```

After connecting the USB drive, the system will automatically mount the USB drive partitions. Use the following command to view the mounted partitions:

```
root@myd-ld25x:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        848M  4.0K  848M   1% /dev
....
```

```
tmpfs          184M  4.0K  184M   1% /run/user/1000
tmpfs          184M    0  184M   0% /run/user/0
/dev/sda1      30G   64K   30G   1% /run/media/sda1
```

If the mounted partitions do not appear after a long time, it may be due to an issue with the USB drive partitions that prevents them from being mounted. Try manually mounting the partitions using the following command:

```
root@myd-ld25x:~# mount /dev/sda1 /mnt
mount: /mnt: wrong fs type, bad option, bad superblock on /dev/sda1, missing
codepage or helper program, or other error.
        dmesg(1) may have more information after failed mount system call.
```

If you encounter this situation, it indicates a problem with the partition format of the USB drive. Please reformat the partition and try again.

- **Mount U disk and read and write test**

```
root@myd-ld25x:~# cd /run/media/sda1
root@myd-ld25x:/run/media/sda1# echo "this is a test file" > test.txt
root@myd-ld25x:/run/media/sda1# cat test.txt
this is a test file
```

2) Test the network function

First, use a USB-TypeC cable to connect the development board to the host PC.

Then configure the development board as follows:

```
/sbin/stm32_usbotg_eth_config.sh start
```

It can be seen that the OTG interface has been emulated as a network port.

```
root@myd-ld25x:~# ifconfig
end2      Link encap:Ethernet  HWaddr C6:49:B9:1A:7C:45
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
```

```

RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
Interrupt:69 Base address:0x8000

lo    Link encap:Local Loopback
      inet addr:127.0.0.1  Mask:255.0.0.0
      inet6 addr: ::1/128 Scope:Host
      UP LOOPBACK RUNNING  MTU:65536  Metric:1
      RX packets:211 errors:0 dropped:0 overruns:0 frame:0
      TX packets:211 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:14126 (13.7 KiB) TX bytes:14126 (13.7 KiB)

usb0   Link encap:Ethernet  HWaddr C6:9F:F6:68:EA:67
      inet addr:192.168.7.2  Bcast:192.168.7.255  Mask:255.255.255.0
      inet6 addr: fe80::c49f:f6ff:fe68:ea67/64 Scope:Link
      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:112 errors:0 dropped:0 overruns:0 frame:0
      TX packets:30 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:19623 (19.1 KiB) TX bytes:7140 (6.9 KiB)

```

At the same time, a virtual port will be created in the network connection management section on the PC. At this point, use the development board to ping the PC's IP address.

```

root@myd-ld25x:~# ping 192.168.7.160
PING 192.168.7.160 (192.168.7.160) 56(84) bytes of data.
64 bytes from 192.168.7.160: icmp_seq=1 ttl=128 time=1.14 ms
64 bytes from 192.168.7.160: icmp_seq=2 ttl=128 time=1.11 ms
64 bytes from 192.168.7.160: icmp_seq=3 ttl=128 time=0.616 ms

```

```
^C
--- 192.168.7.160 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 0.616/0.954/1.135/0.239 ms
```

The network is available.

Note: This feature requires the RNDIS driver on the computer. Most modern computer systems come with the RNDIS driver pre-installed. If it's not installed, please do so manually.

3.6. 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 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 multi-functional memory card with large capacity, high performance and security. Micro SD card generally has 9 pins on the back, contains 4 data lines, and supports 1bit/4bit data transmission width.

This section explains the steps and methods to view and operate the SD card under Linux system.

1) Check the SD card capacity

The fdisk -l command can be used to query the SD card partition information and capacity.

```
root@myd-ld25x:~# fdisk -l
.....
Disk /dev/mmcblk0: 29.12 GiB, 31266439168 bytes, 61067264 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: 0x5f667c2e

Device            Boot  Start      End  Sectors  Size Id Type
/dev/mmcblk0p1                2048   526335   524288  256M  6 FAT16
/dev/mmcblk0p2           526336 9765624 9239289   4.4G 83 Linux
```


- /dev/mmcblk0: Device node corresponding to SD card

The SD card size in this test is 32GB, and the maximum SD card capacity identified is 29.1GB by viewing the device information.

```
root@myd-ld25x:~# cat /proc/partitions
major minor  #blocks  name

179        0    7634944 mmcblk1
179        1         512 mmcblk1p1
179        2         512 mmcblk1p2
179        3        4096 mmcblk1p3
179        4        4096 mmcblk1p4
179        5         512 mmcblk1p5
179        6       65536 mmcblk1p6
179        7      187392 mmcblk1p7
179        8     3145728 mmcblk1p8
179        9     4225536 mmcblk1p9
179       32        4096 mmcblk1boot0
179       64        4096 mmcblk1boot1
179       96    30533632 mmcblk0
179       97     262144 mmcblk0p1
179       98    4619644 mmcblk0p2
```

2) Check the SD card partition information

Through the df-h command, you can query the SD card partition information, usage, mount directory and other information.

```
root@myd-ld25x:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        848M  4.0K  848M   1% /dev
/dev/mmcblk1p8  2.8G  1.4G  1.3G  53% /
tmpfs           917M    0  917M   0% /dev/shm
tmpfs           367M  9.0M  358M   3% /run
```

```
tmpfs          4.0M      0  4.0M    0% /sys/fs/cgroup
tmpfs          917M    28K  917M    1% /tmp
/dev/mmcblk1p6  55M     20M   32M   39% /boot
/dev/mmcblk1p7 166M    27M  127M   18% /vendor
tmpfs          917M    84K  917M    1% /var/volatile
/dev/mmcblk1p9  3.8G    5.4M   3.6G    1% /usr/local
tmpfs          184M    4.0K  184M    1% /run/user/1000
tmpfs          184M      0  184M    0% /run/user/0
/dev/mmcblk0p1  4.7G    21M   4.7G    1% /run/media/mmcblk0p1
/dev/mmcblk0p2  4.3G   3.8G  332M   93% /run/media/mmcblk0p2
```

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

3) Performance test of SD card

Performance testing mainly assesses the read and write speeds of the SD card in a Linux system using the dd command. Here, we will mount the partition of the SD card to be tested. As an example, we will use the partition /dev/mmcblk0p2 and mount it to the directory /run/media/mmcblk0p2.

● Read and Write tests

Let's use the dd command to test the speed of reading and writing.

```
root@myd-ld25x:/run/media/mmcblk0p2# time dd if=/dev/zero of=test_file_w
bs=10M count=50 conv=fsync
50+0 records in
50+0 records out
524288000 bytes (524 MB, 500 MiB) copied, 44.7839 s, 11.7 MB/s
real    0m 44.79s
user    0m 0.00s
```

```
sys    0m 4.36s
root@myd-ld25x:/run/media/mmcblk0p2# time dd if=test_file_w of=/dev/null
bs=10M count=50 iflag=direct,nonblock
50+0 records in
50+0 records out
524288000 bytes (524 MB, 500 MiB) copied, 24.5873 s, 21.3 MB/s
real   0m 24.59s
user   0m 0.00s
sys    0m 0.02s
root@myd-ld25x:/run/media/mmcblk0p2#
```

According to the results, the writing speed of this SD card is 11.7 MB/s, and the reading speed is 21.3 MB/s.

3.7. LVDS Display

This section will cover testing the display. The MYD-LD25X features a single LVDS display. The LVDS display is adapted to the MY-LVDS070C screen. The screen is connected via a 40-pin flexible ribbon cable to the J6 connector on the development board, as shown in Figure 3-3.

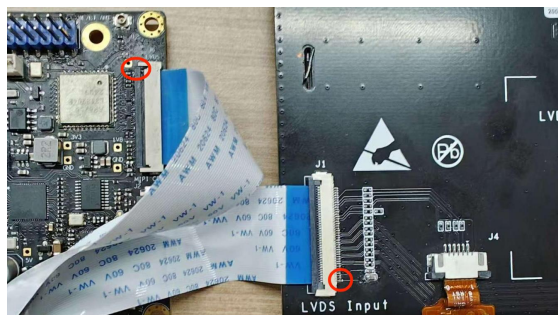


Figure 3-3. J6 Connector and LVDS Screen Connection

Note: The end of the J6 connector with the triangle symbol needs to be connected to the end of the screen connector with the triangle symbol.

To check the current display resolution, for example, the default resolution might be 1024x600.

```

root@myd-ld25x:~# cat /sys/kernel/debug/dri/0/state
plane[37]: plane-0
    crtc=crtc-0
    fb=62
    allocated by = weston
    refcount=2
    format=XR24 little-endian (0x34325258)
    modifier=0x0
    size=1024x600
    layers:
        size[0]=1024x600
        pitch[0]=4096
        offset[0]=0
        obj[0]:
            name=0
            refcount=2
            start=00100bb8
            size=2457600
            imported=yes
            dma_addr=0x00000000fcc3a000
            vaddr=0000000000000000
    crtc-pos=1024x600+0+0
    src-pos=1024.000000x600.000000+0.000000+0.000000
    rotation=1
    normalized-zpos=0
    color-encoding=ITU-R BT.601 YCbCr
    color-range=YCbCr limited range
    user_updates=1fps
plane[45]: plane-1
    crtc=(null)
    fb=0
    crtc-pos=0x0+0+0

```

```

src-pos=0.000000x0.000000+0.000000+0.000000
rotation=1
normalized-zpos=1
color-encoding=ITU-R BT.601 YCbCr
color-range=YCbCr limited range
user_updates=0fps
plane[52]: plane-2
crtc=(null)
fb=0
crtc-pos=0x0+0+0
src-pos=0.000000x0.000000+0.000000+0.000000
rotation=1
normalized-zpos=2
color-encoding=ITU-R BT.601 YCbCr
color-range=YCbCr limited range
user_updates=0fps
crtc[44]: 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=2
encoder_mask=2
mode: "1024x600": 62 52000 1024 1046 1162 1324 600 602 605 629 0
x48 0x0
transfer_error=0
fifo_underrun_error=0

```

```
fifo_underrun_warning=0
fifo_underrun_rotation=0
fifo_underrun_threshold=128
connector[32]: HDMI-A-1
    crtc=(null)
    self_refresh_aware=0
    max_requested_bpc=0
connector[34]: LVDS-1
    crtc=crtc-0
    self_refresh_aware=0
    max_requested_bpc=0
```

3.8. Touch Panel

The MYD-LD25X series development boards support capacitive touch. Mill Electronics offers a 7-inch touchscreen LCD accessory, as listed in Table 1-1. You can purchase accessories according to your actual needs. Capacitive screens are quite sensitive and rarely encounter issues. If the touchscreen responds to clicks, it is functioning properly. Additionally, capacitive screens do not require calibration. Due to the principle of capacitive touch technology, the screen can accurately detect the position of a finger touching it and offers high sensitivity. If you experience issues where the screen does not accurately select software elements, it usually indicates a problem with the screen. Below, we will perform a simple test of the touchscreen functionality using the `evtest` command.

1) Touch screen connection

Connect the MY-LVDS070C touchscreen LCD display to the J6 connector on the development board as described in section 3.7.

2) Touch screen test

Execute the command `evtest` in the terminal to enter the test interface. Select the touchscreen device for testing and choose option '1'. Press Enter to start the test.

