SigmaStar Camera
PROC 调试信息说明

Version 0.1

REVISION HISTORY

Revision No.	Description	Date
{Version 0.1}	• {Initial release}	{12/20/2017}

TABLE OF CONTENTS

1. 概述

1.1. 简介

【说明】

使用 Linux 下的 porc 文件系统，可以打印出各个模块当前的运行状态，便于调试及分析问题。每个模块都自己特定的路径，部分模块有额外的 echo 命令。

【路径】

路径为 /proc/mi_modules

1.2. 文件清单

Table 1: 表 1-1

模块名称	说明
SYS	SYS 模块的信息
AI	音频输入的信息
AO	音频输出的信息
VIF	视频输入的信息
VPE	视频处理引擎的信息
VENC	编码视频的信息
DIVP	去隔行/图像引擎的信息
VDISP	虚拟显示模块的信息
DISP	显示模块的信息
HDMI	HDMI 的信息
FB	Graphic 图形层的信息
GFX	图像引擎的信息
REGION	OSD 的信息
VDEC	解码视频的信息

2. SYS

2.1. common

2.1.1 dump_mmap

【调试信息】

./config/dump_mmap

```
/ # ./config/dump_mmap
u32TotalSize:0x10000000
u32Miu0Size:0x10000000
u32Miu1Size:0x0
u32MiuBoundary:0x10000000
u32MmapItemsNum:14
bIs4kAlign:1
bMiu1Enable:0
E_MMAP_ID_DUMMY2:GID=0,Addr=0x0,Size=0x200000,Layer=0,Align=0x0,MemoryType=0x4,MiuNo=0,CMAID=0
E_MMAP_ID_CMDQ:GID=1,Addr=0x0,Size=0x30000,Layer=1,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
E_MMAP_ID_GFX:GID=2,Addr=0x30000,Size=0x40000,Layer=1,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
E_MMAP_ID_GE_VQ:GID=3,Addr=0x70000,Size=0x20000,Layer=1,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
E_MMAP_ID_AO:GID=4,Addr=0x90000,Size=0x4b000,Layer=1,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
E_MMAP_ID_AI:GID=5,Addr=0xdb000,Size=0x4b000,Layer=1,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
E_MMAP_ID_XC_MLOAD:GID=6,Addr=0x126000,Size=0x40000,Layer=1,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
E_MMAP_ID_HW_CURSOR:GID=7,Addr=0x166000,Size=0x20000,Layer=1,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
E_MMAP_ID_VDEC_CPU:GID=8,Addr=0x200000,Size=0xa00000,Layer=0,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
E_LX_MEM:GID=9,Addr=0xc00000,Size=0xe300000,Layer=0,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
E_EMAC_MEM:GID=10,Addr=0xef00000,Size=0x100000,Layer=0,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
E_MMAP_ID_XC_MAIN_FB:GID=11,Addr=0xf000000,Size=0x800000,Layer=0,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
E_MMAP_ID_FB:GID=12,Addr=0xf800000,Size=0x400000,Layer=0,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
E_MMAP_ID_MAD_R2:GID=13,Addr=0xfc00000,Size=0x400000,Layer=0,Align=0x1000,MemoryType=0x4,MiuNo=0,CMAID=0
```

Figure 2: 图 2-1

【调试信息分析】

Common 下的内容存放的是配置 API，之所以会有配置 API，有两个目的：

1. 把 iniparser 放在用户态做，iniparser 的任何开销不带入内核。
2. 方便调试时确认配置文件的实际内容，因为在 parser 时打印的内容可能会与实际设置的内容有偏差

为了达到上面两个目的，导出了一些文件。

如下是导入 config 的配置文件：

```
MI_COMMON_AddDebugRawFile("config_info", /*(sizeof
MI_COMMON_AddDebugRawFile("pq_info", /*_pqBaseSize
MI_COMMON_AddDebugRawFile("noise_table", /*4096, *
MI_COMMON_AddDebugRawFile("motion_table", /*4096,
MI_COMMON_AddDebugRawFile("motion_hdmi_dtv_table",
MI_COMMON_AddDebugRawFile("motion_comp_pc_table",
MI_COMMON_AddDebugRawFile("misc_table", /*4096, */
MI_COMMON_AddDebugRawFile("misc_luma_table", /*409
MI_COMMON_AddDebugRawFile("misc_param_table", /*40
```

Figure 2: 图 2-2

Systeminfo.c 里面 parse 好的 struct 写入 config_info 里面。

Pqloader.c 里面 parse 好的 struct 写入 pq_info 和*_table 里面，这里的 table 跟 MISDK 里面的那一份不一样的一点是这里的 table 是连续的内存摆放，MISDK 是一个间接数组指向一列一维数组。

如下是导入 mmap 的配置文件：

```
MI_COMMON_AddDebugRawFile("mmap_info",
MI_COMMON_AddDebugFile("memory_info",
```

Figure 3: 图 2-3

Mmapinfo.c 里面 parse 好的 struct 写入 mmap_info 和 memory_info，因为这里比较简单，所以从 mmap_info 和 memory_info 读出来是格式化好的可读信息。

2.1.2 dump_config

【调试信息】

```
# ./config/dump_config

panel size:24
DBC Value=
    55,30,10
    254,66,0
    72,80,72
start dump [motion_table](6, 8)
{0xe8,0xcd,0xab,0x89,0x67,0x45,0x23,0x00,}
{0xd8,0xbc,0x9a,0x78,0x56,0x34,0x12,0x00,}
{0xc8,0xab,0x89,0x67,0x45,0x23,0x01,0x00,}
{0xb8,0x9a,0x78,0x56,0x34,0x12,0x00,0x00,}
{0xa8,0x89,0x67,0x45,0x23,0x01,0x00,0x00,}
{0x98,0x78,0x56,0x34,0x12,0x00,0x00,0x00,}
end dump
start dump [motion_hdmi_dtv_table](4, 8)
{0x68,0x45,0x23,0x01,0x00,0x00,0x00,0x00,}
{0xa8,0x89,0x67,0x45,0x23,0x01,0x00,0x00,}
{0xc8,0xab,0x89,0x67,0x45,0x23,0x01,0x00,}
{0xe8,0xcd,0xab,0x89,0x67,0x45,0x23,0x00,}
end dump
start dump [motion_comp_pc_table](4, 8)
{0x68,0x45,0x23,0x01,0x00,0x00,0x00,0x00,}
{0x98,0x78,0x56,0x34,0x12,0x01,0x00,0x00,}
{0xa8,0x89,0x67,0x45,0x23,0x12,0x01,0x00,}
{0xb8,0x9a,0x78,0x56,0x34,0x12,0x01,0x00,}
end dump
start dump [misc_param_table](4, 4)
{0x02,0xff,0x00,0x00,}
{0x02,0xff,0x01,0x66,}
{0x03,0x88,0x01,0x66,}
{0x03,0xaa,0x01,0x66,}
end dump
```

start dump [misc_luma_table](8, 2)
{0x33,0x08,}
{0x33,0x07,}
{0x22,0x06,}
{0x22,0x05,}
{0x11,0x04,}
{0x11,0x03,}
{0x00,0x02,}
{0x00,0x01,}
end dump
start dump [noise_table](6, 8)
{0x67,0x45,0x34,0x23,0x12,0x01,0x00,0x00,}
{0xab,0x89,0x67,0x45,0x23,0x12,0x01,0x00,}
{0xbc,0x9a,0x78,0x56,0x34,0x23,0x01,0x00,}
{0xde,0xbc,0x9a,0x78,0x56,0x34,0x12,0x00,}
{0xef,0xcd,0xab,0x89,0x67,0x45,0x23,0x00,}
{0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,}
end dump
start dump [misc_table](6, 15)
{0x00,0x00,0x00,0x00,0x00,0x02,0x00,0x00,0x02,0x00,0xc0,0x9f,0x00,0x01,0x66,}
{0x10,0x01,0xc1,0xc1,0xc1,0x04,0x09,0x1f,0x03,0x88,0xc0,0x9f,0x00,0x01,0x66,}
{0x21,0x02,0xc2,0xc2,0xc2,0x08,0x09,0x1f,0x03,0x99,0x00,0x00,0x00,0x01,0x88,}
{0x31,0x04,0xc2,0xc2,0xc2,0x0a,0x09,0x1f,0x03,0xaa,0x00,0x00,0x00,0x01,0xaa,}
{0x41,0x06,0xc3,0xc3,0xc3,0x10,0x19,0x1f,0x03,0xbb,0x00,0x00,0x14,0x01,0xbb,}
{0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,}
end dump

【调试信息分析】
./config/dump_config 得到的是/proc/mi_modules/common/里各文件的信息，含 panel size、DBC Value、motion_table、motion_hdmi_dtv_table、motion_comp_pc_table、misc_param_table、misc_luma_table、noise_table、misc_table 等。

2.2. mi_log_info

2.2.1 cat

【调试信息】
cat /proc/mi_modules/mi_log_info

---------- Log Path ----------
log path:
---------- Store Path ----------
store path: /mnt
---------- Module Log Level ----------

```
Log module          Level
--------------------------
mi_ive              2(WRN)
mi_vdf              2(WRN)
mi_venc             2(WRN)
mi_rgn              2(WRN)
mi_ai               2(WRN)
mi_ao               2(WRN)
mi_vif              2(WRN)
mi_vpe              2(WRN)
mi_vdec             2(WRN)
mi_sys              2(WRN)
mi_fb               2(WRN)
mi_hdmi             2(WRN)
mi_divp             2(WRN)
mi_gfx              2(WRN)
mi_vdisp            2(WRN)
mi_disp             2(WRN)
mi_os               2(WRN)
mi_iae              2(WRN)
mi_md               2(WRN)
mi_od               2(WRN)
mi_shadow           2(WRN)
```

【调试信息分析】

示意了默认的 log path 或 store path，同时示意了各 module 的 debug_level 的值。

【参数分析】

Table 2: 表 2-1

参数		描述
mi_log_info	log path	暂时无效果
	store path	暂时无效果
	Log module	各个模块的名称
	Level	打印等级

2.2.2 echo

Table 3: 表 2-2

功能	修改模块 debug_level
命令	echo [ModID]=[Level] > /proc/mi_modules/mi_log_info
	[ModID] 模块的名字 mi_ive mi_vdf mi_venc mi_rgn

	mi_ai mi_ao mi_vif mi_vpe mi_vdec mi_sys mi_fb mi_hdmi mi_divp mi_gfx mi_vdisp mi_disp mi_os mi_iae mi_md mi_od mi_shadow
参数说明	[Level] 0 无 Debug 信息 1 只显示 error 的信息 (MI_DBG_ERR) 2 只显示 waring 的信息 (MI_DBG_WRN) 3 只显示 info 的信息 (MI_DBG_INFO) 4 显示所有信息
举例	echo mi_sys=2 > /proc/mi_modules/mi_log_info 将 mi_sys 的 debug_level 修改为 2

Table 4: 表 2-3

功能	修改 log 的路径
命令	echo log=[Path] > /proc/mi_modules/mi_log_info
参数说明	[Path] 路径
举例	echo log=/mnt > /proc/mi_modules/mi_log_info

Table 5: 表 2-4

功能	修改存储 log 的路径
命令	echo storepath=[Path] > /proc/mi_modules/mi_log_info
参数说明	[Path] 路径
举例	echo storepath=/mnt > /proc/mi_modules/mi_log_info

2.3. mi_global_info

2.3.1 cat

【调试信息】

```
# cat /proc/mi_modules/mi_global_info

miu_and_lx_info:
ARM_MIU0_BUS_BASE 0x20000000        ARM_MIU0_BASE_ADDR 0x0
ARM_MIU1_BUS_BASE 0x60000000        ARM_MIU1_BASE_ADDR 0x80000000
ARM_MIU2_BUS_BASE 0xffffffff        ARM_MIU2_BASE_ADDR 0xffffffff
lx_mem_addr   0x20c00000            lx_mem_size   0xe300000
lx_mem2_addr 0xffffffff             lx_mem2_size 0xffffffff
lx_mem3_addr 0xffffffff             lx_mem3_size 0xffffffff
```

KernelProtect IP white list:

clientId	name
43	MIU_CLIENT_MIPS_RW
50	MIU_CLIENT_NAND_RW
82	MIU_CLIENT_USB_UHC0_RW
83	MIU_CLIENT_USB_UHC1_RW
84	MIU_CLIENT_USB_UHC2_RW
18	MIU_CLIENT_G3D_RW
140	MIU_CLIENT_USB3_RW
129	MIU_CLIENT_SDIO_RW
165	MIU_CLIENT_SATA_RW
133	MIU_CLIENT_USB_UHC3_RW
225	MIU_CLIENT_USB30_1_RW
226	MIU_CLIENT_USB30_2_RW
5	MIU_CLIENT_BDMA_RW
14	MIU_CLIENT_EMAC_RW

PAGE_OFFSET - the virtual address of the start of the kernel image
PAGE_OFFSET=0xc0000000
TASK_SIZE - the maximum size of a user space task
TASK_SIZE=0xbf000000
MStar SDK version: commit.build_time 0be783c.2017121315
CHIP_VERSION U02

【调试信息分析】
该调试信息提供了全局 global 的一些信息。
【参数分析】
Table 5: 表 2-4

参数		描述
miu_and_lx_info(以只有一个 MIU 为例)	ARM_MIU0_BUS_BASE	Miu0 bus base
	ARM_MIU0_BASE_ADDR	Miu0 base addr
	lx_mem_addr	Linux 镜像占的 memory 的起始地址(属于 cpu bus address)
	lx_mem_size	Linux 镜像占的 memory 的 size
kernelProtect IP white list	clientId	Miu protect 的 IP 白名单里的 IP 的 id（从未分 group 的角度看的全局的 id）
	name	与 clientId 对应的该 IP 的实际的名字
MStar SDK version: commit.build_time	commit	sdk 对应的 commit
	build_time	sdk 的 build 时间

CHIP_VERSION	CHIP_VERSION	当前 chip 的版本，版本号是 U01,U02,U03…

2.4. debug_level

2.4.1 cat/echo

【调试信息】

cat /proc/mi_modules/mi_sys/debug_level

2

【调试信息分析】

每个 module（包括 sys 这个 module）都有各自的 debug level,是为了控制打印级别,各自的打印级别分别在 /proc/mi_modules/mi_modulename/debug_level 控制，其中 modulename 形如 disp,divp,rgn 等等。上面只是以/proc/mi_modules/mi_sys/debug_level 为例。

Table 5: 表 2-4

功能	打印警告级别
命令	cat /proc/mi_modules/[ModuleName]/debug_level
参数说明	[ModuleName] 模块的名字 mi_disp mi_gfx mi_rgn mi_vdec mi_vpe mi_ai mi_divp mi_shadow mi_vdisp mi_ao mi_hdmi mi_sys mi_venc mi_bar mi_vif
举例	cat /proc/mi_modules/mi_sys/debug_level 2 mi_sys 的警告级别为 2(只显示 waring 的信息)

Table 6: 表 2-5

功能	修改警告级别
命令	echo [Level] > /proc/mi_modules/[ModuleName]/debug_level
参数说明	[Level] 0 无 Debug 信息 1 只显示 error 的信息 (MI_DBG_ERR) 2 只显示 waring 的信息 (MI_DBG_WRN) 3 只显示 info 的信息 (MI_DBG_INFO) 4 显示所有信息
	[ModuleName] 模块的名字 mi_disp mi_gfx mi_rgn mi_vdec mi_vpe mi_ai mi_divp mi_shadow mi_vdisp mi_ao

	mi_hdmi mi_sys mi_venc mi_bar mi_vif
举例	echo 1 > /proc/mi_modules/mi_vdec/debug_level 将 mi_vdec 模块的警告级别修改为只显示 error 的信息

2.5. miu_protect

2.5.1 cat

【调试信息】

cat /proc/mi_modules/mi_sys_mma/miu_protect

```
==================== start miu_protect_info
================================
kernel protect enabled
LX :
cpu_start_addr:0x20c00000 size:0xe300000
miu_index miuBlockIndex start_cpu_bus_pa length
0x0 0x00 0x20c00000 0x460000
KernelProtect IP white list:
clientId name
43 MIU_CLIENT_MIPS_RW
50 MIU_CLIENT_NAND_RW
82 MIU_CLIENT_USB_UHC0_RW
83 MIU_CLIENT_USB_UHC1_RW
84 MIU_CLIENT_USB_UHC2_RW
18 MIU_CLIENT_G3D_RW
140 MIU_CLIENT_USB3_RW
129 MIU_CLIENT_SDIO_RW
165 MIU_CLIENT_SATA_RW
133 MIU_CLIENT_USB_UHC3_RW
225 MIU_CLIENT_USB30_1_RW
226 MIU_CLIENT_USB30_2_RW
5 MIU_CLIENT_BDMA_RW
14 MIU_CLIENT_EMAC_RW
```

【调试信息分析】

该命令显示了 miu protect 相关的信息。

【参数分析】

Table 7: 表 2-6

参数		描述
kernel protect		值是 enabled 或者 disabled，表

		示是否有 enable kernel protect，默认是需要 enable kernel protect 的。
LX(以只有一个 LX 为例,LX 表示 linux 镜像对应的 memory)	cpu_start_addr	该 LX 对应的起始 CPU addr。
	size	该 LX 对应的 size。
某个 kernel protect 的 range 的相关信息	miu_index	编号。
	miuBlockIndex	总共 4 个 miu 范围的编号信息。
	start_cpu_bus_pa	该 range 的起始 cpu bus addr。
	length	该 range 的 length。
KernelProtect IP white list	clientId	Miu protect 的 IP 白名单里的 IP 的 id（从未分 group 的角度看的全局 id）。
	name	与 clientId 对应的该 IP 的实际的名字。

2.6. mma_help_name

2.6.1 cat

【调试信息】

```
# cat /proc/mi_modules/mi_sys_mma/mma_heap_name0
mma heap name     heap_base_cpu_bus_addr      length                chunk_mgr_avail
      mma_heap_name0      25200000                9d00000                79c6000
chunk_mgr info:
      pst_chunk_mgr              offset              length              avail
            c3545618                   0              9d00000            79c6000

each chunk info:
            offset              length          used_flag          task_name
                 0                1000                  1             insmod
              1000              100000                  1             insmod
            101000              200000                  1          proc_rtsp
            301000              3f5000                  1          vdisp-dev0
            6f6000              3f5000                  1          vdisp-dev0
            aeb000              3f5000                  1          vdisp-dev0
            ee0000              1f5000                  0                 NA
           10d5000                1000                  1          proc_rtsp
           10d6000               64000                  1          proc_rtsp
           113a000              200000                  1          proc_rtsp
           133a000              700000                  1          proc_rtsp
           1a3a000              3f5000                  0                 NA
           1e2f000              3f5000                  1          vdisp-dev0
```

2224000	700000	1	proc_rtsp
2924000	73dc000		NA

【调试信息分析】

该 cat 信息对应的是该 mma heap 的基本信息和当前的一些状态。

【参数分析】

Table 8: 表 2-7

参数		描述
heap basic info	mma heap name	mma heap 的 name。
	heap_base_cpu_bus_addr	heap 的起始 cpu bus addr。
	length	heap 的 length。
	chunk_mgr_avail	heap 里的剩余的未用的 memory 的总量。
chunk_mgr info	pst_chunk_mgr	chunk mgr 的指针的值。
	offset	chunk mgr 的 offset,由于整个 mma heap 做为一个 chunk mgr,因此该值永远为 0。
	length	chunk mgr 的 length,由于整个 mma heap 做为一个 chunk mgr,因此该值永远为 mma heap length。
	avail	chunk mgr(即 heap)里的剩余的未用的 memory 的总量。
each chunk info (chunk mgr 里各 chunk 的信息和使用情况)	offset	该 chunk 在 chunk mgr 里的 offset。
	length	该 chunk 的 length。
	used_flag	该 chunk 是否被 alloc 出去使用了。是的话该值为 1；否则 free 状态的话该值为 0。
	task_name	如果 used flag 为 1，则 task_name 里存的是哪个 task 使用的它；否则该值为无效值 NA。

2.6.2 echo

Table 9: 表 2-8

功能	即 dump 指定 offset 和 length 的 data 到指定的路径
命令	echo [Path] [Offset] [Length] > /proc/mi_modules/mi_sys_mma/mma_heap_name[Num]

参数说明	[Path] 文件要保存的路径,只需提供路径，不要提供具体的文件名，系统会根据参数自动生成对应的文件名。
	[Offset] 从该 mma heap 的哪个 offset 开始 dump data,必须 4KB 对齐
	[Length] 在该 mma heap 里 dump 的总的数据量的大小,必须 4KB 对齐
	[Num] mma head 对应的数字，目前 mma_heap_name0 可通过 cat /proc/cmdline 查看
举例	echo /mnt/ 0 4096 > /proc/mi_modules/mi_sys_mma/mma_heap_name0 在 /mnt 下产生 mma__mma_heap_name0__0__4096.bin 文件

2.7. vb_pool_qlobal

2.7.1 cat

【调试信息】

cat /proc/mi_modules/mi_sys_mma/vb_pool_global

```
-------start of vbpool  dump attr--------
global vb pool info:
collection_size 0x5
-------start of vbpool_allocator info--------


vbpool_allocator info:
     each_allocation_size   total_allocation_count    total_not_used_allocation_count   kern_map_ptr
                     1000                         3                                 3          (null)

vbpool_allocation info:


   offset_in_vb_pool real_used_flag    u64Pts u64SidebandMsg   bEndOfStream   bUsrBuf       eBufType
                   0              0        NA             NA             NA        NA             NA


   offset_in_vb_pool real_used_flag    u64Pts u64SidebandMsg   bEndOfStream   bUsrBuf       eBufType
                1000              0        NA             NA             NA        NA             NA


   offset_in_vb_pool real_used_flag    u64Pts u64SidebandMsg   bEndOfStream   bUsrBuf       eBufType
                2000              0        NA             NA             NA        NA             NA
-------end of vbpool_allocator info--------

-------start of vbpool_allocator info--------
```

Figure 4: 图 2-4

【调试信息分析】

如果有通过 MI_SYS_IMPL_ConfGloPubPools API 或者 MI_SYS_ConfGloPubPools API 配置了 global public pools 的话，会有/proc/mi_modules/mi_sys_mma/vb_pool_global 文件；如果没有配置的话，那么没有该文件。

【调试信息分析】

Table 10: 表 2-9

参数		描述
	collection_size	该 pools 的 allocator 个数
其中某个 vbpool_allocator 的	each_allocation_size	该 allocator 里的 allocation 的

info		size（同一个 vbpool allocator 的不同 allocation 的 size 是相同的，同一个 pools 里的不同的 allocator 的不同的 allocations 的 size 可能不同）。
	total_allocation_count	该 allocator 里的 allocations 的数目。
	total_not_used_allocation_count	该 allocator 里的 allocations 中未被使用的数目。
	kern_map_ptr	该 allocator 的 offset0 位置的 kernel space va；如果没有的话，该值为 NULL。
指定的 vbpool_allocator 的某个 vbpool_allocation 的 info	offset_in_vb_pool	该 allocation 在对应的 allocator 里的逻辑 offset。
	real_used_flag	是否被使用中，否的话则后续参数为无效值 NA。
	u64Pts	如果有效，该值为 pts。
	u64SidebandMsg	如果有效，该值为 SidebandMsg。
	bEndOfStream	如果有效，该值为是否是 end stream 的 flag。
	bUsrBuf	如果有效，该值为表示是否是 UsrBuf。
	eBufType	如果有效，该值表示 buf 的 type。

2.7.2 echo

Table 11: 表 2-10

功能	即 dump 指定 offset 和 length 的 data 到指定的路径
命令	echo [Path] [Offset] [Length] > /proc/mi_modules/mi_sys_mma/ vb_pool_global
参数说明	[Path] 文件要保存的路径,只需提供路径，不要提供具体的文件名，系统会根据参数自动生成对应的文件名。
	[Offset] 从该 pools 的哪个 offset 开始 dump data,是 pools 的逻辑 offset。
	[Length] 在该 pools 里 dump 的总的数据量的大小。
举例	/ # echo /mnt/ 0 1024 > /proc/mi_modules/mi_sys_mma/vb_pool_global 在 /mnt 下产生 vb_pool__global__0__1024.bin

2.8. vb_pool

2.8.1 cat

【调试信息】

cat /proc/mi_modules/mi_sys_mma/vb_pool_mi_divp0

```
-------start of vbpool  dump attr--------
[eModuleId 0xc      u32DevId 0x0] vb pool info:
collection_size 0x5
-------start of vbpool_allocator info--------


vbpool_allocator info:
     each_allocation_size   total_allocation_count    total_not_used_allocation_count   kern_map_ptr
                     1000                         3                                 3         (null)

vbpool_allocation info:


   offset_in_vb_pool real_used_flag    u64Pts u64SidebandMsg   bEndOfStream   bUsrBuf       eBufType
                   0              0        NA             NA             NA        NA             NA


   offset_in_vb_pool real_used_flag    u64Pts u64SidebandMsg   bEndOfStream   bUsrBuf       eBufType
                1000              0        NA             NA             NA        NA             NA


   offset_in_vb_pool real_used_flag    u64Pts u64SidebandMsg   bEndOfStream   bUsrBuf       eBufType
                2000              0        NA             NA             NA        NA             NA
-------end of vbpool_allocator info--------

-------start of vbpool_allocator info--------
```

Figure 4: 图 2-4

【调试信息分析】

如果有 config 某个 device 的 vb pool，那么会有/proc/mi_modules/mi_sys_mma/vb_pool_modnamedevid 文件,其中 modname 形如 divp,vpe 等，devid 形如 0,1 等。其和 vb_pool_global 的差别在于其对应的是该 device,而 vb_pool_global 则是全局共享的 vb pool。

上面是以 vb_pool_mi_divp0 为例，其也有 cat 命令显示各种信息，也是有且只有”echo path offset length >”这个 echo 命令来 dump 数据，这 2 个都和 vb_pool_global 的类似。

2.8.2 echo

Table 12: 表 2-11

功能	即 dump 指定 offset 和 length 的 data 到指定的路径
命令	echo [Path] [Offset] [Length] > /proc/mi_modules/mi_sys_mma/[ModID]
参数说明	[Path] 文件要保存的路径,只需提供路径，不要提供具体的文件名，系统会根据参数自动生成对应的文件名。
	[Offset] 从该 pools 的哪个 offset 开始 dump data,是 pools 的逻辑 offset。
	[Length] 在该 pools 里 dump 的总的数据量的大小。
	[ModID] vb_pool 的文件名， 如 vb_pool_mi_divp0
举例	echo /mnt/ 0 4096 > /proc/mi_modules/mi_sys_mma/vb_pool_mi_divp0 在/mnt/下 产生 vb_pool__mi_divp0__0__4096.bin

2.9. meta

2.9.1 cat

【调试信息】

cat /proc/mi_modules/mi_sys_mma/meta

```
==================   start meta_info   ================================
basic info:
total page count:16          each meta data size:256      total_count_with_metadatasize   256
total_free_count_with_metadatasize:254
meta info :
ref_cnt=3
each allocation info:
offset_in_pool     length      phy_addr          va_in_kern          real_used_flag
0x0                0x100          0x2336000        c1736000              0
0x100              0x100          0x2336100        c1736100              0
0x200              0x100          0x2336200        c1736200              0
0x300              0x100          0x2336300        c1736300              0
0x400              0x100          0x2336400        c1736400              0
0x500              0x100          0x2336500        c1736500              0
0x600              0x100          0x2336600        c1736600              0
0x700              0x100          0x2336700        c1736700              0
0x800              0x100          0x2336800        c1736800              0
0x900              0x100          0x2336900        c1736900              0
0xa00              0x100          0x2336a00        c1736a00              0
……
==================   end meta_info   ================================
```

【调试信息分析】

只要一旦有人用过了 meta allocator(有且只有一个 meta allocator),那么就会生成 /proc/mi_modules/mi_sys_mma/meta 文件，且其对应的 memory 就会完全 alloc 了出来，等待被使用。