```
root@myd-ld25x:~# 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]:
```

- **Click the touch screen, the terminal will print the corresponding information**

```
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 352

Min 0

Max 65535

Event code 1 (ABS_Y)

Value 544

Min 0

Max 65535

Event code 47 (ABS_MT_SLOT)

Value 0

Min 0

Max 4

Event code 53 (ABS_MT_POSITION_X)

Value 0

Min 0

Max 65535

Event code 54 (ABS_MT_POSITION_Y)

Value 0

Min 0

Max 65535

Event code 57 (ABS_MT_TRACKING_ID)

Value 0

Min 0

Max 65535

Properties:


```

Property type 1 (INPUT_PROP_DIRECT)
Testing ... (interrupt to exit)
Event: time 1677837532.186263, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID), value 68
Event: time 1677837532.186263, type 3 (EV_ABS), code 53 (ABS_MT_POSITION_X), value 380
Event: time 1677837532.186263, type 3 (EV_ABS), code 54 (ABS_MT_POSITION_Y), value 628
Event: time 1677837532.186263, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 1
Event: time 1677837532.186263, type 3 (EV_ABS), code 0 (ABS_X), value 380
Event: time 1677837532.186263, type 3 (EV_ABS), code 1 (ABS_Y), value 628
Event: time 1677837532.186263, ----- SYN_REPORT -----
Event: time 1677837532.205052, type 3 (EV_ABS), code 53 (ABS_MT_POSITION_X), value 378
Event: time 1677837532.205052, type 3 (EV_ABS), code 54 (ABS_MT_POSITION_Y), value 630
Event: time 1677837532.205052, type 3 (EV_ABS), code 0 (ABS_X), value 378
Event: time 1677837532.205052, type 3 (EV_ABS), code 1 (ABS_Y), value 630
Event: time 1677837532.205052, ----- SYN_REPORT -----
Event: time 1677837532.225037, type 3 (EV_ABS), code 53 (ABS_MT_POSITION_X), value 381
Event: time 1677837532.225037, type 3 (EV_ABS), code 54 (ABS_MT_POSITION_Y), value 628
Event: time 1677837532.225037, type 3 (EV_ABS), code 0 (ABS_X), value 381
Event: time 1677837532.225037, type 3 (EV_ABS), code 1 (ABS_Y), value 628
Event: time 1677837532.225037, ----- SYN_REPORT -----
Event: time 1677837532.246438, type 3 (EV_ABS), code 53 (ABS_MT_POSITION_X), value 382
Event: time 1677837532.246438, type 3 (EV_ABS), code 54 (ABS_MT_POSITION_Y), value 631
Event: time 1677837532.246438, type 3 (EV_ABS), code 0 (ABS_X), value 382

```

```
Event: time 1677837532.246438, type 3 (EV_ABS), code 1 (ABS_Y), value 631
Event: time 1677837532.246438, ----- SYN_REPORT -----
Event: time 1677837532.264305, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID), value -1
Event: time 1677837532.264305, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 0
```

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: Synchronization events
- EV_KEY: Key event; for example, BTN_TOUCH indicates a touch key
- EV_ABS: absolute coordinates, such as those reported by the touch screen
- BTN_TOUCH: Touch the key
- ABS_MT_TRACKING_ID: indicates the beginning of collecting information, and the following 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, on the other hand, are carried in ABS_MT and sent in a certain order, such as:

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

4. RS232、RS485、CAN

To use RS232, RS485, and CAN functionalities with the MYD-LD25X development board, you need to connect an external Raspberry Pi (MY-WIREDCOM) module to the J13 connector on the development board, as shown in Figure 4-1.

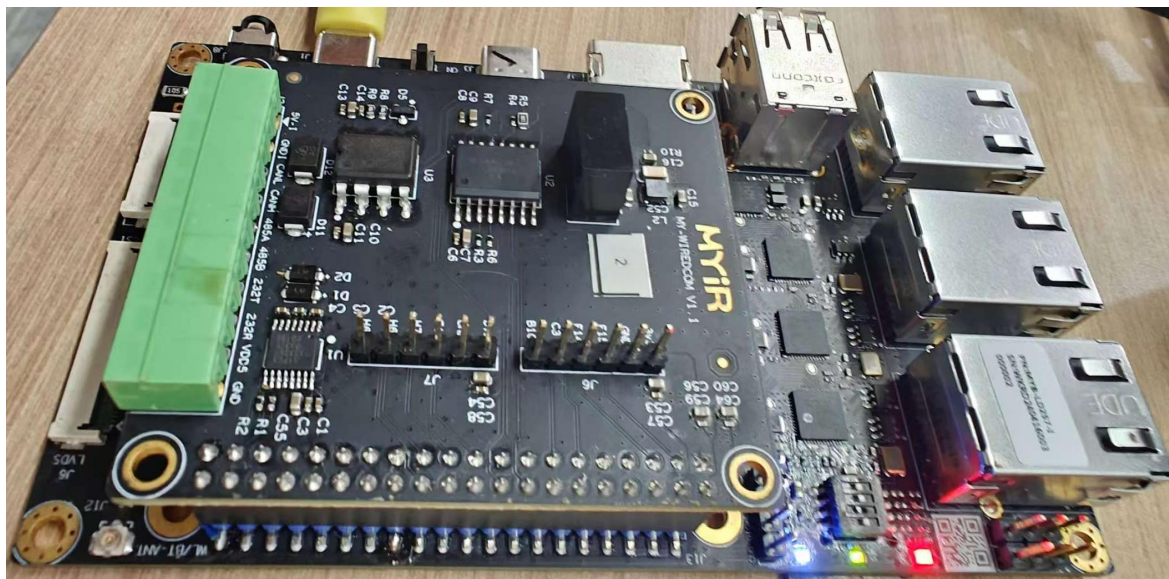


Figure 4-1. MY-WIREDCOM Connected to the Development Board

The pin definitions of the Raspberry Pi and the corresponding pinout of the 40-pin connector on the development board are shown in the following diagram:

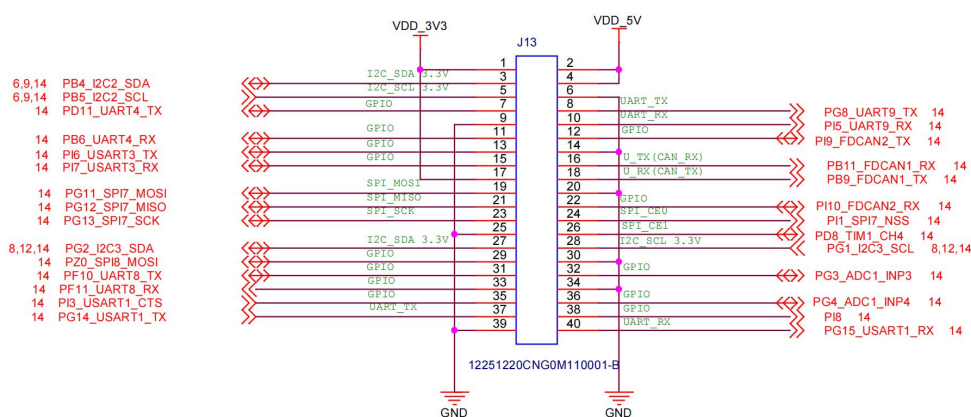


Figure 4-2. 40-Pin Connector Pinout of the Development Board

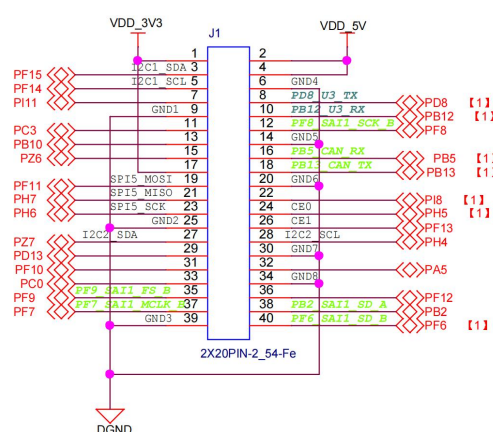


Figure 4-3. Raspberry Pi Pin Definitions

If the Raspberry Pi module is not connected, you can directly browse Figure 4-2 or refer to the pin list to understand the definitions of each of the 40 pins.

The pin correspondence between the Raspberry Pi module, the development board's 40-pin connector, and the pin numbers is shown in the diagram below. You can refer to Figure 4-2 and Figure 4-3 for cross-referencing.