2.9.2 echo

Table 13: 表 2-12

功能	即 dump 指定 offset 和 length 的 data 到指定的路径
命令	echo [Path] [Offset] [Length] > /proc/mi_modules/mi_sys_mma/ vb_pool_global
参数说明	[Path] 文件要保存的路径,只需提供路径，不要提供具体的文件名，系统会根据参数自动生成对应的文件名。
	[Offset] 从该 allocator 的哪个 offset 开始 dump data,是 allocator 的逻辑 offset。。

	在该 allocator 里 dump 的总的数据量的大小。
举例	echo /mt 0 1024 > /proc/mi_modules/mi_sys_mma/meta 在 /mnt 下产生 meta__0__1024.bin

2.10. mi_modulenamedevid

2.10.1 cat

【调试信息】

cat /proc/mi_modules/mi_disp/mi_disp0

Figure 5: 图 2-5

【调试信息分析】

cat 操作的结果是分为 2 部分的，第一部分是 MI_SYS 提供的针对该 device 的通用的调试信息，从 device、channel、input port、output port 四个角度提供；第二部分是该 device 私有的信息。私有的信息各 module 的内容别处会阐述，这里 SYS 角度只阐述通用的信息。

以 cat /proc/mi_modules/mi_vdisp/mi_vdisp0 为例，得到上面的结果。

【参数说明】

Table 14: 表 2-13

参数		旧版本	新版本	描述
Common info for device	ChnNum	y	y	该 device 的总的 channel 的数目。
	EnChnNum	y	y	该 device 的 enabled 了的 channel 的数目。
	InPortNum	y	y	该 device 的 input port 的数目。
	OutPortNum	y	y	该 device 的 output port 的数目。
	CollectSize	y	y	该 dev 对应的 AllocatorCollection 含有的 allocator 的数目总和。
Common info for channel(only dump enabled channel)	ChnId	y	y	Channel 的 id 值.
	EnInPNum	y	y	该 channel 的 enabled input port 数量。
	EnOutPNum	y	y	该 channel 的 enabled output port 数量。

	MMAHeapName	y	y	如果有 SetChnMMAConf 的话，该值为对应的 mma heap name;如果没有设置的话，该值为 NULL。
Input port common info(only dump enabled Input port)	ChnId	y	y	该 input port 所在的 channel id。
	PortId	y	y	该 input port 的 id。
	SrcFrmrate	y	y	Src 帧率。
	DstFrmrate	y	y	Dst 帧率。
	user_buf_quota	y	y	该 InputPort 的 buff 数目的 Quota。
	UsrInjectQ_cnt	y	y	该 InputPort 里的 UsrInjectBufQueue 里的 buff 数目。
	UsrInjectQ_size	y	n	该 InputPort 里的 UsrInjectBufQueue 里的 buff 的总的 size。
	BindInQ_cnt	y	y	该 InputPort 里的 BindInputBufQueue 里的 buff 数目。
	BindInQ_size	y	n	该 InputPort 里的 BindInputBufQueue 里的 buff 的总的 size。
	WorkingQ_cnt	y	n	该 InputPort 里的 WorkingQueue 里的 buff 数目。
	WorkingQ_size	y	n	该 InputPort 里的 WorkingQueue 里的 buff 的总的 size。
	TotalPendingBuf_size	n	y	该 InputPort 里的 cur_working_input_queue 里的 buff 的总的 size。
	usrLockedInjectCnt	y	y	用户当前拿到了多少个 buf。
	newPulseQ_cnt	n	y	该 InputPort 里的 new_pulse_fifo_inputqueue 里的 buff 数目。
	nextTodoPulseQ_cnt	n	y	该 InputPort 里的 next_todo_pulse_inputqueue 里的 buff 数目。
	curWorkingQ_cnt	n	y	该 InputPort 里的 cur_working_input_queue 里的 buff 数目。
	workingTask_cnt	n	y	该 InputPort 里的

				input_working_tasklist 里的 buff 数目。
	lazzyRewindTask_cnt	n	y	该 InputPort 里的 lazzy_rewind_inputtask_list 里的 buff 数目。
Input port bind info(only dump enabled Input port)	ChnId	y	y	该 input port 所在的 channel id。
	PortId	y	y	该 input port 的 id。
	bind_module_id	y	y	与该 input port 进行了 binded 的 output port 所在的 module 的 id。
	bind_module_name	y	y	与该 input port 进行了 binded 的 output port 所在的 module 的 name。
	bind_ChnId	y	y	与该 input port 进行了 binded 的 output port 所在的 channel 的 id。
	bind_PortId	y	y	与该 input port 进行了 binded 的 output port 的 id。
Output port common info (only dump enabled Output port)	ChnId	y	y	该 output port 所在的 channel 的 id。
	PortId	y	y	该 output port 的 id。
	BufCntQuota	y	y	该 OutputPort 的 buff 数目的 Quota。
	usrLockedCnt	y	y	从 UsrGetFifoBufQueue 里用户实际拿走了多少个 buffer。
	totalOutPortInUsed	y	y	totalOutputPortInUsedBuf 数目。
	DrvBkRefFifoQ_cnt	y	y	该 OutputPort 的 DrvBkRefFifoQueue 里的 buffer 数目。
	DrvBkRefFifoQ_size	y	y	该 OutputPort 的 DrvBkRefFifoQueue 里的 buffer 的总 size。
	UsrGetFifoQ_cnt	y	y	该 OutputPort 的 UsrGetFifoBufQueue 里的 buffer 数目。
	UsrGetFifoQ_size	y	y	该 OutputPort 的 UsrGetFifoBufQueue 里的 buffer 的总 size。
	WorkingQ_cnt	y	n	该 OutputPort 的 WorkingQueue 里的 buffer 数目。
	WorkingQ_size	y	n	该 OutputPort 的 WorkingQueue

				里的 buffer 的总 size。
	UsrGetFifoQ_seqnum	n	y	该 OutputPort 的 UsrGetFifoBufQueue 里的 buffer 的累计获取数目。
	UsrGetFifoQ_discardnum	n	y	该 OutputPort 的 UsrGetFifoBufQueue 里的 buffer 的累计丢弃数目。
Output port BindPeerInputPortList info(only dump enabled Output port)	ChnId	y	y	该 output port 所在的 channel 的 id。
	PortId	y	y	该 output port 的 id。
	bind_module_id	y	y	与该 output port 进行了 binded 的 input port list 中的其中一个 input port（以下简称该 binded input port）所在的 module 的 id。
	bind_module_name	y	y	该 binded input port 所在的 module 的 name。
	bind_ChnId	y	y	该 binded input port 所在的 channel id。
	bind_PortId	y	y	该 binded input port 的 id。

2.10.2 echo

Table 15: 表 2-14

功能	获得模块的 help 信息
命令	echo help > /proc/mi_modules/[ModName]/[ModID]
参数说明	[ModName] 模块的名字 mi_disp mi_gfx mi_rgn mi_vdec mi_vpe mi_ai mi_divp mi_shadow mi_vdisp mi_ao mi_hdmi mi_sys mi_venc mi_bar mi_vif
	[ModID] 模块中对应的文件 一般为 一般为模块的名字+数字 如 mi_disp0 mi_gfx0 mi_rgn0
举例	echo help > /proc/mi_modules/mi_disp/mi_disp0 得到 mi_disp0 节点文件的帮助信息

Table 16: 表 2-15

功能	Dump Port 的信息
命令	echo dump_buffer [ChnID] [PortType] [PortID] [QueueName][Path] [EndMethod] > /proc/mi_modules/[ModName]/[ModID]
	[ChnID] 通道的 ID

参数说明	[PortType] iport 或 oport 分别代表 input port output port
	[PortID] 该 port 的 id
	[QueueName] 如果是 input port 的话，值只能是"UsrInject" 或者"BindInput" 如果是 output port 的话，值只能是"UsrGetFifo"
	[Path] 数据要保存的文件所在的路径。注意是绝对路径，不要提供数据要保存的文件的名字，系统会根据参数自动生成文件名，不同的 buffer 保存在不同的文件
	[EndMethod] dump buffer 的结束方法，目前仅支持三类： (a). "bufnum=xxx":dump 指定数目的 buffer (b) "time=xxx"(here unit is ms):dump 过程持续 time 对应的时间 (c). "start/end" pair:用 start 来开始 dump,用 end 来结束对应的 dump,要求这 2 个命令时其他的 5 个参数完全一样
	[ModName] 模块的名字 mi_disp mi_gfx mi_rgn mi_vdec mi_vpe mi_ai mi_divp mi_shadow mi_vdisp mi_ao mi_hdmi mi_sys mi_venc mi_bar mi_vif
	[ModID] 模块中对应的文件 一般为 一般为模块的名字+数字
举例	echo dump_buffer 0 iport 0 BindInput /mnt bufnum=1 > /proc/mi_modules/mi_disp/mi_disp0 dump disp 模块 inport 的数据

2.11. mi_dump_buffer_delay_worker

2.11.1 cat

【调试信息】

cat /proc/mi_modules/mi_dump_buffer_delay_worker

delay_worker_id module_name force_stop dev_id chn_id port_type port_id
0 mi_disp 0 0 0 inport 0
Queue_name stored_dir dump_method dump_method_value
BindInput /mnt bunfnum 2

【调试信息分析】

/proc/mi_modules/mi_dump_buffer_delay_worker 文件是和 mi_modulenamedevid 相连的一个概念，dump device 里的 Queue 里的 buffer 采用的是 delay worker 的方式实现的，可以通过对 mi_dump_buffer_delay_worker 的 cat 操作查看还有哪些 delay worker 正在进行中，echo force_stop_dump delay_worker_id 强制结束指定的一个 delay worker 过程。

【参数说明】

Table 17: 表 2-16

参数	描述

delay_worker_id	该 delay worker 的 id。如果 delay worker 完成的话，该 id 会回收，给后续其他的 delay worker 作为 id 使用。
module_name	该 delay worker 所对应的 module 的 id。
force_stop	该 delay worker 当前是否处于强制结束阶段，1 表示已经被设置了强制结束，0 表示未被设置。
dev_id	该 delay worker 所对应的 device id。
chn_id	该 delay worker 所对应的 channel id。
port_type	该 delay worker 所对应的 port 的 type,是 input port,还是 output port
port_id	该 delay worker 所对应的 port 的 id
Queue_name	该 delay worker 所对应的 Queue 的 name
stored_dir	该 delay worker 所对应的要 dump 的 data 所要存放的绝对路径（不含最终存放的文件名）
dump_method	该 delay worker 所对应的 dump 方法:bufnum，还是 time,还是 start
dump_method_value	Dump 方法所对应的值.

2.11.2 echo

Table 18: 表 2-17

功能	强制结束 delay worker
命令	echo force_stop_dump [WorkID] >/proc/mi_modules/mi_dump_buffer_delay_worker
参数说明	[WorkID] delay_work 的 ID
举例	echo force_stop_dump 0 > /proc/mi_modules/mi_dump_buffer_delay_worker 强制停止 delay_work 0

2.12. module version

2.12.1 cat

【调试信息】

cat /proc/mi_modules/mi_ai/module_version_file

MStar Module version: project_commit.sdk_commit.build_time c1799df.fd2d52b.20171225180033

cat /proc/mi_modules/mi_global_info

.....

MStar Module version: project_commit.sdk_commit.build_time cb68bfd.44aca45.20171226100257
.....

【调试信息分析】
xxxx_version_file 提供了份 version 信息， /proc/mi_modules/mi_global_info 里我们也提供了份 version 信息，不过其本质上指的是 mi_sys 模块的 version 信息，上面只是以 mi_vi 和 mi_global_info 为例子，每个模块都有自己对应的 version file。

【参数说明】
Table 19: 表 2-18

参数		描述
Version Info	project_commit	模块编译 ko 时整包的 project commit 信息，如果单独替换 ko 时，模块基于的 commit 有变化，那么该 ko 的 version 里得到的 commit 也会跟着变更
	sdk_commit	模块编译 ko 时整包的 sdk commit 信息，如果单独替换 ko 时，模块基于的 commit 有变化，那么该 ko 的 version 里得到的 commit 也会跟着变更
	build_time	时间指的是实际的 build 时间，精确到秒，即使 make clean;make image 整体来 build 的话，各个 ko 的时间也会有差别。

3. AI

3.1. cat

【调试信息】
cat proc/mi_modules/mi_ai/mi_ai0

==========Private AI0 Info==========
-----Start AI Dev0 Attr----------
AiDev WorkMode SampR BitWidth SondMod PtNumPerFrm TotalReadFrmCnt
0 i2s-mas 8000 16bit mono 1024 135
I2sMclk I2sFmt bI2sSync
disable I2S-MSB 0

-----Start AI CHN0 STATUS----------
AiDev AiChn bReSmp InSampR OutSampR ReadFrmCnt
0 0 0 8000 0 135
-----AI CHN0 VQE STATUS----------
AiDev AiChn bVqe
0 0 0
-----AI CHN0 ANR STATUS----------
AiDev AiChn bAnr bUsr Speed Intensity SmoothLevel
0 0 0 0 speed-low 0 0
-----AI CHN0 Eq TATUS----------
AiDev AiChn bEq bUsr 100Hz 200Hz 250Hz 350Hz 500Hz 800Hz 1.2KHz 2.5KHz 4KHz
8KHz
0 0 0 0 0 0 0 0 0 0 0 0 0 0
-----AI CHN0 Hpf STATUS----------
AiDev AiChn bHpf bUsr HpfFreq
0 0 0 0 0
-----AI CHN0 Agc STATUS----------
AiDev AiChn bAgc bUsr AttackTime ReleaseTime CompressionRatio DropGainMax
0 0 0 0 0 0 0 0
GainInit GainMin GainMax NoiseGateAttenuationDb NoiseGateDb TargetLevelDb
0 0 0 0 0 0
-----AI CHN0 AEC STATUS----------
AiDev AiChn bAec bNoise
0 0 0 0

```
Aec freq:    0    0    0    0    0    0
Aec Intensity:    0    0    0    0    0    0    0
-----AI CHN0 Aenc STATUS----------------------------------------------------
AiDev   AiChn   bAenc   AencType
    0       0       0      g711a
-----End AI Chn0 STATUS-----------------------------------------------------------
```