Table 4-1. Pin Correspondence Table-1

	RS232-TXD	RS232-RXD	RS485-A	RS485-B
Raspberry Pi Module	PF7	PF6	PD8	PB12
Development Board 40-Pin Connector	PG14_USART_TX	PG15_USART1_RX	PG8_UART9_TX	PI5_UART9_RX
Pin Number	37	40	8	10

Table 4-2. Pin Correspondence Table-2

	RS485-RTS	CANH	CANL
Raspberry Pi Module	PI8	PB13	PB5
Development Board 40-Pin Connector	PI10_FDCAN2RX	PB9_FDCAN1_TX	PB11_FDCAN1_RX
Pin Number	22	18	16

4.1. RS232

This example demonstrates how to use RS232 for sending and receiving data. The RS232 interface is available on pins 7, 8, and 10 of the J2 connector on the MY-WIREDCOM module, corresponding to UART1, with the device node as ttySTM1. The following will use a USB-RS232 isolator converter to connect to a PC for RS232 functionality testing. The hardware interface configuration is shown in the table below:

Table 4-1. RS232 Interface Configuration

	MYD-LD25X
Interface	RS232
Device Node	ttySTM1
Test Software	uart_test

Connect the RS232 TXD and RXD to the isolated converter's RXD and TXD in a cross configuration (i.e., RXD to TXD and TXD to RXD). Also, connect the GND pin of the MY-WIREDCOM module J2 connector (pin 10) to the GND of the isolated converter, as shown in Figure 4-2.

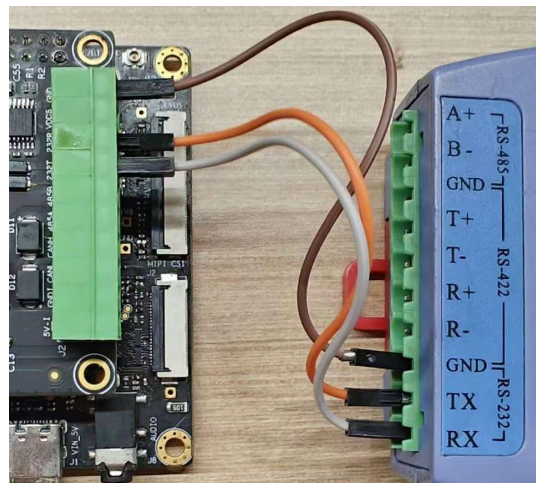


Figure 4-2. RS232 Wiring Diagram

● RS232 Receive and Transmit on Development Board

On the PC, open SSCOM or similar software, configure the baud rate to 115200, disable RTS and DTR, and ensure the serial port is open.

On the development board, execute the following command to send data via RS232:

```
root@myd-ld25x:/# uart_test -d /dev/ttySTM1 -b 115200 -m 1
/dev/ttySTM1 SEND: 1234567890abcdefg
/dev/ttySTM1 SEND: 1234567890abcdefg
/dev/ttySTM1 SEND: 1234567890abcdefg
/dev/ttySTM1 SEND: 1234567890abcdefg
/dev/ttySTM1 SEND: 1234567890abcdefg
```

At this point, you should see that the data has been received on the PC's SSCOM:

```
[18:23:13.172]Received ←◆1234567890abcdefg
[18:23:13.272]Received ←◆1234567890abcdefg
[18:23:13.372]Received ←◆1234567890abcdefg
[18:23:13.472]Received ←◆1234567890abcdefg
[18:23:13.573]Received ←◆1234567890abcdefg
```

On the development board, execute the following command to have RS232 wait for data reception:

```
root@myd-ld25x:/# uart_test -d /dev/ttySTM1 -b 115200 -m 0
```

Then, on the PC, input any data and send it:

```
[18:24:02.718]Send→◆ABCDEFGH
[18:24:03.080]Send→◆ABCDEFGH
[18:24:03.342]Send→◆ABCDEFGH
```

You should see that the data has been received on the development board:

```
root@myd-ld25x:/# uart_test -d /dev/ttySTM1 -b 115200 -m 0
/dev/ttySTM1 RECV[7]: ABCDEFG
/dev/ttySTM1 RECV[7]: ABCDEFG
/dev/ttySTM1 RECV[7]: ABCDEFG
```

4.2. RS485

This example demonstrates how to use RS485 for sending and receiving data. The RS485 interface is available on pins 5 and 6 of the J2 connector on the MY-WIREDCOM module, corresponding to UART9, with the device node as `ttystm9`. The following will use a USB-RS485 isolator converter to connect to a PC for RS485 functionality testing. The hardware interface configuration is shown in the table below:

Table 4-2. RS485 Interface Configuration

	MYD-LD25X
Interface	RS485
Device Node	ttystm9
Test Software	uart_test

Before using RS485, you need to configure the GPIO pin to control the direction. According to the schematic, the RS485 nRTS GPIO pin is PI10. The pin number calculation is as follows:

```
root@myd-ld25x:~# echo 138 > /sys/class/gpio/export
root@myd-ld25x:~# echo out > /sys/class/gpio/PI10/direction
```

Connect the A and B pins of the J2 connector on the MY-WIREDCOM module to the A and B pins of the isolator converter, as shown in Figure 4-3.

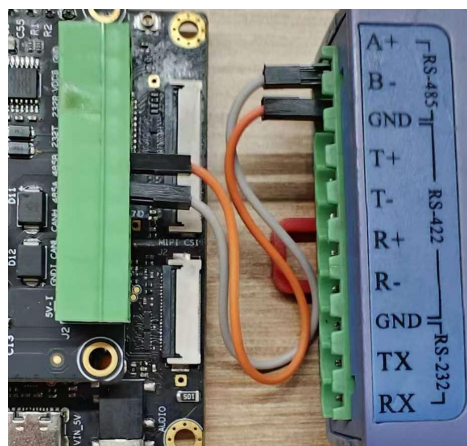


Figure 4-3. RS485 Wiring Diagram

● RS485 Reception on Development Board

First, set the RS485 nRTS GPIO pin to 0:

```
root@myd-ld25x:~# echo 0 > /sys/class/gpio/PI10/value
```

Then, open SSCOM or similar software on the PC, configure the baud rate to 115200, disable RTS and DTR, and ensure the serial port is open.

On the development board, execute the following command to make RS485 wait for data reception:

```
root@myd-ld25x:/# uart_test -d /dev/ttySTM9 -b 115200 -m 0
```

At this point, input any data on the PC's SSCOM and send it.

```
[18:24:02.718]Send→◇ABCDEFGF
[18:24:03.080]Send→◇ABCDEFGF
[18:24:03.342]Send→◇ABCDEFGF
[18:24:10.117]Send→◇ABCDEFGF
[18:24:10.454]Send→◇ABCDEFGF
```

You can see that the development board is able to receive data:

```
root@myd-ld25x:~# uart_test -d /dev/ttySTM9 -b 115200 -m 0
/dev/ttySTM9 RECV[7]: ABCDEFG
/dev/ttySTM9 RECV[7]: ABCDEFG
/dev/ttySTM9 RECV[7]: ABCDEFG
/dev/ttySTM9 RECV[7]: ABCDEFG
/dev/ttySTM9 RECV[7]: ABCDEFG
```

● For RS485 transmission

Set the RS485 nRTS GPIO pin to 1 and execute the following commands on the development board to send data:

```
root@myd-ld25x:~# echo 1 > /sys/class/gpio/PI10/value
```



```
root@myd-ld25x:~# uart_test -d /dev/ttySTM9 -b 115200 -m 1
/dev/ttySTM9 SEND: 1234567890abcdefg
/dev/ttySTM9 SEND: 1234567890abcdefg
/dev/ttySTM9 SEND: 1234567890abcdefg
/dev/ttySTM9 SEND: 1234567890abcdefg
/dev/ttySTM9 SEND: 1234567890abcdefg
/dev/ttySTM9 SEND: 1234567890abcdefg
/dev/ttySTM9 SEND: 1234567890abcdefg
/dev/ttySTM9 SEND: 1234567890abcdefg
```

At this point, you should see that the data has been received on the PC's SSCOM:

```
[10:17:59.272]Received ←◆1234567890abcdefg
[10:17:59.372]Received ←◆1234567890abcdefg
[10:17:59.472]Received ←◆1234567890abcdefg
[10:17:59.572]Received ←◆1234567890abcdefg
[10:17:59.672]Received ←◆1234567890abcdefg
[10:17:59.772]Received ←◆1234567890abcdefg
[10:17:59.872]Received ←◆1234567890abcdefg
[10:17:59.973]Received ←◆1234567890abcdefg
```

4.3. CAN

In this section, we will use the common Linux commands `cansend` and `candump` to test CAN communication. The following describes the process of testing with a standard CAN interface on similar development boards.

Connect the MYD-LD25X development board to the MY-WIREDCOM module, with the CAN interface connected to pins 3 and 4 on the J2 connector of the MY-WIREDCOM module. Connect the CANL and CANH interfaces of the MYD-LD25X to a development board or other device with CAN functionality. This section will use the same type of development board as an example, as shown in Figure 4-4:

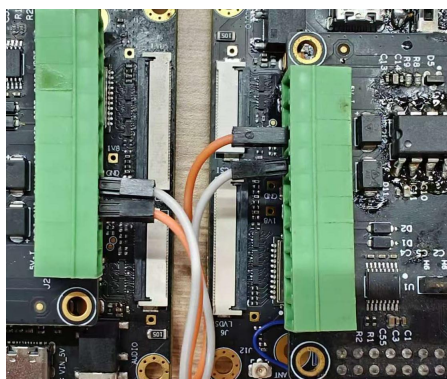


Figure 4-4. CAN Wiring Diagram

1) Initialize CAN network interface

- Set CAN baud rate

Before using CAN, you need to set the baud rate and bring up the CAN network interface. Use the following commands to set the arbitration baud rate to 50Kbps and the data baud rate to 50Kbps for both boards:

```
root@myd-ld25x:~# ifconfig can0 down
root@myd-ld25x:~# ip link set can0 type can bitrate 50000
root@myd-ld25x:~# ifconfig can0 up
```

2) Send and receive data

Set one of the development boards' CAN0 interfaces to receive mode and use the candump command to receive data:

```
root@myd-ld25x:~# candump can0 &
```

On the other development board, set the CAN0 interface to send mode and use the cansend command to send data:

```
root@myd-ld25x:~# cansend can0 123#0811223344556677
root@myd-ld25x:~# cansend can0 123#0811223344556677
root@myd-ld25x:~# cansend can0 123#0811223344556677
root@myd-ld25x:~# cansend can0 123#0811223344556677
```

At this point, the first development board should receive the data:

```
root@myd-ld25x:~#
can0 123 [8] 08 11 22 33 44 55 66 77
can0 123 [8] 08 11 22 33 44 55 66 77
can0 123 [8] 08 11 22 33 44 55 66 77
can0 123 [8] 08 11 22 33 44 55 66 77
```

3) Configure can0 interface

After CAN data transmission and reception, display details and statistics of the CAN device. The "clock" value represents the CAN clock, "drop" indicates packet drops, "overrun" signifies overflow, and "error" refers to bus errors.

```
root@myd-ld25x:~# ip -details -statistics link show can0
4: can0: <NOARP,UP,LOWER_UP,ECHO> mtu 16 qdisc pfifo_fast state UP mode
DEFAULT group default qlen 10
    link/can  promiscuity 0  allmulti 0 minmtu 0 maxmtu 0
    can state  ERROR-ACTIVE (berr-counter tx 0 rx 0) restart-ms 0
    bitrate 50000 sample-point 0.875
    tq 100 prop-seg 87 phase-seg1 87 phase-seg2 25 sjw 1 brp 10
    m_can: tseg1 2..256 tseg2 2..128 sjw 1..128 brp 1..512 brp_inc 1
    m_can: dtseg1 1..32 dtseg2 1..16 dsjw 1..16 dbrp 1..32 dbrp_inc 1
```

```

clock 100000000
re-started bus-errors arbit-lost error-warn error-pass bus-off
0          0          0          0          0          0          num
txqueues 1 numrxqueues 1 gso_max_size 65536 gso_max_segs 65535 tso_max
_size 65536 tso_max_segs 65535 gro_max_size 65536 parentbus platform pare
ntdev 402d0000.can
RX:  bytes packets errors dropped missed  mcast
     32         4         0         0         0         0
TX:  bytes packets errors dropped carrier collsns
     0         0         0         0         0         0

```

5. Network

This chapter is designed to test and evaluate the connection, configuration, and use of network devices such as Ethernet, Wi-Fi, and (optionally) mobile networks.

5.1. Network Devices

The MYD-LD25X development board includes two Gigabit Ethernet interfaces (end1, end2), an Ethernet Switch interface, and a Wi-Fi/Bluetooth module. Below, we will introduce the configurations for these three types of network devices.

5.1.1. Ethernet

There are many tools for Linux network configuration, common net-tools, iproute2, systemd-networkd, network manager and connman, etc. which can be customized according to the actual needs of the system to choose. The MYD-LD25X development board uses systemd-networkd to manage the network.

1) SSH to the virtual IP

The MYD-LD25X development board is configured with static IPs for end1: 192.168.0.10 and end2: 192.168.1.10 (a virtual IP will be set once a network cable is connected). This setup allows users to debug the development board without needing a serial port, as they can use SSH tools to log into the board. To do this, first connect the development board to a computer with an Ethernet cable, manually set the computer's IP to an address in the same subnet as the board, such as 192.168.0.2.

At this point, you can use SecureCRT or other tools to select SSH and log into the development board, as shown in the figure below:

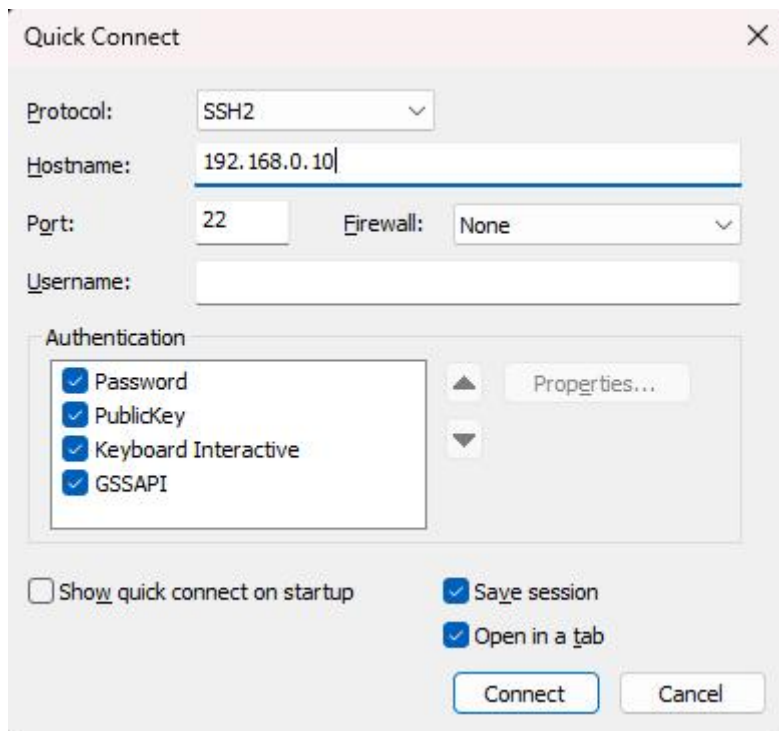


Figure 5-1. SSH into the board

A successful login will look like the following:

WARNING! The remote SSH server rejected X11 forwarding request.
root@myd-ld25x:~#

2) The terminal gets the IP

If the user logs in directly to the development board with the serial port, he can view the network port information of the development board with the command `ifconfig`:

```
root@myd-ld25x:~# ifconfig
end1      Link encap:Ethernet  HWaddr CA:10:F7:27:65:A0
          inet addr:192.168.0.10  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::c810:f7ff:fe27:65a0/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7 errors:0 dropped:0 overruns:0 frame:0
          TX packets:56 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

```

RX bytes:1466 (1.4 KiB) TX bytes:9032 (8.8 KiB)
Interrupt:66

end2    Link encap:Ethernet HWaddr CA:A1:7D:DC:A6:00
UP BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
Interrupt:68

lo      Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:211 errors:0 dropped:0 overruns:0 frame:0
TX packets:211 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:14126 (13.7 KiB) TX bytes:14126 (13.7 KiB)

```

After connecting the Ethernet cable to the router, the network port will automatically obtain an IP address. However, the obtained IP address may not be visible using the `ifconfig` command. You can check it using the ``ip route`` command, as shown below:

```

root@myd-ld25x:~# ip route
default via 192.168.30.1 dev end1 proto dhcp src 192.168.30.203 metric 1024
192.168.0.0/24 dev end1 proto kernel scope link src 192.168.0.10
192.168.30.0/24 dev end1 proto kernel scope link src 192.168.30.203 metric 1024
192.168.30.1 dev end1 proto dhcp scope link src 192.168.30.203 metric 1024

```

5.1.2. Ethernet Switch

This section mainly introduces the use of the Ethernet Switch interface (J11). To utilize this feature, you need to switch the device tree. During startup, there will be two 2-second intervals for selection: the first is to enter U-Boot, and the second is to choose the device tree, as shown below:

```
.....
Error: eth2@482d0000 address not set.
No ethernet found.

MMC: no card present
MMC: no card present
No EFI system partition

Error: eth2@482d0000 address not set.

Error: eth1@482c0000 address not set.
Hit any key to stop autoboot:  0
Boot over mmc1!
switch to partitions #0, OK
mmc1(part 0) is current device
Scanning mmc 1:6...
Found /mmc1_extlinux/myb-stm32mp257x-2GB_extlinux.conf
Retrieving file: /mmc1_extlinux/myb-stm32mp257x-2GB_extlinux.conf
Retrieving file: /splash_landscape.bmp
Select the boot mode
1:   OpenSTLinux
2:   myb-stm32mp257x-2GB
3:   myb-stm32mp257x-2GB-ethswitch
Enter choice: 3
```

Press Enter at this point to pause and then select option 3 to continue booting. This will enable the Ethernet Switch functionality, allowing the development board

to function as a switch. After booting, you can check the information by running `ifconfig` as follows:

```
root@myd-ld25x:~# ifconfig
br0      Link encap:Ethernet  HWaddr F2:C0:A2:28:D4:FE
        inet6 addr: fe80::f0c0:a2ff:fe28:d4fe/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:24 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:0 (0.0 B)  TX bytes:2728 (2.6 KiB)

end1     Link encap:Ethernet  HWaddr 38:D5:47:00:29:62
        inet addr:192.168.0.10  Bcast:192.168.0.255  Mask:255.255.255.0
        inet6 addr: fe80::3ad5:47ff:fe00:2962/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:238 errors:0 dropped:0 overruns:0 frame:0
        TX packets:388 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:34099 (33.2 KiB)  TX bytes:60143 (58.7 KiB)
        Interrupt:66 Base address:0xc000

end2     Link encap:Ethernet  HWaddr A2:18:33:61:FA:29
        UP BROADCAST MULTICAST  MTU:1500  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
        Interrupt:68 Base address:0x8000

lo       Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        inet6 addr: ::1/128 Scope:Host
```

```

UP LOOPBACK RUNNING  MTU:65536  Metric:1
RX packets:211 errors:0 dropped:0 overruns:0 frame:0
TX packets:211 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:14140 (13.8 KiB)  TX bytes:14140 (13.8 KiB)

sw0ep    Link encap:Ethernet  HWaddr 38:D5:47:00:29:62
          inet addr:192.168.0.10  Bcast:0.0.0.0  Mask:255.255.255.255
          inet6 addr: fe80::3ad5:47ff:fe00:2962/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:247 errors:0 dropped:0 overruns:0 frame:0
          TX packets:388 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:38084 (37.1 KiB)  TX bytes:61705 (60.2 KiB)

sw0p1    Link encap:Ethernet  HWaddr 72:B8:98:FB:49:00
          inet6 addr: fe80::70b8:98ff:fefb:4900/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:388 errors:0 dropped:12 overruns:0 frame:0
          TX packets:247 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:61705 (60.2 KiB)  TX bytes:38084 (37.1 KiB)

sw0p2    Link encap:Ethernet  HWaddr 72:B8:98:FB:49:01
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:17 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:432 (432.0 B)

sw0p3    Link encap:Ethernet  HWaddr 72:B8:98:FB:49:02
          UP BROADCAST MULTICAST  MTU:1500  Metric:1

```

```
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:11 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
```

At this point, the J9 and J11 Ethernet ports become switch ports labeled as `sw0ep`. When connecting two different devices to the J9 and J11 ports, these devices will be able to communicate with each other over the network.

5.1.3. 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, and some devices also support STA and AP mode to work at the same time. STA mode allows the device to connect to external Wi-Fi hotspots, and AP mode turns the device into a Wi-Fi hotspot for other devices to connect to. The STA mode and AP mode can be used together.

1) Connect WIFI in STA mode

1.1) Connect to Wi-Fi using a shell script

We will organize the manual Wi-Fi configuration process into a script `wifi_on_sta` in the `/etc/myir_test/` directory for user reference. Let's try to connect to the nearby Wi-Fi hotspot "MYIR_5G", which is a Wi-Fi hotspot using WPA2 encryption. The password is `myir@1601`.