【调试信息分析】

记录当前 AI 的使用状况以及 device 属性、channel 属性，可以动态地获取到这些信息，方便调试和测试。

【参数说明】

Table 20: 表 3-1

参数		描述
AI Device Attr (AI 设备属性)	AiDev	AI 设备号
	WorkMode	AI I2S 工作模式(K6/K6L 不支持) i2s-mas: I2S master i2s-sla: I2S slave tdm-mas: TDM master
	SamR	Sample Rate(采样率): 8000, 16000, 32000, 48000(I6 Dmic 不支持 48k)
	BitWidth	采样精度: 16bit, 24bit(暂不支持)
	SondMod	声音模式 mono：单声道 stereo：立体声 Queue：一个通道包含所有通道的数据
	PtNumPerFrm	每帧的采样点个数: 128 倍数(I6 上每帧数据的采样点 >= PtNumPerFrm)
	TotalReadFrmCnt	当前 Dev 获取到的总帧数
	I2sMclk	I2S 设定的 Mclk 时钟(只有 I2,I5 支持 Mclk 设定)
	I2sFmt	I2S 的数据格式
	bI2sSync	I2S Rx Bclk 和 Tx Bclk 使用同样的时钟源

Table 21: 表 3-2

参数		描述
AI Chn Status (AI 通道信息)	AiDev	AI 设备号
	AiChn	AI 通道号: 取值范围：[0~ChnCnt](ChnCnt 为设置的 Dev 最大通道数)
	bReSmp	该信道是否启用了重采样功能 1: Enable

		0: Disable
	InSampR	该通道重采样时，输入给重采样的数据帧的采样率
	OutSampR	该通道重采样时，重采样后的音频帧采样率
	ReadFrmCnt	该通道读取的帧数
AI CHN VQE STATUS (AI 通道 VQE 信息)	AiDev	AI 设备号
	AiChn	AI 通道号: 取值范围：[0~ChnCnt](ChnCnt 为设置的 Dev 最大通道数)
	bVqe	该通道是否启用 VQE 1: Enable 0: Disable
AI CHN ANR of VQE STATUS (AI 通道 ANR 信息)	AiDev	AI 设备号
	AiChn	AI 通道号: 取值范围：[0~ChnCnt](ChnCnt 为设置的 Dev 最大通道数)
	bAnr	该通道是否启用 ANR 1: Enable 0: Disable
	bUsr	Anr 算法运行的模式
	Speed	噪声收敛速度，低速，中速，高速
	NrIntensity	降噪力度配置: 取值为[0,30]
	SmoothLevel	平滑化程度 取值为[0,10]
AI CHN AGC of VQE STATUS (AI 通道 AGC 信息)	AiDev	AI 设备号
	AiChn	AI 通道号: 取值范围：[0~ChnCnt](ChnCnt 为设置的 Dev 最大通道数)
	bAgc	该通道是否启用 AGC 1: Enable 0: Disable
	bUsr	Agc 算法运行的模式
	AttackTime	增益下降时间区间长度，以 16 毫秒为 1 单位 范围[1, 20]
	ReleaseTime	增益上升时间区间长度，以 16 毫秒为 1 单位 范围[1, 20]
	CompressionRatio	输入和输出之间的相关性，值越大目标增益变化越大 范围[1,10]
	DropGainMax	增益下降的最大值，防止输出饱和 范围[0,60]
	GainInit	增益最小值 范围[-20, 30]
	GainMin	增益中间值

		范围[-20, 30]
	GainMax	增益最大值 范围[0, 30]
	NoiseGateAttenuationDb	当噪声底值起效果时，输入源的衰减百分比 范围[0, 100]
	NoiseGateDb	噪声底值 范围[-80, 0]
	TargetLevelDb	目标电平，经过处理后的最大电平门限 范围[-80, 0]dB
AI CHN EQ of VQE STATUS (AI 通道 EQ 信息)	AiDev	AI 设备号
	AiChn	AI 通道号: 取值范围：[0~ChnCnt](ChnCnt 为设置的 Dev 最大通道数)
	bEq	该通道是否启用 EQ 1: Enable 0: Disable
	bUsr	EQ 算法的运行模式
	100Hz	100Hz 频段增益取值 范围[-50, 20]
	200Hz	200Hz 频段增益取值 范围[-50, 20]
	250Hz	250Hz 频段增益取值 范围[-50, 20]
	350Hz	350Hz 频段增益取值 范围[-50, 20]
	500Hz	500Hz 频段增益取值 范围[-50, 20]
	800Hz	800Hz 频段增益取值 范围[-50, 20]
	1.2KHz	1200Hz 频段增益取值 范围[-50, 20]
	2.5KHz	2500Hz 频段增益取值 范围[-50, 20]
	4KHz	4000Hz 频段增益取值 范围[-50, 20]
	8KHz	8000Hz 频段增益取值 范围[-50, 20]
AI CHN0 HPF of VQE STATUS	AiDev	AI 设备号
	AiChn	AI 通道号: 取值范围：[0~ChnCnt](ChnCnt 为设置的 Dev 最大通道数)
	bHpf	该通道是否启用 HPF 1: Enable

		0: Disable
	bUsr	Hpf 算法的运行模式
	HpfFreq	截止频率（Hz）： 80, 120, 150
AI CHN AEC of VQE STATUS (AI 通道 AEC 信息)	AiDev	AI 设备号
	AiChn	AI 通道号: 取值范围：[0~ChnCnt](ChnCnt 为设置的 Dev 最大通道数)
	bAec	该通道是否启用 AEC 1: Enable 0: Disable
	bNoise	是否添加噪声
	AecSupfreq	回声消除保护频率范围 范围[1, 127]
	AecSupIntensity	回音消除保护力度 范围[0, 15]
AI CHN AENC STATUS (AI 通道 AENC 信息)	AiDev	AI 设备号
	AiChn	AI 通道号: 取值范围：[0~ChnCnt](ChnCnt 为设置的 Dev 最大通道数)
	bAenc	该通道是否启用 AENC 1: Enable 0: Disable
	AencType	编码类型

4. AO

4.1. cat

【调试信息】

```
# cat proc/mi_modules/mi_ao/mi_ao0

=======================================Private AO0 Info
=====================================
-----Start AO Dev0 Attr-----------------------------------------------------------------------
AoDev   WorkMode   SampR   BitWidth   SondMod   PtNumPerFrm
     0    i2s-mas    8000       16bit       mono            1024
I2sMclk      I2sFmt   bI2sSync   bMute   VolumeDb
disable    I2S-MSB            0       0            0
-----End AO Dev0 Attr-------------------------------------------------------------------------

-----Start AO CHN0 STATUS---------------------------------------------------------------------
AoDev   AoChn   bReSmp   InSampR   OutSampR
     0       0        0         0       8000
AoDev   AoChn      TotalFrmCnt        TotalSize           RunTime
     0       0               88           180224                12
-----Start AO CHN0 Usr Queue STATUS-----------------------------------------------------------
AoDev   AoChn   MaxSize   RemainSize        TotalSize           RunTime
     0       0     25600              0           180224                12
-----AO CHN0 Vqe STATUS-----------------------------------------------------------------------
AoDev   AoChn   bVqe
     0       0      0            0         0
-----AO CHN0 Anr STATUS-----------------------------------------------------------------------
AoDev   AoChn   bAnr   bUsr          Speed   Intensity   SmoothLevel
     0       0      0      0    speed-low             0             0
-----AO CHN0 Eq TATUS-------------------------------------------------------------------------
AoDev   AoChn   bEq   bUsr   100Hz   200Hz   250Hz   350Hz   500Hz   800Hz   1.2KHz   2.5KHz   4KHz
8KHz
     0       0     0      0       0       0       0       0       0       0        0        0      0       0
-----AO CHN0 Hpf STATUS-----------------------------------------------------------------------
AoDev   AoChn   bHpf   bUsr   HpfFreq
     0       0      0      0          0
-----AO CHN0 Agc STATUS-----------------------------------------------------------------------
AoDev   AoChn   bAgc   bUsr   AttackTime   ReleaseTime   CompressionRatio   DropGainMax
     0       0      0      0             0             0                  0             0
```

```
GainInit   GainMin   GainMax   NoiseGateAttenuationDb   NoiseGateDb   TargetLevelDb
        0          0          0                        0             0               0
-----AO CHN0 Adec STATUS-------------------------------------------------------------------------
AoDev   AoChn   bAdec      Type      SondMod      SampR
    0       0       0     g711a         mono          0
-----End AO Chn0 STATUS---------------------------------------------------------------------------
```

【调试信息分析】

记录当前 AO 的使用状况以及 device 属性、channel 属性，可以动态地获取到这些信息，方便调试和测试。

【参数说明】

Table 23: 表 4-1

参数		描述
AO Device Attr (AO 设备属性)	AoDev	AO 设备号
	WorkMode	AO I2S 工作模式 i2s-mas: I2S master i2s-sla: I2S slave tdm-mas: TDM master
	SamR	Sample Rate(采样率): 8000, 16000, 32000, 48000
	BitWidth	采样精度: 16bit, 24bit(暂不支持)
	SondMod	声音模式 mono：单声道 stereo：立体声
	PtNumPerFrm	每帧的采样点个数: 128 倍数
	I2sMclk	I2S Mclk 时钟频率(只有 I2,I5 支持)
	I2sFmt	I2S 数据格式
	bI2sSync	I2S Rx Bclk 和 Tx Bclk 使用同样的时钟源
	bMute	AO 设备静音功能是否开 1: Enable 0: Disable
	VolumedB	AO 设备的输出音量大小(单位 dB)

Table 24: 表 4-2

参数		描述
	AoDev	AO 设备号
	AoChn	AO 通道号:

AO CHN Status (AO 通道信息)		取值范围：[0,1]
	bReSmp	该信道是否启用了重采样功能 1: Enable 0: Disable
	InSampR	该通道重采样时，输入给重采样的数据帧的采样率
	OutSampR	该通道重采样时，重采样后的音频帧采样率
	TotalFrmCnt	AO 目前输出的帧数
	TotalSize	AO 目前输出的数据量
	RunTime	AO 目前运行的时间(ns)
AO CHN USER QUEUE STATUS (AO 通道 Queue 信息)	AoDev	AO 设备号
	AoChn	AO 通道号: 取值范围：[0,1]
	MaxSize	Queue 的最大 size
	RemainSize	Queue 的剩余 size
AO CHN VQE STATUS (AO 通道 VQE 信息)	AoDev	AO 设备号
	AoChn	AO 通道号: 取值范围：[0,1]
	bVqe	该通道是否启用 VQE 1: Enable 0: Disable
AO CHN ANR of VQE STATUS (AO 通道 ANR 信息)	AoDev	AO 设备号
	AoChn	AO 通道号: 取值范围：[0,1]
	bAnr	该通道是否启用 ANR 1: Enable 0: Disable
	bUsr	Anr 算法运行的模式
	Speed	噪声收敛速度，低速，中速，高速
	NrIntensity	降噪力度配置: 取值为[0,30]
	SmoothLevel	平滑化程度 取值为[0,10]
AO CHN AGC of VQE STATUS (AO 通道 AGC 信息)	AoDev	AO 设备号
	AoChn	AO 通道号: 取值范围：[0,1]
	bAgc	该通道是否启用 AGC 1: Enable 0: Disable

	bUsr	Agc 算法运行的模式
	AttackTime	增益下降时间区间长度，以 16 毫秒为 1 单位 范围[1, 20]
	ReleaseTime	增益上升时间区间长度，以 16 毫秒为 1 单位 范围[1, 20]
	CompressionRatio	输入和输出之间的相关性，值越大目标增益变化越大 范围[1,10]
	DropGainMax	增益下降的最大值，防止输出饱和 范围[0,60]
	GainInit	增益最小值 范围[-20, 30]
	GainMin	增益中间值 范围[-20, 30]
	GainMax	增益最大值 范围[0, 30]
	NoiseGateAttenuationDb	当噪声底值起效果时，输入源的衰减百分比 范围[0, 100]
	NoiseGateDb	噪声底值 范围[-80, 0]
	TargetLevelDb	目标电平，经过处理后的最大电平门限 范围[-80, 0]dB
AO CHN EQ of VQE STATUS (AO 通道 EQ 信息)	AoDev	AO 设备号
	AoChn	AO 通道号: 取值范围：[0,1]
	bEq	该通道是否启用 EQ 1: Enable 0: Disable
	bUsr	EQ 算法的运行模式
	100Hz	100Hz 频段增益取值 范围[-50, 20]
	200Hz	200Hz 频段增益取值 范围[-50, 20]
	250Hz	250Hz 频段增益取值 范围[-50, 20]
	350Hz	350Hz 频段增益取值 范围[-50, 20]
	500Hz	500Hz 频段增益取值 范围[-50, 20]
	800Hz	800Hz 频段增益取值 范围[-50, 20]
	1.2KHz	1200Hz 频段增益取值 范围[-50, 20]

	2.5KHz	2500Hz 频段增益取值 范围[-50, 20]
	4KHz	4000Hz 频段增益取值 范围[-50, 20]
	8KHz	8000Hz 频段增益取值 范围[-50, 20]
AO CHN0 HPF of VQE STATUS	AoDev	AO 设备号
	AoChn	AO 通道号: 取值范围：[0,1]
	bHpf	该通道是否启用 HPF 1: Enable 0: Disable
	bUsr	Hpf 算法的运行模式
	HpfFreq	截止频率（Hz）: 80, 120, 150

4.2. echo

Table 25: 表 4-3

功能	动态启用/关闭 AO 设备静音模式
命令	echo setaomute [Status] > proc/mi_modules/mi_ao/mi_ao[ID]
参数说明	[Status] ON 开启静音模式 OFF 关闭静音模式
	[ID] 设备号
举例	无
功能	动态修改 AO volume 大小
命令	echo setaovolume [Steps] > proc/mi_modules/mi_ao/mi_ao[ID]
参数说明	[Steps] 调整步长 [n]dB
	[ID] 设备号
举例	无

5. SNR

5.1. cat

【调试信息】

cat proc/mi_modules/mi_sensor/mi_sensor0

```
------------------------- start dump Pad info -------------------------------
  PadId  PlaneMode  bEnable  bmirror  bflip  fps  ResCnt  intfmode  hdrmode  planecnt
     0          0        1        0      025000       1      MIPI        2         2
      MipiAttr  YuvOrder  HdrHwMode  DataFmt  HdrVchNum  HsyncMode  LaneNum LPackType0 LPackType1  samDelay
                     0          0        0          0          0         0          0          0          0
      Res        strResDesc  CropX  CropY  CropW  CropH  OutW  OutH  MaxFps  MinFps
           1920x1080@30fps      0      0   1920   1080  1948  1097      30       3
      Cur  1920x1080@30fps      0      0   1920   1080  1948  1097      30       3
------------------------- End dump Pad  info -------------------------------

------------------------- start dump plane info -------------------------------
  Padid  Planeid            SnrName  BayerId  ePixPrec  eHdrSrc  CropX  CropY  CropW  CropH
     0        0         IMX307_MIPI       RG     12BPP        0      0      0   1920   1080
     0        0         IMX307_MIPI       RG     12BPP        0      0      0   1920   1080
```

【调试信息分析】

记录当前 VIF 的使用状况以及 device 属性、OutPort 属性、可以动态地获取到这些信息，方便调试和测试。

【参数说明】

Table 25: 表 5-1

参数		描述
PAD info	PadId	Pad Id 号
	PlaneMode	Plane mode(0:非 HDR, 1:HDR)
	bEnable	Pad 使能开关
	bmirror	垂直镜像翻转使能
	bflip	水平镜像翻转使能
	fps	Sensor 帧率
	ResCnt	支持 resolution 数量
	intfmode	Pad 接口
	hdrmode	使用 HDR 的模式
	planecnt	支持 plane 通道数量
	Res	支持的 resolution 映射表
	strResDesc	Resolution 字符串

	CropX CropY CropW CropH	从 sensor 原始数据上 crop 的范围
	OutW OutH	Sensor 原始范围
	MaxFps MinFps	当前 resolution 可以设置的最大和最小帧率
	Cur	当前 sensor 输出的 resolution

Table 26: 表 5-2

参数		描述
Plane info	SnrName	当前 plane 使用的 sensor interface
	BayerId	RGB 对齐方式
	ePixPrec	使用的 bit mode
	eHdrSrc	当前 Plane 使用 HDR source 通道
	CropX CropY CropW CropH	Crop 位置

5.2. echo

Table 27: 表 5-3

功能	打印 debug 相关信息
命令	echo help > /proc/mi_modules/mi_sensor/mi_sensor0
参数说明	无
举例	无

Table 28: 表 5-4

功能	设置 sensor mirror/flip
命令	echo setmirrorflip [PadId] [bmirror] [bflip] > /proc/mi_modules/mi_sensor/mi_sensor0
参数说明	无
举例	echo setmirrorflip 0 0 1 > /proc/mi_modules/mi_sensor/mi_sensor0 Pad0 做 flip

Table 29: 表 5-5

功能	设置 sensor 帧率
命令	echo setfps [PadId] [fps] > /proc/mi_modules/mi_sensor/mi_sensor0
参数说明	无
举例	echo setfps 0 20 > /proc/mi_modules/mi_sensor/mi_sensor0 设置 Pad0 输出帧率为 20fps

6. VIF

6.1. cat

【调试信息】

#cat /proc/mi_modules/mi_vif/mi_vif0

```
----------------------------------dump Dev Attr-----------------------------------------------
dev intf work clk hdr
  0    4    3   0   0
-----------------------------------dump outport attr-------------------------------------------
dev chn port cap             dest       sel scan  fmt rate  linecnt  mi_frame_status
  0   0    0 (0,0,1920,1080) (1920,1080) 3    0    40    0    0    (0,0,0,0)

dev chn port rev_size    out_size     sub_out_size  rwd_idx    mhal_frame_status    ring_buf_status
  0   0    0 (   0,   0) (   0,3229425096) (   0,3229425096) (0,0,0) (0,0,1425,0)(0,0,0,0,0,0,0,0)
```

【调试信息分析】

记录当前 VIF 的使用状况以及 device 属性、OutPort 属性、可以动态地获取到这些信息，方便调试和测试。

【参数说明】

Table 30: 表 6-1

参数		描述
Dev info	dev	Dev 号
	intf	输入数据的协议
	work	工作模式
	clk	时钟边沿触发模式
	hdr	Hdr 模式

Table 31: 表 6-2

参数		描述
OutPort Info	cap	Crop 参数
	dest	输出宽高
	sel	隔行扫描时的顶场/底场
	scan	扫描模式
	fmt	Pixelformat 格式
	rate	帧率
	linecnt	Lowlatency mode linecnt 设置

	mi_frame_status	(bufferhold, incnt, outcnt, fps)

7. VPE

7.1. cat

【调试信息】

cat proc/mi_modules/mi_vpe/mi_vpe0

```
------------------------- start dump DEV info -------------------------------
  DevID  3DNR_Status  ROI_Status  SupportIRQ  IRQ_num  IRQ_Enable  DropStatus0  DropFence0  DropStatus1  DropFence1
      0      Idle(0)        Idle           1       27           1            0           0            0           0
  eRunMode   EnIrqMode  UseCmdq  vsyncCnt  WorkCnt  IsrthrCnt  BotthrCnt  Point
  RealTime          af        1     53526   107381     121199      45572   3548
------------------------- End dump dev 0 info -------------------------------

------------------------- start dump CHN info -------------------------------
  ChnId  status      InWH     MaxWH      InPix               Crop    IspIn               HwCrop  bRot  eRot  RunMode
      0   Start  1920x1080  1920x1080   RG_12BPP(    0,    0,1920,1080)1920x1080(    0,    0,1920,1080)      0     0 RealTime
  ChnId  InCnt  InTodoCnt  DropInCnt  InStride  Atom  sclInMode  sclOutMode  HDR  3DnrLevel  SensorId  scl2UsedBy
      0  53628      52892          0         0     2          0         10f    0          3         0         vpe
  ChnId  GetTPo  PoToKo  KoToRe  SwaitDonePre  EwaitdonePre  SwaitDoneCur  EwaitdoneCur
      0      17     732    6786     971417923     971417951     971434723     971434751
  ChnId  bNr  bEdge  bEs  bContrast  bUV  cSfStr  cTfStr  ySfStr  yTfStr  MotionTh  StillTh  MWer  OWei  SWei  Constrast
      0    1      0    0          1    0       0       0       0       0         0        0     0     0          0
  ChnId  Edge0  Edge1  Edge2  Edge3  Edge4  Edge5
      0      0      0      0      0      0      0
------------------------- End dump CHN info -------------------------------

------------------------- start dump OUTPUT PORT info -----------------------
  ChnId  PortId  Enable      Pixel  bMirr/flip          PortCrop  OutputW  OutputH  Stride  Compress  GetCnt  FailCnt  FinishCnt  fps
      0       0       1   YUV420SP(    0,    0)   0,    0,    0     1920     1080    1920         0   52398      883      52396   59.70
      0       1       1   YUV420SP(    0,    0)   0,    0,    0     1280      720    1280         0   51711     1563      51709   57.86
      0       2       0        MAX(    0,    0)   0,    0,    0        0        0       0         0       0        0          0    0.00
      0       3       1   YUV420SP(    0,    0)   0,    0,    0      720      576     736         0   51580     1694      51578   57.78
------------------------- End dump OUTPUT PORT info -------------------------
```

Figure 6: 图 7-1

【调试信息分析】

记录当前 VPE 的使用状况以及 device 属性、channel 属性、output port 属性，可以动态地获取到这些信息，方便调试和测试。

【参数说明】

Table 27: 表 7-1

参数		描述
	DevID	0，只有一个 Device, ID 为 0
	3DNR_Status	chnid 为当前 3DNR 正在处理的 channelID
	ROI_Status	当前 ROI 状态
	SupportIRQ	0：当前不支持 IRQ 1：当前支持 IRQ. (在 procfs 中可以控制该值开关中断)
	IRQ_num	中断号
	IRQ_Enable	0：IRQ 失能 1：IRQ 使能

Device Info	DropStatus0 DropFence0 DropStatus1 DropFence1	Drop buffer 的 status 和对应的 fenceid *SSC323、SSC325、SSC325DE、SSC327、SSC327Q 暂不支持
	EnIrqMode	Irq 使能模式
	UseCmdq	是否使用 cmdq
	vsyncCnt	Vsync isr 计数
	WorkCnt	Work thread 运行计数 *SSC323、SSC325、SSC325DE、SSC327、SSC32 不支持
	IsrthrCnt	Isr thread 运行计数 *SSC323、SSC325、SSC325DE、SSC327、SSC32 不支持
	BotthrCnt	Bottom thread 运行计数 *SSC323、SSC325、SSC325DE、SSC327、SSC32 不支持
	Point	Point 运行到的 line number

Table 28: 表 7-2

参数		描述
Channel Info	ChnId	Channel ID 0~63
	status	Channel 运行状态
	InWH	输入的宽高
	MaxWH	设置 MAX width/height
	InPix	输入 pixel
	Crop	Crop 的位置 *SSC323、SSC325、SSC325DE、SSC327、SSC32 不支持
	IspIn	Isp input 配置 size
	HwCrop	Hardware crop 配置 size *SSC323、SSC325、SSC325DE、SSC327、SSC32 不支持
	bRot	rotation 使能
	eRot	E_MI_SYS_ROTATE_NONE, //Rotate 0 degrees E_MI_SYS_ROTATE_90, //Rotate 90 degrees E_MI_SYS_ROTATE_180, //Rotate 180 degrees E_MI_SYS_ROTATE_270, //Rotate 270 degrees E_MI_SYS_ROTATE_NUM,
	RunMode	Channel 对应的 running mode
	InCnt	接收到 input 的计数
	InTodoCnt	将 input 送到底层的计数
	DropInCnt	主动丢掉 input 的计数
	InStride	Input buffer 的 stride

	Atom	底层拿住的 buffer 数量
	sclInMode	Scale input 配置
	sclOutMode	Scale output 配置
	HDR	HDR 使能模式
	3DnrLevel	3DNR 开启级别
	SensorId	ISP 绑定 sensorid
	scl2UsedBy	Scale2 被使用的模块名 *SSC323、SSC325、SSC325DE、SSC327、SSC32 不支持
	GetTPo	Get buffer 到送给底层的时间
	PoToKo	送给底层到硬件开始处理的时间
	KoToRe	硬件处理到硬件处理结束的时间
	SwaitDonePre EwaitdonePre SwaitDoneCur EwaitdoneCur	Waitdone 函数运行的时间点

Table 29: 表 7-3

参数		描述
Outputport Info	ChnId	Channel ID 0~63
	PortId	0~3
	Enable	0:port disable 1:port enable
	Pixel	Pixel format
	bMirr/flip	垂直/水平翻转使能
	PortCrop	Port crop 位置
	OutputW ， OutputH	输出宽高
	Stride	Output buffer stride
	Compress	E_MI_SYS_COMPRESS_MODE_NONE,//no compress E_MI_SYS_COMPRESS_MODE_SEG,//compress unit is 256 bytes as a segment E_MI_SYS_COMPRESS_MODE_LINE,//compress unit is the whole line E_MI_SYS_COMPRESS_MODE_FRAME,//compress unit is the whole frame E_MI_SYS_COMPRESS_MODE_BUTT, //number
	GetCnt	获取 outputbuffer 的数量
	FailCnt	获取 output buffer 失败的次数
	FinishCnt	处理完 outputbuffer 的数量

	fps	帧率

7.2. echo

Table 30: 表 7-4

功能	控制使用 cmdq 方式，设置下次启动 VPE 的时候生效。
命令	echo disable_cmdq [Status] > /proc/mi_modules/mi_vpe0
参数说明	[Status] ON 下寄存器指令来一个下一个 OFF 使用造剧本的方式下寄存器指令
举例	无

*SSC323、SSC325、SSC325DE、SSC327、SSC32 不支持

Table 31: 表 7-5

功能	控制是否使用 IRQ，设置下次启动 VPE 的时候生效
命令	echo disable_irq [Status] > /proc/mi_modules/mi_vpe0
参数说明	[Status] ON 不使用 IRQ 中断 OFF 使用 IRQ 中断
举例	无

*SSC323、SSC325、SSC325DE、SSC327、SSC32 不支持

Table 32: 表 7-6

功能	打印接收到每一帧的 PTS，Debug Frame 顺序问题，正常情况下 PTS 是递增的
命令	echo checkframepts [ChnID] [Status] > /proc/mi_modules/mi_vpe/mi_vpe0
参数说明	[ChnID] 通道号 [0~63]
	[Status] ON 开始打印 OFF 停止打印
举例	echo checkframepts 1 ON > /proc/mi_modules/mi_vpe/mi_vpe0 [MI VPE PROCFS]:ChnID 1, receive buffer ID = 111, PTS = 31461 [MI VPE PROCFS]:ChnID 1, receive buffer ID = 112, PTS = 31467 [MI VPE PROCFS]:ChnID 1, receive buffer ID = 113, PTS = 31471 [MI VPE PROCFS]:ChnID 1, receive buffer ID = 114, PTS = 31475 [MI VPE PROCFS]:ChnID 1, receive buffer ID = 115, PTS = 31479 [MI VPE PROCFS]:ChnID 1, receive buffer ID = 116, PTS = 31485 [MI VPE PROCFS]:ChnID 1, receive buffer ID = 117, PTS = 31493 recieve buffer ID：每个 channel 接收到 buffer 的 ID。 PTS: FrameBuffer 的 PTS。

Table 31: 表 7-7

功能	Dump channel 的输出 frame
命令	echo dumptaskfile [chnid] [cnt] [path] > /proc/mi_modules/mi_vpe/mi_vpe0
参数说明	无
举例	echo dumptaskfile 0 3 /mnt > /proc/mi_modules/mi_vpe/mi_vpe0

Table 31: 表 7-8

功能	设置旋转角度
命令	echo setroation [chnid] [eRot] > /proc/mi_modules/mi_vpe/mi_vpe0
参数说明	eRot:0->0° 、1->90° 、2->180° 、3->270°
举例	echo setroation 1 1 > /proc/mi_modules/mi_vpe/mi_vpe0