```
root@myd-ld25x:/etc/myir_test# ./wifi_on_sta -ssid MYIR_5G -passwd myir@1601
SSID:MYIR_5G PASSWD:myir@1601 DRIVER:nl80211
killall: udhcpc: no process killed
killall: wpa_supplicant: no process killed
killall: hostapd: no process killed
```

```
killall: udhcpd: no process killed
find phy0 enable it
udhcpd: started, v1.36.1
Dropped protocol specifier '.udhcpd' from 'wlan0.udhcpd'. Using 'wlan0' (ifindex=5).
udhcpd: broadcasting discover
udhcpd: broadcasting select for 192.168.20.59, server 192.168.20.92
udhcpd: lease of 192.168.20.59 obtained from 192.168.20.92, lease time 3599
/etc/udhcpd.d/50default: Adding DNS 192.168.20.92
```

1.2)Manually connect to WiFi

Using the above script will generate the /etc/wpa_supplicant.conf configuration file according to the input ssid and passwd parameters, and connect to the specified WiFi hotspot, configure the IP address information of wlan0 through DHCP, and then manually configure the connection WIFI.

- **Ensure that the wlan0 network device is active**

```
root@myd-ld25x:~# ifconfig wlan0 up
```

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

```
root@myd-ld25x:~# wpa_passphrase MYIR_5G myir@1601 >> /etc/wpa_supplicant.conf
root@myd-ld25x:~# 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_5G"
    #psk="myir@1601"
```

```
psk=dcc730fb7753a3e7464924b58a7fcff62a3af5abdf91db8a3265e759d427
2810
}
```

A WPA PSK is generated from the ASCII password of an SSID for encryption operation.

● Shut down the wpa_supplicant process

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

```
root@myd-ld25x:~# killall wpa_supplicant
```

● Initialize wpa_supplicant

wpa_supplicant is a tool to connect and configure WIFI. Its main work 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 like this:

```
root@myd-ld25x:~# wpa_supplicant -B -i wlan0 -c /etc/wpa_supplicant.conf
Successfully initialized wpa_supplicant
nl80211: kernel reports: Match already configured
ctrl_iface exists and seems to be in use - cannot override it
Delete '/var/run/wpa_supplicant/wlan0' manually if it is not used anymore
Failed to initialize control interface '/var/run/wpa_supplicant'.
You may have another wpa_supplicant process already running or the file was
left by an unclean termination of wpa_supplicant in which case you will need
to manually remove this file before starting wpa_supplicant again.

wlan0: Do not deauthenticate as part of interface deinit since WoWLAN is enabled
```

```
wlan0: CTRL-EVENT-DSCP-POLICY clear_all
nl80211: deinit ifname=wlan0 disabled_11b_rates=0
```

➤ -c: The path for the configuration information

● **Get the WiFi IP address**

```
root@myd-ld25x:~# udhcpc -i wlan0
udhcpc: started, v1.36.1
Dropped protocol specifier '.udhcpc' from 'wlan0.udhcpc'. Using 'wlan0' (ifindex=5).
udhcpc: broadcasting discover
udhcpc: broadcasting select for 192.168.20.59, server 192.168.20.92
udhcpc: lease of 192.168.20.59 obtained from 192.168.20.92, lease time 3599
/etc/udhcpc.d/50default: Adding DNS 192.168.20.92
Dropped protocol specifier '.udhcpc' from 'wlan0.udhcpc'. Using 'wlan0' (ifindex=5)
```

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

```
root@myd-ld25x:~# ifconfig wlan0
wlan0      Link encap:Ethernet  HWaddr C0:F5:35:C7:2D:5D
           inet addr:192.168.20.59  Bcast:192.168.20.255  Mask:255.255.255.0
           inet6 addr: fe80::c2f5:35ff:fec7:2d5d/64 Scope:Link
           UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
           RX packets:7 errors:0 dropped:0 overruns:0 frame:0
           TX packets:87 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:1000
           RX bytes:1126 (1.0 KiB)  TX bytes:12768 (12.4 KiB)
```

Then try to PING Baidu website.

```
root@myd-ld25x:~# ping www.baidu.com
PING www.a.shifen.com (153.3.238.102) 56(84) bytes of data.
```

```
64 bytes from 153.3.238.102 (153.3.238.102): icmp_seq=1 ttl=52 time=21.2 ms
64 bytes from 153.3.238.102 (153.3.238.102): icmp_seq=2 ttl=52 time=37.1 ms
64 bytes from 153.3.238.102 (153.3.238.102): icmp_seq=3 ttl=52 time=31.2 ms
^C
--- www.a.shifen.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 21.219/29.839/37.141/6.566 ms
```

Indicates that WIFI network is available.

2)Open the hotspot in AP mode

Hostapd is a wireless access point program with encryption capabilities. It is a convenient tool for building wireless access points on Linux systems and supports IEEE 802.11 protocols as well as IEEE 802.1X/WPA/WPA2/EAP/RADIUS encryption. When the board is configured as a Wi-Fi hotspot, it needs to assign IP addresses, routing, and other network parameters to each device connected to the hotspot (e.g., smartphones). For instance, if the SSID is `test` and the PASSWORD is `12345678`, you can use the following script to enable the hotspot:

2.1)shell script enable hotspot

We will organize the manual Wi-Fi configuration process into a script /etc/myir_test/wifi_on_ap for user reference, execute this script to configure the AP mode, after successful execution, users can use other devices to connect to the MYIR_TEST hotspot for testing.

```
root@myd-ld25x:~# cd /etc/myir_test/
root@myd-ld25x:/etc/myir_test# ./wifi_on_ap
killall: hostapd: no process killed
find phy0 enable it
wlan0: interface state UNINITIALIZED->ENABLED
wlan0: AP-ENABLED
```

After connecting the development board hotspot with the mobile device, the network ping packet is carried out.

```
root@myd-LD25X-emmc:/etc/myir_test# ping -c 10 192.168.10.22
PING 192.168.10.22 (192.168.10.22): 56 data bytes
64 bytes from 192.168.10.22: seq=0 ttl=64 time=97.795 ms
64 bytes from 192.168.10.22: seq=1 ttl=64 time=120.983 ms
64 bytes from 192.168.10.22: seq=2 ttl=64 time=27.000 ms
64 bytes from 192.168.10.22: seq=3 ttl=64 time=67.036 ms
^C
--- 192.168.10.22 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 27.000/78.203/120.983 ms

rtt min/avg/max/mdev = 4.682/47.490/113.485/40.762 ms
root@myd-LD25X-emmc:/#
```

5.1.4. 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.

- **Configuring Bluetooth**

We have created a script for configuring Bluetooth, which is placed in /etc/myir_test. You can simply run this script to configure Bluetooth.

```
root@myd-ld25x:~# cd /etc/myir_test/
root@myd-ld25x:/etc/myir_test# ./myir_bt
./myir_bt: line 3: basicsh: No such file or directory
./myir_bt: line 10: kill_process_fun: command not found
root@myd-ld25x:/etc/myir_test# option tosleep with arg 200000
option no2bytes
```



```
option enable_hci
option patchram with arg /lib/firmware/bcmd/BCM4345C5_003.006.006.1043.10
93.hcd
/dev/ttySTM6
writing
01 03 0c 00
received 7
04 0e 04 01 03 0c 00
writing
01 2e fc 00
received 7
04 0e 04 01 2e fc 00
[ 43.161511] Bluetooth: Core ver 2.22
[ 43.161655] NET: Registered PF_BLUETOOTH protocol family
[ 43.164851] Bluetooth: HCI device and connection manager initialized
[ 43.171156] Bluetooth: HCI socket layer initialized
[ 43.175911] Bluetooth: L2CAP socket layer initialized
[ 43.180984] Bluetooth: SCO socket layer initialized
writing
01 4c fc 46 00 00 22 00 42 52 43 4d 63 66 67 53
00 00 00 00 32 00 00 00 01 01 04 18 92 00 00 00
03 06 ac 1f 00 c5 45 43 00 01 1c 42 00 22 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 fe 00 00
received 7
04 0e 04 01 4c fc 00
writing
.....
```

At this point, the system is configuring Bluetooth. Please wait for a few seconds.

- **View Bluetooth device information**

```
root@myd-ld25x:/etc/myir_test# hciconfig -a
```

```
hci0: Type: Primary Bus: UART
      BD Address: C0:F5:35:C7:2D:5E ACL MTU: 1021:8 SCO MTU: 64:1
      DOWN
      RX bytes:740 acl:0 sco:0 events:42 errors:0
      TX bytes:463 acl:0 sco:0 commands:42 errors:0
      Features: 0xbf 0xfe 0xcf 0xfe 0xdb 0xff 0x7b 0x87
      Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
      Link policy: RSWITCH SNIFF
      Link mode: PERIPHERAL ACCEPT
```

- **Enable Bluetooth devices**

```
root@myd-ld25x:/etc/myir_test# hciconfig hci0 up
```

- **Enter the Bluetooth control interface**