*SSC323、SSC325、SSC325DE、SSC327、SSC32 不支持

Table 31: 表 7-9

功能	查询 cmdq 是否一直都 busy
命令	echo checkcmdq ON/OFF > /proc/mi_modules/mi_vpe/mi_vpe0
参数说明	无
举例	echo checkcmdq ON > /proc/mi_modules/mi_vpe/mi_vpe0 1ms 查询一次， 如果是 idle 将马上打印， 如果连续查询 500 次都是 busy，打印 busy echo checkcmdq OFF > /proc/mi_modules/mi_vpe/mi_vpe0 停止查询

*SSC323、SSC325、SSC325DE、SSC327、SSC32 不支持

Table 31: 表 7-10

功能	设置某个 channel 停止工作
命令	echo stopchnl [chnid] ON/OFF > /proc/mi_modules/mi_vpe/mi_vpe0
参数说明	无
举例	echo stopchnl 0 ON > /proc/mi_modules/mi_vpe/mi_vpe0 停止 channel 0 echo stopchnl 0 OFF > /proc/mi_modules/mi_vpe/mi_vpe0 启动 channel 0

Table 31: 表 7-11

功能	设置某一个 output port crop size
命令	echo setprecrop [chnid] [portid] [X] [Y] [Width] [Height]> /proc/mi_modules/mi_vpe/mi_vpe0
参数说明	无
举例	echo setprecrop [chnid] [portid] [X] [Y] [Width] [Height]> /proc/mi_modules/mi_vpe/mi_vpe0

Table 31: 表 7-12

功能	读取 cmdq 的 fence Id
命令	echo readfence [chnid] > /proc/mi_modules/mi_vpe/mi_vpe0
参数说明	无
举例	echo readfence 0 > /proc/mi_modules/mi_vpe/mi_vpe0

	打印 workthread/bottom thread 中待输出的 reserve 值和 cmdq 当前 fence 值

*SSC323、SSC325、SSC325DE、SSC327、SSC32 暂不支持

Table 31: 表 7-13

功能	调试锁的位置
命令	echo debugmutex > /proc/mi_modules/mi_vpe/mi_vpe0
参数说明	无
举例	echo debugmutex > /proc/mi_modules/mi_vpe/mi_vpe0 channel 的锁， 和 workinglist 的锁的位置和时间。

*SSC323、SSC325、SSC325DE、SSC327、SSC32 不支持

Table 31: 表 7-14

功能	调试 Atom 的值
命令	echo setatom [AtomValue] > /proc/mi_modules/mi_vpe/mi_vpe0
参数说明	[AtomValue] 默认为 1. AtomValue+1 代表 mhal 拿住 buffer 的数量
举例	echo setatom 2 > /proc/mi_modules/mi_vpe/mi_vpe0

Table 31: 表 7-15

功能	设置 vpe port mirror/flip
命令	echo setmirrorflip [chnid] [portid][bmirror][bflip] > /proc/mi_modules/mi_vpe/mi_vpe0
参数说明	无
举例	echo setmirrorflip 0 0 1 1 > /proc/mi_modules/mi_vpe/mi_vpe0

8. VENC

8.1. cat

【调试信息】

#cat /proc/mi_modules/mi_venc/mi_venc0

```
------------------------------ All VENC Dev info ------------------------------
------------------------------ VENC 0 Dev info -------------------------------
  DevId  CmdqId  CmdqSize  CmdqCnt  IrqNum  IsrProc  IsrCnt
      0     -1     16384        0  25/24         0       0
  DevId  UtilHw  UtilMi  PeakHw  PeakMi     FPS  MbRate   %
      0       0       0       0       0    0.00  204000  37

------------------------------ VENC 0 CHN info -------------------------------
ChnId  RefMemPA  RefMemVA  RefMemBufSize   AlMemPA   AlMemVA  AlMemBufSize
    0   5F5A000    (null)        9822720   5F02000  d0888000        358080
ChnId  DevId  bStart  eBufState  FrameCnt  Fps_1s     kbps  Fps10s     kbps
    0      0       1          0         0    0.00        0    0.00        0
---------------------------- InputPort of dev: 0 ----------------------------
ChnId  Width  Height  SrcFrmRate  MaxW  MaxH  FrameCnt  DropCnt  BlockCnt
    0   1920    1080      25/1     1920  1080         0        0         0
---------------------------- OutputPort of dev: 0 ----------------------------
ChnId  CODEC  Profile   BufSize  RefNum  bByFrame  FrameCnt  DropCnt  ReEncCnt
    0   H264        0         0       0         1         0        0         0
ChnId RateCtl  GOP MaxBitrate  StatTime  FluctuateLevel
    0     CBR   50    2000000         0               0
```

【调试信息分析】

记录当前某 VENC device 的使用状况，以及 device 属性、layer 属性、inputport 属性，可以动态地获取到这些信息，方便调试和测试。

【参数说明】

Table 34: 表 8-1

参数		描述
Dev info	DevId	设备号，作为 Dev ID 使用（对映内核 MI_VENC_Dev_e）
	CmdqId	Command Queue 号
	CmdqSize	Command Queue 单帧大小（bytes）
	CmdqCnt	Command Queue 数量（number of frames）
	IrqNum	中断号
	IsrProc	是否有未处理 ISR 信号：0：无 1：有
	IsrCnt	Isr 信号接收次数
	UtilHw	encoder hw driver 使用率
	UtilMi	encoder mi sw 使用率

	PeakHw	encoder hw driver 使用率峰值
	PeakMi	encoder mi sw 使用率峰值
	FPS	Device 包含的所有 Channel 的 FPS 之和
	MbRate	Device 宏块编码速率(MB/s)
	%	Device 当前宏块编码速率占极限值的百分比

Table 35: 表 8-2

参数		描述
CHN info	ChnId	Channel ID 通道号
	RefMemPA	底层 driver 要求的硬件物理位址
	RefMemVA	底层 driver 要求的硬件虚拟位址
	RefMemBufSize	底层 driver 要求的硬件内存大小
	AlMemPA	底层 driver 要求的 CPU 物理位址
	AlMemVA	底层 driver 要求的 CPU 虚拟位址
	AlMemBufSize	底层 driver 要求的 CPU 内存大小
	bStart	是否已开始接收数据
	eBufState	Channel 当前的状态
	FrameCnt	Channel 当前编码帧的数量
	Fps_1s	Channel 最近 1s 内统计的帧率
	Kbps	Channel 最近 1s 内统计的码率
	Fps_10s	Channel 最近 10s 内统计的帧率
	Kbps	Channel 最近 10s 内统计的码率

Table 36: 表 8-3

参数		描述
Inputport info	ChnId	Channel ID 通道号
	Width	画面宽度 (pixels)
	Height	画面高度 (pixels)
	SrcFrmRate	用户设定的 VencAttr 帧率，通常以此作为 rate control 的参数
	MaxW	最大画面宽度 (pixels)
	MaxH	最大画面高度 (pixels)
	FrameCnt	已拿到的 input 帧数
	DropCnt	已丢弃的 input 帧数
	BlockCnt	因为 output buffer 申请不到导致丢弃的帧数，同时 DropCnt+1

Table 37: 表 8-4

参数		描述
Outputport info	ChnId	Channel ID 通道号
	CODEC	使用何种 CODEC: H264,H265,MJPG
	Profile	Profile 值
	BufSize	由 user 设定的每帧最大缓存 (bytes)
	RefNum	最大参考帧数
	bByFrame	后级是否一次取走一帧？ 0：否，通常代表按包取数据　1：是
	FrameCnt	已 release 的 output 帧数
	DropCnt	编码完成后，已丢弃的 output 帧数，会影响当前 GOP 值
	ReEncCnt	重编码的帧数
	(RingFrameCnt)	Ring Buffer 内已经处理完的帧数
Outputport Rate Control Info	RateCtl	Rate Control 模式：CBR, VBR, FixQp 等
	GOP	Group of Picture 大小
	MaxBitrate	CBR, VBR 时，最大的目标码率。(bits per second)
	QP.I	FixQp 时，I 帧的 QP
	QP.P	FixQp 时，P 帧的 QP
	StatTime	CBR, VBR 时，算法操作的时间参数
	FluctuateLevel	CBR 时，允许的摆荡程度
	MaxQp	VBR 时，允许的最大 QP
	MinQp	VBR 时，允许的最小 QP
	Qfactor	Jpeg 时，设定的 Qfactor 值

8.2. Echo

可以通过 echo help > /proc/mi_modules/mi_venc/mi_venc[DevID] 查看支持的所有 echo 指令。

Table 38: 表 8-5

功能	开启或者关闭 command queue 开机初始设定或者 Demo 退出后，下次生效
命令	echo en_cmdq [Status] > /proc/mi_modules/mi_venc/mi_venc[DevID]
参数说明	[Status] 1 开启 0 关闭
	[DevID] 设备 ID [0~1]
举例	echo en_cmdq 0 > /proc/mi_modules/mi_venc/mi_venc0 关闭 devcie 0 的 cmdq

Table 39: 表 8-6

功能	开启或者关闭中断 开机初始设定或者 Demo 退出后，下次生效

命令	echo en_irq [Status] > /proc/mi_modules/mi_venc/mi_venc[DevID]
参数说明	[Status] 1 开启 0 关闭
	[DevID] 设备 ID [0~1]
举例	echo en_irq 0 > /proc/mi_modules/mi_venc/mi_venc0 关闭 devide 0 的中断

Table 40: 表 8-7

功能	设定模块工程 Log 等级 直接修改输出 Log 等级，立即生效
命令	echo dbg_level [Level] > /proc/mi_modules/mi_venc/mi_venc[DevID]
参数说明	[Level] [0~3] 0: None 1: Error 2: option 1+ warning 3: option 2+ information [DevID] 设备号 [0~1]
举例	无

Table 40: 表 8-8

功能	保存 venc 输入的 yuv 数据
命令	echo dump_in [ChnID] [0~999] path > /proc/mi_modules/mi_venc/mi_venc[DevID]
参数说明	[ChnID]: 指定的通道号 [0~999]: 需要保存的帧数 Path: 保存的路径 [DevID] 设备号 [0~1]
举例	无

Table 40: 表 8-9

功能	检测当前编码后的 stream size 骤升，保存 venc 当前输入的 yuv 数据
命令	echo dump_in_big [ChnID] [0~999] path > /proc/mi_modules/mi_venc/mi_venc[DevID]
参数说明	[ChnID]: 指定的通道号 [0~999]: 需要保存的帧数 Path: 保存的路径 [DevID] 设备号 [0~1]

举例	因为 H264/H265 参考前一帧的原因，如果 yuv 变化剧烈会导致编码后的 stream size 突然增加，此时保存 yuv 数据，判断是否 yuv 存在花屏等异常

Table 40: 表 8-10

功能	保存 venc 输出的 bit stream 数据
命令	echo dump_out [ChnID] [0~9999] [i|n] path > /proc/mi_modules/mi_venc/mi_venc[DevID]
参数说明	[ChnID]: 指定的通道号 [0~9999]: 需要保存的帧数 [i|n] i: 从下一个 I 帧开始保存 n: normal 模式，马上开始保存 Path: 保存的路径 [DevID] 设备号 [0~1]
举例	无

Table 40: 表 8-11

功能	设定 venc 处理 output 模式
命令	echo drop_stream [ChnID] [d|r] > /proc/mi_modules/mi_venc/mi_venc[DevID]
参数说明	[ChnID] [0~63] | all [0~63]: 指定通道号 all: 指定所有的通道 [d|r] d: droped. Venc 编码完的帧直接丢弃，不会 release r: released. Venc 编码完的帧全部 release [DevID] 设备号 [0~1]
举例	无

9. DIVP

9.1. cat

【调试信息】

cat /proc/mi_modules/mi_divp/mi_divp0

```
------------------------- start dump CHN info --------------------------------
 ChnId  Status  AttrChg  bSendFrameTwice  FieldType0  FieldType1
     0       2        0                0           0           0
            ChnID  MaxW  MaxH  CropX  CropY  CropW  CropH  bHMirror  bVMirror  eDiType  eRotateType  eTnrLevel
   ChnAttr      0  1920  1080      0      0   1920   1080         0         0        0            0          0
 ChnAttrPre     0  1920  1080      0      0   1920   1080         0         0        0            0          0
 ChnAttrOrg     0  1920  1080      0      0   1920   1080         0         0        0            0          0
------------------------- end dump CHN info ----------------------------------

------------------------- start dump INPUT PORT info -------------------------
 ChnId  InputChg  bIPChg  Width  Height  Pixel  Stride  GetBuffCnt  2Pmode
     0         1       0   1920    1080      9    1920           3       0
------------------------- end dump INPUT PORT info ---------------------------

------------------------- start dump OUTPUT PORT info ------------------------
 ChnId  OutputChg  CompMode  PreWidth  PreHeight  PrePixel  Stride  phyaddr  FinishTaskCnt
     0          0         0      1280        720         1    5120  32f9000              3
------------------------- end dump OUTPUT PORT info --------------------------
```

Figure 7: 图 8-1

【调试信息分析】

记录当前某 DIVP device 的使用状况，以及 device 属性、layer 属性、inputport 属性，可以动态地获取到这些信息，方便调试和测试。

【参数说明】

Table 41: 表 8-1

参数		描述
	ChnId	0~31 32 Capture Channel 不对上层开放
	Status	0 INITED 1 CREATED 2 STARTED 3 STOPED 4 DISTROYED
	AttrChg	0：not change 1：changed
	bSendFrameTwice	0: not sendFrameTwice 1:SendFrameTwice *if the channel's TNR is opend, when I/P mode change or the first frame is comming in, the frame need to be send twice to

channelinfo		avoid showwing garbage.
	FieldType0, FieldType1 (last two fields's type)	0 NONE, //< no field. 1 TOP, //< Top field only. 2 BOTTOM, //< Bottom field only. 3 BOTH, //< Both fields. 4 NUM
	ChnAttr	当前 channel 属性
	ChnAttrPre	channel 的上一次设置属性
	ChnAttrOrg	channel 的原始设置属性
	MaxW，MaxH	设置最大宽高
	CropX，CropY， CropW，CropH	上层设置的 crop 位置
	bHMirror bVMirror	水平翻转使能 垂直翻转使能 0:disable 1:enable
	eDiType	0 OFF,//off 1 2D,///2.5D DI 2 3D,///3D DI 3 NUM,
	eRotateType	0 NONE, //Rotate 0 degrees 1 90, //Rotate 90 degrees 2 180, //Rotate 180 degrees 3 270, //Rotate 270 degrees 4 NUM,
	eTnrLevel	0 OFF, 1 LOW, 2 MIDDLE, 3 HIGH, 4 NUM,