```
root@myd-ld25x:/etc/myir_test# bluetoothctl
Agent registered
[CHG] Controller C0:F5:35:C7:2D:5E Pairable: yes
[bluetooth]# power on
Changing power on succeeded
```

- **Enable Bluetooth proxy**

```
[bluetooth]# agent on
Agent is already registered
```

- **Scan for nearby Bluetooth devices that can be connected**

```
[bluetooth]# scan on
Discovery started
[CHG] Controller C0:F5:35:C7:2D:5E Discovering: yes
[NEW] Device 75:DE:09:35:31:7F 75-DE-09-35-31-7F
[NEW] Device 43:A7:13:E1:52:C2 43-A7-13-E1-52-C2
[NEW] Device 54:14:F3:CA:70:37 BESTE
[CHG] Device 43:A7:13:E1:52:C2 ManufacturerData Key: 0x004c
[CHG] Device 43:A7:13:E1:52:C2 ManufacturerData Value:
[DEL] Device 75:DE:09:35:31:7F 75-DE-09-35-31-7F
```

```
[DEL] Device 43:A7:13:E1:52:C2 43-A7-13-E1-52-C2
[NEW] Device 75:DE:09:35:31:7F 75-DE-09-35-31-7F
[NEW] Device 60:D2:36:C1:09:4B 60-D2-36-C1-09-4B
[NEW] Device E4:E2:6C:95:69:58 12
```

● Pairing devices

```
[bluetooth]# pair E4:E2:6C:95:69:58
Attempting to pair with E4:E2:6C:95:69:58
[CHG] Device E4:E2:6C:95:69:58 Connected: yes
Request confirmation
[agent] Confirm passkey 790226 (yes/no): yes
[CHG] Device E4:E2:6C:95:69:58 Bonded: yes
[CHG] Device E4:E2:6C:95:69:58 Modalias: bluetooth:v079Ap1200d1436
[CHG] Device E4:E2:6C:95:69:58 UUIDs: 00001105-0000-1000-8000-00805f9b34f
b
[CHG] Device E4:E2:6C:95:69:58 ServicesResolved: yes
[CHG] Device E4:E2:6C:95:69:58 Paired: yes
Pairing successful
```

● Connecting devices

```
[bluetooth]# connect E4:E2:6C:95:69:58
Attempting to connect to E4:E2:6C:95:69:58
[CHG] Device E4:E2:6C:95:69:58 Connected: yes
[NEW] Device 76:D6:26:B4:10:D9 76-D6-26-B4-10-D9
[NEW] Endpoint /org/bluez/hci0/dev_E4_E2_6C_95_69_58/sep9
[NEW] Endpoint /org/bluez/hci0/dev_E4_E2_6C_95_69_58/sep11
[NEW] Endpoint /org/bluez/hci0/dev_E4_E2_6C_95_69_58/sep12
[NEW] Endpoint /org/bluez/hci0/dev_E4_E2_6C_95_69_58/sep13
[NEW] Endpoint /org/bluez/hci0/dev_E4_E2_6C_95_69_58/sep14
[NEW] Transport /org/bluez/hci0/dev_E4_E2_6C_95_69_58/sep9/fd0
Connection successful
[12]# [ 3342.452000] rtk_btcoex: hci (periodic)inquiry start
```

5.2. Network applications

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

5.2.1. PING

PING is primarily used to test network connectivity, but it can also be used to test network latency and packet loss rates. Once you've configured your Ethernet connection as described in 5.1, you can simply test your network connection using PING.

1) Wiring and information output

Connect the device to a switch or router through a network cable, such as the end2 network port of the development board.

```
root@myd-ld25x:~# ip route
default via 192.168.30.1 dev end2 proto dhcp src 192.168.30.129 metric 1024
192.168.1.0/24 dev end2 proto kernel scope link src 192.168.1.10
192.168.30.0/24 dev end2 proto kernel scope link src 192.168.30.129 metric 1
024
192.168.30.1 dev end2 proto dhcp scope link src 192.168.30.129 metric 1024
```

2) Test external sites

```
root@myd-ld25x:~# ping www.baidu.com -l end2
PING www.a.shifen.com (153.3.238.102) from 192.168.30.129 end2: 56(84) byte
s of data.
64 bytes from 153.3.238.102 (153.3.238.102): icmp_seq=1 ttl=53 time=14.8 ms
64 bytes from 153.3.238.102 (153.3.238.102): icmp_seq=2 ttl=53 time=17.1 ms
64 bytes from 153.3.238.102 (153.3.238.102): icmp_seq=3 ttl=53 time=13.3 ms
^C
--- www.a.shifen.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
```

```
rtt min/avg/max/mdev = 13.339/15.065/17.086/1.543 ms
```

Note: ping the public network needs to ensure that the DNS is working properly.

The result shows that the IP address of www.baidu.com after domain name resolution is 153.3.238.102, time is the delay in the response, but the shorter the better. In addition to testing Ethernet, the ping command can also be used to test Wi-Fi.

5.2.2. SSH

SSH stands for Secure Shell and was developed by the Network Working Group of the IETF. SSH is a security protocol built on the application layer and is designed to provide secure remote login sessions and other network services. It is a reliable protocol for this purpose. On Linux platforms, OpenSSH or Dropbear is commonly used to implement SSH server and client functionalities. Below, we will test the use of SSH client and server over an Ethernet connection. The device currently includes a client and server program provided by Dropbear (<http://www.Dropbear.com/>) by default.

First, configure the Ethernet interface for connection to the SSH server according to section 5.1.1. The address of the configured Ethernet card is as follows:

```
root@myd-ld25x:~# ip route
default via 192.168.30.1 dev end2 proto dhcp src 192.168.30.129 metric 1024
192.168.1.0/24 dev end2 proto kernel scope link src 192.168.1.10
192.168.30.0/24 dev end2 proto kernel scope link src 192.168.30.129 metric 1
024
192.168.30.1 dev end2 proto dhcp scope link src 192.168.30.129 metric 1024
```

The IP address of the SSH server is 192.168.30.129. After testing the connection between the device and the SSH server with the ping command, the following tests can be performed.

- **SSH client testing**

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

```
root@myd-ld25x:~# ssh myir@192.168.40.22

Host '192.168.40.22' is not in the trusted hosts file.
(ecdsa-sha2-nistp256 fingerprint sha1!! 41:67:c2:e2:13:f7:52:b9:6b:52:d1:cd:85:5a:43:30:c3:f5:67:aa)
Do you want to continue connecting? (y/n) y
myir@192.168.40.22's password:
Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.15.0-113-generic x86_64)

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

* Strictly confined Kubernetes makes edge and IoT secure. Learn how Micro
K8s
  just raised the bar for easy, resilient and secure K8s cluster deployment.

  https://ubuntu.com/engage/secure-kubernetes-at-the-edge

268 updates can be installed immediately.
1 of these updates is a security update.
To see these additional updates run: apt list --upgradable

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 Jul 31 13:58:51 2024 from 192.168.40.41
```

```
myir@system2:~$
```

After the successful login, the SSH server will automatically enter the console console, the user can perform the control of the remote server in the client mylinux user rights. If you need to exit, simply execute the "exit" command in the current console.

- **SSH server test**

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

Since the SSH server is also enabled on the device by default, we can also use the SSH command to log in to the current device on any other device that has an ssh client. The command and results are as follows:

```
myir@system2:~$ ssh root@192.168.30.129
```

```
Host '192.168.30.151' is not in the trusted hosts file.
```

```
(ssh-rsa fingerprint sha1!! c8:c1:22:17:d0:f1:67:fb:c7:b1:5c:1c:2a:d0:ce:18:d4:ef:21:09)
```

```
Do you want to continue connecting? (y/n) y
```

```
root@myd-ld25x:~#
```

In the above example, 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 exit, simply issue the "exit" command in the console.

Dropbear is a lightweight SSH server and client implementation designed to offer a small, simple, and efficient SSH solution, particularly suitable for embedded systems and resource-constrained environments. It occupies much less ROM and RAM compared to OpenSSH. It encrypts all traffic to ensure the confidentiality and integrity of data, protecting against eavesdropping, connection hijacking, and other network attacks.

For example, if you want the SSH server to disallow remote login for the root account, you can modify the /etc/default/dropbear file on the server device and add -w after -B.

```
root@myd-ld25x:~# vi /etc/default/dropbear
# Disallow root logins by default
DROPBEAR_EXTRA_ARGS=" -B -w"
```

5.2.3. 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 is very practical.

In section 5.2.2, we have already discussed examples of remote login using the SSH protocol with SSH clients and servers. Here, we will now introduce an example of file remote copying using the SCP command:

1) Copy files from remote to local

```
$myir@bsp:/media/myir$ scp test.txt root@192.168.30.151:/
test.txt
```

Go to the board/directory to see this file, as follows:

```
root@myd-ld25x:/# ls test.txt
test.txt
```

2) Copy files from local to remote

```
root@myd-ld25x:/# scp test.txt myir@192.168.40.22:/media/myir
myir@192.168.40.22's password:
test.txt                                100%    0    0.0KB/s    00:00
```

In the process of copying, you need to follow the prompts to input. After successful verification, the file is copied from the device to the/directory of the specified account on the server.

It is also possible to copy directories by adding the "-r" flag; see the scp command for help.

5.2.4. 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 server-side Linux system through the network and realize the function of network boot.

The default image file contains the tftp client program provided by busybox, which has the following command syntax:

```
root@myd-ld25x:~# tftp --help
Usage: tftp [-4][-6][-v][-l][-m mode] [host [port]] [-c command]
```

The detailed parameters are as follows:

- -4: Force the use of IPv4 addresses.
- -6: Force the use of IPv6 addresses.
- -v: Display detailed version information.
- -l: Specify a local file name (when using put or get commands).
- -m mode: Specify the transfer mode (typically netascii or octet).
- -host: Specify the hostname or IP address of the TFTP server.
- -port: Specify the port of the TFTP server (default is 69).
- -c command: Specify the TFTP command to execute, such as get, put, ls, etc.

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 tftp server configuration.

1) Install 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 machine.

```
root@myd-ld25x:/# tftp -g -r test.txt -l test.txt 192.168.40.22
```

This will download test.txt from <WORKDIR>/tftpboot on the tftp server to the current directory of the target machine.

```
root@myd-ld25x:/# tftp -p -l LD25X.txt -r LD25X-board.txt 192.168.40.22
```

The preceding command will upload the LD25Xx.txt file from the current directory on the target device to the <WORKDIR>/tftpboot directory configured earlier on the tftp server and rename it LD25X-board.txt.

5.2.5. Ethtool

ethtool is a tool for viewing and modifying Ethernet device parameters, 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, we look at the help for the command via ethtool -h:

```
root@myd-ld25x:~# ethtool --help
ethtool version 6.2
Usage:
    ethtool [ FLAGS ] DEVNAME      Display standard information about
device
    ethtool [ FLAGS ] -s|--change DEVNAME      Change generic options
    [ speed %d ]
    [ lanes %d ]
    [ duplex half|full ]
    [ port tp|aui|bnc|mii|fibre|da ]
    [ mdix auto|on|off ]
    [ autoneg on|off ]
    [ advertise %x[%x] | mode on|off ... [--] ]
    [ phyad %d ]
    [ xcvr internal|external ]
    [ wol %d[%d] | p|u|m|b|a|g|s|f|d... ]
    [ sopass %x:%x:%x:%x:%x:%x ]
....
```

View basic information about the current device's Ethernet card:

```
root@myd-ld25x:~# ethtool end2
Settings for end2:
    Supported ports: [ TP      MII ]
    Supported link modes:  10baseT/Full
                        100baseT/Full
```

```

1000baseT/Full
Supported pause frame use: Symmetric Receive-only
Supports auto-negotiation: Yes
Supported FEC modes: Not reported
Advertised link modes: 10baseT/Full
                        100baseT/Full
                        1000baseT/Full
Advertised pause frame use: Symmetric Receive-only
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: ug
Wake-on: d
    Current message level: 0x0000003f (63)
                        drv probe link timer ifdown ifup

Link detected: yes

```

Through the `ethtool` command, you can see that 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 5 and so on.

We can also use the `ethtool` tool to set Ethernet parameters, which are useful for Ethernet debugging and diagnostics. For example, we can force Ethernet to be set to 100 MBPS full-duplex and turn off self-negotiation with the following command:

```
root@myd-ld25x:~# ethtool -s end2 speed 100 duplex full autoneg off
```

```
[ 987.439801] stm32-dwmac 482d0000.eth2 end2: yt8521_read_status, phy address: 5, link down
r[ 987.442212] stm32-dwmac 482d0000.eth2 end2: Link is Down
root@myd-ld25x:~# [ 991.515016] stm32-dwmac 482d0000.eth2 end2: yt8521_read_status, phy address: 5, link up, media: UTP, mii reg 0x11 = 0x6c40
[ 991.522153] stm32-dwmac 482d0000.eth2 end2: Link is Up - 100Mbps/Full - flow control off
```

```
root@myd-ld25x:~# ethtool end2
```

```
Settings for end2:
```

```
Supported ports: [ TP      MII ]
```

```
Supported link modes:  10baseT/Full
```

```
                      100baseT/Full
```

```
                      1000baseT/Full
```

```
Supported pause frame use: Symmetric Receive-only
```

```
Supports auto-negotiation: Yes
```

```
Supported FEC modes: Not reported
```

```
Advertised link modes: 10baseT/Full
```

```
                      100baseT/Full
```

```
                      1000baseT/Full
```

```
Advertised pause frame use: Symmetric Receive-only
```

```
Advertised auto-negotiation: No
```

```
Advertised FEC modes: Not reported
```

```
Speed: 100Mb/s
```

```
Duplex: Full
```

```
Auto-negotiation: off
```

```
Port: Twisted Pair
```

```
PHYAD: 5
```

```
Transceiver: internal
```

```
MDI-X: Unknown
```

```
Supports Wake-on: ug
```

```
Wake-on: d
```

```
Current message level: 0x0000003f (63)
```

```
drv probe link timer ifdown ifup
```

```
Link detected: yes
```

For more on ethtool: <http://man7.org/linux/man-pages/man8/ethtool.8.html>.

5.2.6. iPerf3

iPerf3 is a tool to actively measure the maximum achievable bandwidth over IP networks. It supports tuning various parameters such as test time, buffer size, and protocol (TCP, UDP, SCTP for 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 system, a server with a gigabit network card 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 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 192.168.0.5 and the device IP to 192.168.0.10 and use the ping command to test to make sure they are connected.

Note: Try not to connect routers or switches, lest the test results be affected by intermediate devices.

1) Test TCP performance

- **Server side (192.168.0.5)**

The on-server iperf3 uses the -s parameter to indicate that it works in server-side mode.

```
$: iperf3 -s -i 2
```

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

● **Client(192.168.0.10)**

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

- -c 192.168.0.5: Working on client side, connecting to server side 192.168.0.5
- -i 2: Test results are reported at a time interval of 2 seconds
- -t 10: The total test duration is 10 seconds
- -B 192.168.0.10: Specifies the id of the current eth1

```
root@myd-ld25x:/# iperf3 -c 192.168.0.5 -i 2 -t 10 -B 192.168.0.10
```

```
Connecting to host 192.168.0.5, port 5201
```

```
[ 5] local 192.168.0.10 port 46289 connected to 192.168.0.5 port 5201
```

[ID]	Interval	Transfer	Bitrate	Retr	Cwnd
[5]	0.00-2.00 sec	223 MBytes	937 Mb/s	0	2.27 MBytes
[5]	2.00-4.00 sec	222 MBytes	933 Mb/s	0	3.12 MBytes
[5]	4.00-6.00 sec	225 MBytes	944 Mb/s	0	1.76 MBytes
[5]	6.00-8.00 sec	225 MBytes	944 Mb/s	0	1.91 MBytes
[5]	8.00-10.00 sec	224 MBytes	938 Mb/s	0	2.00 MBytes

```
-----  
[ ID] Interval      Transfer      Bitrate      Retr  
[ 5]  0.00-10.00 sec  1.09 GBytes  939 Mb/s      0  
[ 5]  0.00-10.04 sec  1.09 GBytes  938 Mb/s  
iperf Done.
```

The client ends the test after 10 seconds and displays the above test result, which shows that the TCP bandwidth is around 930Mbps.

At the same time, the server displays the test results as follows and continues listening on port 5201 for the client to connect

```
$ iperf3 -s -i 2
-----
Server listening on 5201
-----
Accepted connection from 192.168.0.10, port 59261
[ 5] local 192.168.0.5 port 5201 connected to 192.168.0.10 port 46289
[ ID] Interval            Transfer        Bitrate
[ 5]  0.00-1.00    sec    107 MBytes    894 Mbits/sec
[ 5]  1.00-2.00    sec    111 MBytes    933 Mbits/sec
[ 5]  2.00-3.00    sec    112 MBytes    936 Mbits/sec
[ 5]  3.00-4.00    sec    112 MBytes    939 Mbits/sec
[ 5]  4.00-5.00    sec    111 MBytes    934 Mbits/sec
[ 5]  5.00-6.00    sec    112 MBytes    937 Mbits/sec
[ 5]  6.00-7.00    sec    112 MBytes    939 Mbits/sec
[ 5]  7.00-8.00    sec    112 MBytes    940 Mbits/sec
[ 5]  8.00-9.00    sec    112 MBytes    939 Mbits/sec
[ 5]  9.00-10.00   sec    112 MBytes    937 Mbits/sec
[ 5] 10.00-10.04   sec    4.83 MBytes    942 Mbits/sec
-----
[ ID] Interval            Transfer        Bitrate
[ 5]  0.00-10.04   sec    1.09 GBytes    939 Mbits/sec
-----
Server listening on 5201
-----
```

2) Test UDP performance

- Server (192.168.0.5)

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


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

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

● **Client (192.168.0.10)**

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

- -u : Works in UDP mode
- -c 192.168.0.5 : Working on client side, connecting to server side 192.168.0.5
- -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 1000Mbps.

```
root@myd-ld25x:/# iperf3 -u -c 192.168.0.5 -i 2 -t 10 -b 1000M -B 192.168.0.10 -b 1G
```

```
Connecting to host 192.168.0.5, port 5201
```

```
[ 5] local 192.168.0.10 port 40610 connected to 192.168.0.5 port 5201
```

[ID]	Interval		Transfer	Bitrate	Total Datagrams
[5]	0.00-2.00	sec	154 MBytes	647 Mbites/sec	111761
[5]	2.00-4.00	sec	155 MBytes	648 Mbites/sec	111905
[5]	4.00-6.00	sec	155 MBytes	651 Mbites/sec	112453
[5]	6.00-8.00	sec	155 MBytes	651 Mbites/sec	112324
[5]	8.00-10.00	sec	155 MBytes	650 Mbites/sec	112268

```
-----  
[ ID] Interval          Transfer      Bitrate          Jitter    Lost/Total Datagrams  
ms  
[ 5]  0.00-10.00 sec    774 MBytes   650 Mbites/sec  0.000 ms   0/560711  
(0%) sender
```

```
[ 5] 0.00-10.04 sec 774 MBytes 647 Mb/s 0.017 ms 0/560711
(0%) receiver
iperf Done.
```

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