Table 42: 表 8-2

参数		描述
	ChnId	Channel ID 0~31
	InputChg	input port 属性是否发生变化
	bIPChg	0 隔行和逐行之间没有发生变化 1 隔行切换到逐行，or 逐行到隔行
	Width	input port width
	Height	input port height

inputport info	Pixel	E_MI_SYS_PIXEL_FRAME_YUV422_YUYV = 0, E_MI_SYS_PIXEL_FRAME_ARGB8888, E_MI_SYS_PIXEL_FRAME_ABGR8888, E_MI_SYS_PIXEL_FRAME_RGB565, E_MI_SYS_PIXEL_FRAME_ARGB1555, E_MI_SYS_PIXEL_FRAME_I2, E_MI_SYS_PIXEL_FRAME_I4, E_MI_SYS_PIXEL_FRAME_I8, E_MI_SYS_PIXEL_FRAME_YUV_SEMIPLANAR_422, E_MI_SYS_PIXEL_FRAME_YUV_SEMIPLANAR_420, E_MI_SYS_PIXEL_FRAME_YUV_MST_420, //vdec mstar private video format E_MI_SYS_PIXEL_FRAME_YC420_MSTTILE1_H264, E_MI_SYS_PIXEL_FRAME_YC420_MSTTILE2_H265, E_MI_SYS_PIXEL_FRAME_YC420_MSTTILE3_H265, E_MI_SYS_PIXEL_FRAME_FORMAT_MAX
	GetBuffCnt	获取 buffer 的数量
	2Pmode	是否有开 2P mode 0:disable 1:enable

Table 43: 表 8-3

参数		描述
Outputport Info	ChnId	Channel ID 0~31
	OutputChg	Output 属性是否发生变化 0: not change 1: changed
	CompMode	0 //no compress 1 //compress unit is 256 bytes as a segment 2 //compress unit is the whole line 3 //compress unit is the whole frame 4 //number
	PreWidth PreHeight PrePixel	上一次设置的 output port 宽高和像素格式
	Stride	从 sys 拿到的 outputbuffer stride
	phyhaddr	Divp output 内存物理地址
	FinishTaskCnt	Divp output frame 计数

9.2. Echo

Table 44: 表 8-4

功能	打印接收到每一帧的 PTS，如果前端是 VDEC,会打印 VDEC 送过来的 FrameID，Debug 丢帧问题,正常情况下 FrameID 和 PTS 的值都是递增的。
命令	echo checkframeid [ChnID] [Status] > /proc/mi_modules/mi_divp/mi_divp0
参数说明	[ChnID] 通道号 [0~32]
	[Status] ON 开始打印 OFF 停止打印
举例	echo checkframeid 0 ON/OFF > /proc/mi_modules/mi_divp/mi_divp0 [MI DIVP PROCFS]:ChnID 0, PTS = 43956000 [MI DIVP PROCFS]:ChnID 0, PTS = 43989000 [MI DIVP PROCFS]:ChnID 0, PTS = 44022000 [MI DIVP PROCFS]:ChnID 0, PTS = 44055000 [MI DIVP PROCFS]:ChnID 0, PTS = 44088000 [MI DIVP PROCFS]:ChnID 0, PTS = 44121000 [MI DIVP PROCFS]:ChnID 0, PTS = 44154000 [MI DIVP PROCFS]:ChnID 0, PTS = 44187000 receive buffer ID：每个 channel 接收到 buffer 的 ID。 PTS:FrameBuffer 的 PTS。

Table 45: 表 8-5

功能	DIVP 不再处理某一个 channel，以节省频宽，验证频宽相关问题
命令	echo stoponechannel [ChnID] [Status] > /proc/mi_modules/mi_divp/mi_divp0
参数说明	[ChnID] 通道号 [0~32]
	[Status] ON 停止处理设置的 channel OFF 继续处理设置的 channel
举例	echo stoponechannel 0 ON > /proc/mi_modules/mi_divp/mi_divp0 通道 0 的画面将会停住

Table 46: 表 8-6

功能	DIVP 不处理所有的 channel，DIVP 硬件的属性为设置的 Channel 的属性。可以通过 Debug Register 来调试指定的 channel。
命令	echo processonechannel [ChnID] [Status] > /proc/mi_modules/mi_divp/mi_divp0
参数说明	[ChnID] 通道号 [0~32]
	[Status] ON 停止所有的通道，硬件属性为所设置的通道的属性（即此通道会再处理一次） OFF 所有通道正常运行
举例	echo processonechannel 0 ON > /proc/mi_modules/mi_divp/mi_divp0

	所有通道全部停止即画面卡住，但通道 0 会多处理一帧 echo processonechannel 0 OFF > /proc/mi_modules/mi_divp/mi_divp0 所有通道回复正常工作

10. VDISP

10.1. cat

【调试信息】

```
# cat proc/mi_modules/mi_vdisp/mi_vdisp0

==============================Private Vdisp0 Info
==============================
  DevStatus
    Start
---------------------------------------------------- Input Port Info ----------------------------------------
 PortID  PortStatus  ChnX  ChnY  ChnW  ChnH  IsFreeRun  TryOk  RecvOk
      0  Enabled     0     0     960   540   1          245356299  313332
      1  Enabled     960   0     960   540   1          355670297  313332
      2  Enabled     0     540   960   540   1          578760014  313331
      3  Enabled     960   540   960   540   1          757579130  313330
---------------------------------------------------- Output Port Info ----------------------------------------
 Inited  FrmInterval  BgColor  PixelFmt  FrmRate  Width  Height  SendOk
      1  33333        8388736  0         30       1920   1080    471418
```

【调试信息分析】

记录当前 VDISP 的使用状况以及 device 属性、Input port 属性、Output port 属性，可以动态地获取到这些信息，方便调试和测试。

【参数说明】

Table 48: 表 9-1

参数		描述
device info	DevStatus	Vdisp 设备的工作状态 Uninit：未初始化 Init：初始化 Start：运行状态 Stop：停止状态

Table 49: 表 9-2

参数		描述
	PortID	Inport ID 取值范围：[0~16]，16 为 overlay 通道，即 PIP 通道。
	PortStatus	Port 口状态

layer info		Uninit：inport 未初始化 Init：inport 已初始化 Enabled：inport 已经使能 Disabled：inport 已经禁用
	ChnX	输入 inport 的起始坐标 X 地址 取值范围：[0~4094]，需要根据 chip 的 align 对齐
	ChnY	输入 inport 的起始坐标 Y 地址 取值范围：[0~2159]
	ChnW	输入 inport 的图像宽度，需要根据 chip 的 align 对齐 取值范围：[0~4096]
	ChnH	输入 inport 的图像高度 取值范围：[0~4096]
	IsFreeRun	是否不做帧率控制 0：按 pts 控制播放 1：自由播放
	TryOk	尝试取前级绑定端口的数据次数
	RecvOk	从前级绑定端口成功拿到数据的次数

Table 50: 表 9-3

参数		描述
Output port info	Inited	是否有初始化 Output Port 0：未初始化 1：已初始化
	FrmInterval	出帧时间间隔，单位 US
	BgColor	背景色，此处打印的是 10 进制
	PixelFmt	色彩空间 取值范围[0~E_MI_SYS_PIXEL_FRAME_FORMAT_MAX-1] 0-YUYV422，9-YUVSP420
	FrmRate	输出帧率
	Width	图像输出宽度 取值范围：[0~4096]，需要根据 chip 的 align 对齐
	Height	图像输出高度 取值范围：[0~2160]，需要根据 chip 的 align 对齐
	SendOk	成功推往后级的帧数统计

11. DISP

11.1. cat

【调试信息】

cat /proc/mi_modules/mi_disp/mi_disp0

```
------------------------------------------- Private DISP0 Info -------------------------------------
    DevStatus          IrqNum          IrqStatus          BgColor          CvbsStatus
            1             211                  1                0                   0
    Interface0      DevTiming0          CscMatrix             Luma             Contrast
          HDMI        1080p60                  0               50                   50
           Hue      Saturation                Gain       Sharpness
            50              50                   0                0
------------------------------------------- Layer Info ---------------------------------------------
       LayerId     LayerStatus     BindedDevID      ePixFormat         Compress
             0               1               0               0                0
   LayerWidth     LayerHeight    LayDispWidth   LayDispHeight        Toleration
          1920            1080            1920            1080               100
  customer_alloc
          enable
------------------------------------------- Layer0 InputPort Info ----------------------------------
       PortId      PortStatus        DispWin_x        DispWin_y     DispWinWidth
            0               0                0                0             1920
DispWinHeight       ZoomWin_x        ZoomWin_y    ZoomWinWidth    ZoomWinHeight
         1080               0                0                0                0
     SyncMode       FlipByGe        RecvBufCnt     RecvBufWidth     RecvBufHeight
            0               0               20             1920             1080
RecvBufStride
         3840
```

【调试信息分析】

记录当前 DISP 的使用状况以及 device 属性、layer 属性、inputport 属性，可以动态地获取到这些信息，方便调试和测试。

【参数说明】

Table 51: 表 10-1

参数		描述
	DevStatus	使能或者禁用 0：禁用

device info		1：使能
	IrqNum	中断号
	IrqStatus	是否开启中断 0：中断禁用 1：中断使能
	BgColor	背景色（RGB format）
	CvbsStatus	是否开启 cvbs 0：关闭 cvbs 1：开启 cvbs
	Interface	接口类型 取值范围：（CVBS、YPBPR、VGA、BT656、BT1120、HDMI、LCD、BT656_H、BT656_L）
	DevTiming	输出时序 取值范围：[0~ E_MI_DISP_OUTPUT_USER]
	CscMatrix	CSC 矩阵选择 取值范围：[0~ E_MI_DISP_CSC_MATRIX_RGB_TO_BT709_PC]
	Luma	亮度 取值范围：[0 ~ 100] 默认 50
	Contrast	对比度 取值范围：[0 ~ 100] 默认 50
	Hue	色调 取值范围：[0 ~ 100] 默认 50
	Saturation	饱和度 取值范围：[0 ~ 100] 默认 40
	Sharpness	锐度 取值范围：[0 ~ 100] 默认 50

Table 52：表 10-2

参数		描述
layer info	LayerId	layer ID 取值范围：[0~1]
	LayerStatus	layer 状态 0：关闭 layer 1：开启 layer
	BindedDevID	绑定的 device ID 取值范围：[0~1]
	ePixFormat	像素格式 取值范围：

		[0~E_MI_SYS_PIXEL_FRAME_YC420_MSTTILE3_H265]
	Compress	数据压缩 取值范围： [E_MI_SYS_COMPRESS_MODE_NONE~E_MI_SYS_COMPRESS_MODE_BUTT]
	LayerWidth	Layer 的宽
	LayerHeight	Layer 的高
	LayDispWidth	Layer 显示的宽
	LayDispHeight	Layer 显示的高
	Toleration	PTS 误差允许阈值，单位毫秒
	customer_alloc	disp 是否开启 buff 管理

Table 53: 表 10-3

参数		描述
port info	PortStatus	port 状态 取值范围：[E_MI_LAYER_INPUTPORT_STATUS_INVALID~E_MI_LAYER_INPUTPORT_STATUS_HIDE]
	DispWin_x	该 port 在 layer 上的起始横坐标
	DispWin_y	该 port 在 layer 上的起始纵坐标
	DispWinWidth	该 port 输入的宽度
	DispWinHeight	该 port 输入的高度
	ZoomWin_x	局部放大区域起始横坐标
	ZoomWin_y	局部放大区域起始纵坐标
	ZoomWinWidth	局部放到区域宽度
	ZoomWinHeight	局部放大区域高度
	Sync Mode	Check PTS/Free Run 0：无效 1：Check PTS 2：Free Run
	FlipByGe	是否使用 GE 0：不使用 GE 1：使用 GE
	RecvBufCnt	DISP 拿到前端 buff 总个数
	RecvBufWidth	DISP 拿到 buff 的 width
	RecvBufHeight	DISP 拿到 buff 的 height
	RecvBufStride	DISP 拿到 buff 的 stride

11.2. echo

Table 53: 表 10-4

功能	设置对应 port 的播放模式
命令	echo setsyncmode [layerid] [portid] [mode] > /proc/mi_modules/mi_disp/mi_disp0
参数说明	[layerid] 视频层号
	[portid] port 号
	[mode]播放模式
举例	echo setsyncmode 0 0 2 > /proc/mi_modules/mi_disp/mi_disp0 设置 layer0 port0 为 free run

Table 54: 表 10-5

功能	使能或失能 irq
命令	echo setIrqStatus [status] > /proc/mi_modules/mi_disp/mi_disp0
参数说明	[status] ON 打开 OFF 关闭
举例	echo setIrqStatus ON > /proc/mi_modules/mi_disp/mi_disp0 使能 disp irq

Table 55: 表 10-6

功能	动态设置 layer 属性
命令	echo setlayer [layerid] [lay_width] [lay_height] [laydisp_x] [laydisp_y] [laydisp_width] [laydisp_height] > /proc/mi_modules/mi_disp/mi_disp0
参数说明	[layerid] 视频层 id [lay_width] 视频层宽 [lay_height] 视频层高 [laydisp_x] 视频层显示的 x 坐标 [laydisp_y] 视频层显示的 y 坐标 [laydisp_width] 视频层显示的宽 [laydisp_height] 视频层显示的高
举例	echo setlayer 0 1280 720 0 0 1920 1080 > /proc/mi_modules/mi_disp/mi_disp0

Table 56: 表 10-7

功能	动态设置视频层的状态
命令	echo setlayerstatus [layerid] [status] > /proc/mi_modules/mi_disp/mi_disp0
参数说明	[layerid] 视频层 id [status] ON 使能 OFF 失能
举例	echo setlayerstatus 0 ON > /proc/mi_modules/mi_disp/mi_disp0

Table 57: 表 10-8

功能	动态设置画面效果
命令	echo setcsc [devid] [CscMatrix] [Contrast] [Hue] [Luma] [Saturation] > /proc/mi_modules/mi_disp/mi_disp0
参数说明	[devid] 设备 id