```
$ iperf3 -s -i 2

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

Accepted connection from 192.168.0.10, port 59261
[ 5] local 192.168.0.5 port 5201 connected to 192.168.0.10 port 46289
[ ID] Interval            Transfer        Bitrate          Jitter    Lost/Total Datagrams
ms
[ 5] 0.00-1.00 sec 73.5 MBytes 616 Mb/s 0.017 ms 0/53201 (0%)
[ 5] 1.00-2.00 sec 77.5 MBytes 650 Mb/s 0.017 ms 0/56126 (0%)
[ 5] 2.00-3.00 sec 77.5 MBytes 650 Mb/s 0.017 ms 0/56089 (0%)
[ 5] 3.00-4.00 sec 77.1 MBytes 647 Mb/s 0.015 ms 0/55830 (0%)
[ 5] 4.00-5.00 sec 77.5 MBytes 650 Mb/s 0.016 ms 0/56089 (0%)
[ 5] 5.00-6.00 sec 77.8 MBytes 653 Mb/s 0.017 ms 0/56348 (0%)
[ 5] 6.00-7.00 sec 77.5 MBytes 650 Mb/s 0.016 ms 0/56150 (0%)
[ 5] 7.00-8.00 sec 77.6 MBytes 651 Mb/s 0.018 ms 0/56187 (0%)
[ 5] 8.00-9.00 sec 77.5 MBytes 650 Mb/s 0.019 ms 0/56101 (0%)
[ 5] 9.00-10.00 sec 77.6 MBytes 651 Mb/s 0.015 ms 0/56162
(0%)
[ 5] 10.00-10.04 sec 3.35 MBytes 648 Mb/s 0.017 ms 0/2428 (0%)
-----
[ ID] Interval            Transfer        Bitrate          Jitter    Lost/Total Datagrams
ms
[SUM] 0.0-10.0 sec 11 datagrams received out-of-order
[ 5] 0.00-10.04 sec 774 MBytes 647 Mb/s 0.017 ms 0/560711
(0%) receiver
```

Server listening on 5201

There are many parameters of iperf3 that can be configured in the process of testing. Users can adjust the test according to the actual application needs. For example, we can increase the -t parameter to stress over a long period of time, or specify the -P parameter to stress test multiple connections concurrently, and so on. For more information about the iperf3 test, see: <https://iperf.fr/iperf-doc.php#3doc>

6. Multimedia application system

6.1. Camera

In this section, the system's own gstreamer are used to evaluate and test the csi camera. The main test camera preview and take pictures.

The camera wiring is shown in the diagram below:

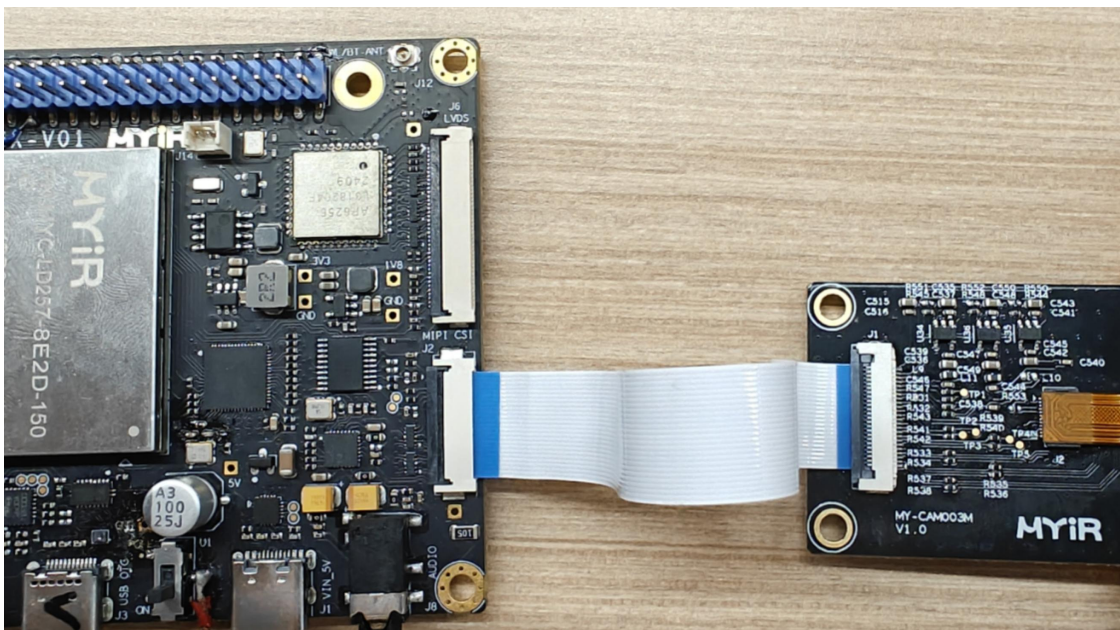


Figure 6-1. Camera Wiring

1) Camera window preview and photo test

- View the basic information of the device

When a MY-CAM003M camera is inserted, the character device `/dev/video0` is generated

```
root@myd-ld25x:~# ls /dev/video*  
video0 video1 video2 video3 video4 video5 video6
```

- Use gstream scrpit for preview

We have placed a script for previewing the camera in `/etc/myir_test`. To run the script, execute the following command in the terminal:

```
root@myd-ld25x:~# cd /etc/myir_test/
root@myd-ld25x:/etc/myir_test# killall mxapp2
root@myd-ld25x:/etc/myir_test# ./myir_camera_play
With GPU
ls: cannot access 'media*': No such file or directory
/etc/myir_test
VIDIOC_S_PARM: failed: Inappropriate ioctl for device
SCREEN_WIDTH= 1920
SCREEN_HEIGHT=1080
With GPU
ls: cannot access 'media*': No such file or directory
/etc/myir_test
VIDIOC_S_PARM: failed: Inappropriate ioctl for device
SCREEN_WIDTH= 1920
SCREEN_HEIGHT=1080
GStreamer graph:
v4l2src device=/dev/video0 io-mode=4 ! video/x-raw, width=640, height=480 ! queue ! gtkwaylandsink name=gtkwsink
killall: launch_camera_control_mp25.sh: no process killed
Terminated
.....
```

After execution, the screen generates a preview window with a length of 640, a width of 480 and a frame rate of 30 frames per second.

- **Use the gstream tool to take a picture**

To capture a photo from the camera and save it as test.jpg using the gst-launch-1.0 command from the GStreamer toolkit, execute the following command in the terminal:

Note: Ensure that video* matches the video* used in the previous camera preview setup.

```
root@myd-ld25x:~# gst-launch-1.0 v4l2src device=/dev/video0 num-buffers=1 !  
video/x-raw,format=RGB16,width=640,height=480 ! queue ! videoconvert ! jp  
egenc ! filesink location=test.jpg
```

6.2. Audio

1) Audio Playback Test

Use the system's built-in music and script for playback. Connect the playback device to the onboard 3.5mm audio jack. The playback script `myir_audio_play` is located in the directory `/etc/myir_test` as follows:

```
root@myd-ld25x:/etc/myir_test# ./myir_audio_play
Press 'k' to see a list of keyboard shortcuts.
Now playing /usr/share/myir/Music/song.mp3
Redistribute latency...
Redistribute latency...
0:00:00.8 / 0:01:22.8
```

2) Audio Recording Test

Use the following commands for the recording test:

```
root@myd-ld25x:~# amixer -c 0 set "Capture ZC" on
root@myd-ld25x:~# amixer -c 0 set "Capture Digital" 120
root@myd-ld25x:~# arecord -D hw:0,0 -f cd test.wav
```

After executing the `arecord` command, the system will enter recording mode. Start recording, and then press Ctrl+C to stop. You can then use the `aplay` command to play back the recorded file:

```
root@myd-ld25x:~# aplay -D hw:0,0 test.wav
```

6.3. Video Playback

We have placed a script for automatic video playback, named `myir_video_play`, in the system directory `/etc/myir_test/` on the MYD-LD25X development board. To perform a playback test, connect either the MY-LVDS070C display or an HDMI screen as described in section 3.7, and then execute the following command:

```
root@myd-ld25x:/etc/myir_test# ./myir_video_play
Press 'k' to see a list of keyboard shortcuts.
Now playing /usr/share/myir/Video/myir.mp4
Redistribute latency...
Redistribute latency...
Redistribute latency...
Redistribute latency...
Redistribute latency...
Redistribute latency...
Redistribute latency...
0:00:04.2 / 0:00:11.3
```

The video playback is shown in the diagram below:

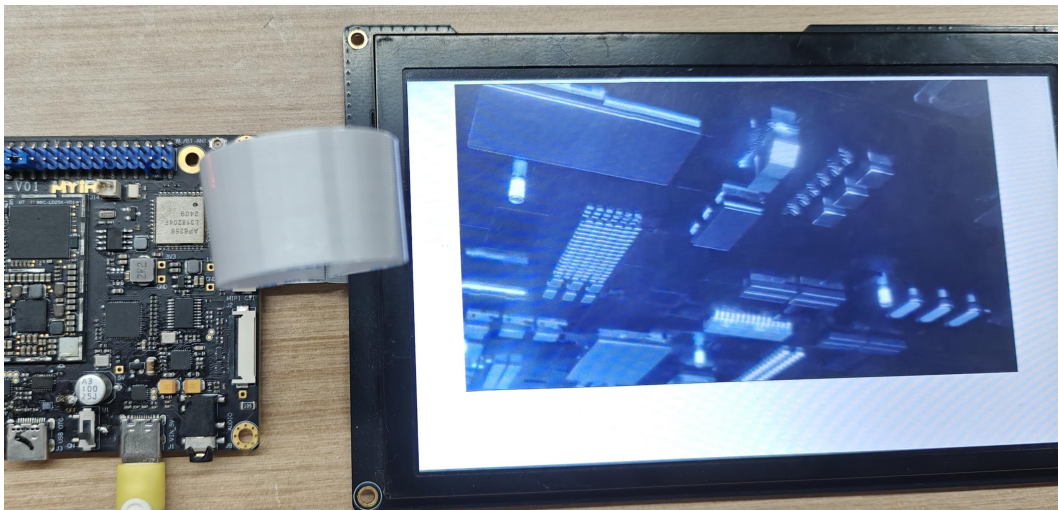


Figure 6-2. Video Playback

7. References

- **"MYD-LD25X Linux Software Development Guide"**

- **Introduction to the Linux kernel watchdog**

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

- **Linux kernel open source community**

<https://www.kernel.org/>

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@myir.cn), 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@myir.cn

Sales Email: sales@myir.cn

Phone: +86-755-22984836

Fax: +86-755-25532724

Website: en.myir.cn