	[CscMatrix] 颜色矩阵 [Contrast] 对比度 [Hue] 色调 [Luma] 亮度 [Saturation] 饱和度
举例	echo setcsc 0 0 50 50 50 50 > /proc/mi_modules/mi_disp/mi_disp0

Table 58: 表 10-9

功能	设置背景色
命令	echo setbgcolor [devid] [value] > /proc/mi_modules/mi_disp/mi_disp0
参数说明	[devid] 设备 id [value] 颜色值
举例	echo setbgcolor 0 255 > /proc/mi_modules/mi_disp/mi_disp0

Table 59: 表 10-10

功能	dump 每帧 pts
命令	echo checkframepts [layeid] [portid] [ON/OFF] > /proc/mi_modules/mi_disp/mi_disp0
参数说明	[layerid] 视频层 id [portid] port id [ON/OFF] ON 打开 OFF 关闭
举例	echo checkframepts 0 0 OFF > /proc/mi_modules/mi_disp/mi_disp0

Table 60: 表 10-11

功能	dump 帧率
命令	echo checkframerate [layerid] [portid] [ON/OFF] > /proc/mi_modules/mi_disp/mi_disp0
参数说明	[layerid] 视频层 id [portid] port id [ON/OFF] ON 打开 OFF 关闭
举例	echo checkframerate 0 0 OFF > /proc/mi_modules/mi_disp/mi_disp0

Table 61: 表 10-12

功能	dump fame data
命令	echo dumpframe [layerid] [portid] [path] > /proc/mi_modules/mi_disp/mi_disp0
参数说明	[layerid] 视频层 id [portid] port id
举例	echo dumpframe 0 0 /mnt > /proc/mi_modules/mi_disp/mi_disp0

Table 62: 表 10-13

功能	stop 或 resume disp 单个 port get buff
命令	echo stopgetbuff [layerid] [portid] [ON/OFF] > /proc/mi_modules/mi_disp/mi_disp0
参数说明	[layerid] 视频层 id [portid] port id [ON/OFF] ON:stop OFF:resume

举例	echo stopgetbuff 0 0 ON > /proc/mi_modules/mi_disp/mi_disp0

12. HDMI

12.1. cat

【调试信息】

```
# cat /proc/mi_modules/mi_hdmi/mi_hdmi0

------------------------------------ HDMI0 Dev Info ------------------------------------
  InitFlag   AvMute   PowerOn
     Y         N         Y
  EnableVideo   TimingType   OutputMode   ColorType   DeepColorMode
      1             9             0            2              4
  EnableAudio   IsMultiChannel   BitDepth   CodeType   SampleRate
      1             0             16          0            3
```

【调试信息分析】

记录当前 HDMI 的使用状况以及 device 属性可以动态地获取到这些信息，方便调试和测试。

【参数说明】

Table 54: 表 11-1

参数		描述
device info	InitFlag	HDMI 模块是否初始化 Y：已初始化 N：未初始化
	AvMute	音视频是否处于 MUTE 状态 Y：被 MUTE N：未被 MUTE
	PowerOn	HDMI 是否使能电源管理 Y：是 N：否
	EnableVideo	是否使能视频 0：未使能 1：使能
	TimingType	当前 HDMI 的输出分辨率 取值范围[0~ E_MI_HDMI_TIMING_MAX]
	OutputMode	HDMI 输出模式 0：HDMI 模式 2：DVI 模式

		1/3 为 HDCP 模式，当前暂不支持
	ColorType	颜色空间 0：RGB44 1：YUV422 2：YUV444 3：YUV420
	DeepColorMode	色彩位深 0：8Bit 1：10Bit 2：12Bit 3：16Bit
	EnableAudio	是否使能音频 0：未使能 1：使能
	IsMultiChannel	是否多通道 0：否 1：是
	BitDepth	音频采样位深 8：8Bit 16：16Bit 32：32Bit
	CodeType	音频输出的压缩格式 0：PCM 1：非 PCM
	SampleRate	音频输出采样率 0：未知 1：32K 2：44K 3：48K 4：88K 5：96K 6：176K 7：192K

13. FB

13.1. cat

【调试信息】

cat /proc/mi_modules/mi_fb0

##############################OnDumpOSDAttr##############################
#########
Framebuffer id = MStar FB0
xres=1280, yres=720
xres_virtual=1280, yres_virtual=1440
xoffset=0,yoffset=0
fix.line_length=0xa00 Bytes
fix.smem_start=0xf800000
Memory Size=0x400000 Bytes
Gop ID=0
Gwin ID=0
Open Count=1
Visible State=0
MIU Sel=0
ColorSpace=YUV
ColorFomrmat=ARGB1555
StretchWindow Pos[0,0]
StretchWindow Src[1280,720],StretchWindow Dst[1920,1080]
Gwin Pos[0,0]
Gwin Size[1280,720]
Gwin PhyAddr=0x0
ColorKey Enable=0
ColorKey Val=[0,0,0]
Enable Alpha Blend=1
Enalbe Multi Alpha=0
Global Alpha Val=0x0
Alpha0=0x0,Alpha1=0x0,(Only for ARGB1555)
GOP Hstart=192
Current TimingWidth=1920,TimingWidth=1080,hstar=192
##################################end##################################
#############

##########################OnDumpHwcursorAttr##############################
#########
Cursor Gop ID=4
Cursor Gwin ID=6
Cursor MIU Sel=0
Cursor PhyAddr=0x166000
Cursor ColorFmt=ARGB8888
Cursor Icon Width=44,Height=56

Cursor HotSpot[18,9]
Cursor request pos[712,207]
Cursor Visible=1
Cursor Gwin Pos[694,198]
Cursor Gwin Size[44,56]
Cursor Gwin Pitch=0x200 Bytes
Curosr StretchWindow pos[0,0]
Cursor StretchWin Src[1280,720],Dst[1920,1080]
Cursor ColorKey Enable=0
Cursor ColorKey Value=[0xff,0xff,0xff]
##########################OnDumpHwcursorAttr#########################
#########

【调试信息分析】

记录当前 Fbdev && Hwcursor 的使用状况以及 OSD 图层属性、OSD device 属性、Hwcursor 属性，可以动态地获取到这些信息，方便调试和测试。

【参数说明】

Table 55: 表 12-1

参数		描述
OSD info	Framebuffer id	Framebuffer 设备 ID
	xres	可见分辨率宽度
	yres	可见分辨率宽高度
	xres_virtual	虚拟分辨率的宽度
	yres_virtual	虚拟分辨率的高度
	xoffset	可见分辨率到虚拟分辨率的 x 偏移量
	yoffset	可见分辨率到虚拟分辨率的 y 偏移量
	fix.line_length	虚拟分辨率每行的字节数
	fix.smem_start	帧缓冲内起始地址
	Memory Size	帧缓冲内存长度，单位字节数

Table 56: 表 12-2

参数		描述
OSD device info	Gop ID	图层硬件 ID
	Gwin ID	图层硬件显示窗口 ID
	Open Count	该图形层打开次数。在用户调用 open()时增 1；调用 close()时减 1
	Visible State	该图层显示状态。 取值:{1 表示可见，0 表示不可见} 用户显示调用 FBIOSET_SHOW 去设置图层可见状态。
	MIU Sel	保存帧缓冲数据的物理内存编号。 对于 MSR620 设备该值为 0

	ColorSpace	该图形层输出的颜色空间。 取值:{YUV 表示输出 YUV color space;RGB 表示输出 RGB color space }。默认为 YUV
	ColorFomrmat	图形层格式 取值:{ARGB1555,ARGB4444,ARGB8888,RGB565}。 用户设置可变屏幕信息中的格式项后更新
	StretchWindow Pos	图形层在显示设备上的起始位置。 单位:像素 默认为 0,用户可调用 FBIOSET_SCREEN_LOCATION 去设置
	StretchWindow Src	图形层原始分辨率的宽度和高度。 单位:像素 通过硬件放大功能放大到当前 timing。用户设置可变屏幕信息中的可见分辨率后更新
	StretchWindow Dst	图形层在显示设备上的宽度和高度。 默认设置为输出 timing 大小。可以通过 FBIOSET_SCREEN_LOCATION 去设置
	Gwin Pos	图形层显示窗口的起始位置 单位:像素 对于 OSD 图层来说,该值总是为(0,0)
	Gwin Size	图形层显示窗口的宽度、高度。 单位:像素 对于 OSD 图层来说，该值总是被设置为可见分辨率的宽度、高度。用户设置可变屏幕信息中的可见分辨率后更新。
	Gwin PhyAddr	图形层当前显示数据的物理地址
	ColorKey Enable	图形层 ColorKey 是否使能 取值:{0 表示 Disable ColorKey; 1 表示 Enable ColorKey}用户设置可通过 FBIOSET_COLORKEY 去设置
	ColorKey Val	图形层 ColorKey 的 RED,GREEN,BLUE 通道数值 取值:每个通道的取值范围从 0-255。ColorKey val 的数值用 RGB888 格式表示 用户可通过 FBIOSET_COLORKEY 去设置后更新
	Enable Alpha Blend	图形层 alpha 是否使能。 取值:{0 表示否，1 表示是},默认为 1。 用户可通过 FBIOSET_GLOBAL_ALPHA 去设置后更新。 该项关闭，则像素 alpha 配置不生效。该项开启且 Enable Multi Alpha 关闭,仅像素 alpha 生效(对于 ARGB1555 格式,Alpha bit 为 1 的像素使用 Global Alpha 作为像素 alpha,Alpha bit 为 0 的像素使用 0xff 作为像素 alpha)。当该项目开启且 Enable Multi Alpha 生效时,则像素 alpha 和 Global Alpha 都生效

	Enable Multi Alpha	全局 alpha 是否生效开关 取值{0 时，1 否}默认值为 0 Enable Multi Alpha 使能，Global Alpha 便生效
	Global Alpha Val	全局 Alpha
	Alpha0	ARGB1555 格式时，当最高位为 0 时,选择 Alpha0 作为 Alpha 混合的的 Alpha 数值 取值: 0~255 默认为 0
	Alpha1	ARGB1555 格式时，当最高位为 1 时，选择 Alpha1 作为 Alpha 混合的 Alpha 数值 取值: 0~255 默认为 0
	GOP Hstart	图层硬件向对于显示窗口的偏移。 取值:该值和当前输出 timing 相关，当输出 timing 发生变化时，通过 FBIOSET_DISPLAYLAYER_ATTRIBUTES 并将输入参数 MI_FB_DisplayLayerAttr_s 的 u32SetAttrMask 数值包含 E_MI_FB_DISPLAYLAYER_ATTR_MASK_SCREEN_SIZE mask 位来通知图层驱动输出 timing 发生变化。图层驱动会根据当前 timing 设置相应的 hstart 数值
	Current TimingWidth, Current TimingHeight, hstar	当前输出 timing 的宽度、高度以及当前 timing 下应该给图形层硬件设置的像对于显示窗口的偏移。 该项与 GOP Hstart 配合使用，用于检查设置给硬件图层的 hstart 数值是否和当前 timing 一致。

Table 57: 表 12-3

参数		描述
Hwcursor info	Cursor Gop ID	鼠标图层硬件 ID
	Cursor Gwin ID	鼠标图层硬件显示窗口 ID
	Cursor MIU Sel	用于保存鼠标图标数据的物理内存编号。 对于 MSR620 设备该值为 0
	Cursor PhyAddr	鼠标层当前显示数据的物理地址
	Cursor ColorFmt	鼠标图层格式 值:{ARGB1555,ARGB4444,ARGB8888,RGB565}。 用户可通过 FBIOSET_CURSOR_ATTRIBUTE 设置
	Cursor Icon Width	鼠标图标的宽度、高度。 单位：像素 用户可通过 FBIOSET_CURSOR_ATTRIBUTE 设置
	Cursor HotSpot	鼠标图标的热点信息 单位:像素

		热点信息通常和鼠标图标相关，用户可通过 FBIOSET_CURSOR_ATTRIBUTE 设置
	Cursor request pos	鼠标图标的位置信息 单位：像素 用户通过 FBIOSET_CURSOR_ATTRIBUTE 设置
	Cursor Visible	鼠标图层是否可见 取值:{0,表示不可见，1 表示可见} 用户通过 FBIOSET_CURSOR_ATTRIBUTE 设置
	Cursor Gwin Pos	鼠标图层显示窗口的起始位置 单位:像素
	Cursor Gwin Pitch	鼠标图层每行的字节数。 对于鼠标图层来说，该值固定为 0x200
	Curosr StretchWindow pos	鼠标图层在显示设备上的起始位置。 单位:像素
	Cursor StretchWin Src	鼠标图层原始大小 单位:像素。该值通常和 OSD 的显示分辨率一致。
	Cursor StretchWin Dst	鼠标图层在显示设备上的宽度、高度 该值和输出 timing 一致。
	Cursor ColorKey Enable	鼠标图层 ColorKey 是否使能 取值:{0 表示 Disable ColorKey; 1 表示 Enable ColorKey}。 用户通过 FBIOSET_CURSOR_ATTRIBUTE 设置
	Cursor ColorKey Value	鼠标图层 ColorKey 的 RED,GREEN,BLUE 通道数值 取值:每个通道的取值范围从 0-255。ColorKey val 的数值用 RGB888 格式表示 用户可通过 FBIOSET_CURSOR_ATTRIBUTE 去设置后更新

13.2. echo

Table 58: 表 12-4

功能	打开或关闭指定的图层
命令	echo GUI_SHOW [LayerID] [Status] > /proc/mi_modules/mi_fb0
参数说明	[LayerID] 图层号
	[Status] on 打开 off 关闭
举例	echo GUI_SHOW 0 on > /proc/mi_modules/mi_fb0 显示图层 fb0 echo GUI_SHOW 0 off > /proc/mi_modules/mi_fb0

	显示图层 fb0

Table 59: 表 12-5

功能	打开或关闭硬件鼠标
命令	echo CURSOR_SHOW [Status] > /proc/mi_modules/mi_fb0
参数说明	[Status] on 打开 off 关闭
举例	echo CURSOR_SHOW on > /proc/mi_modules/mi_fb0 显示鼠标图标 echo CURSOR_SHOW off > /proc/mi_modules/mi_fb0 关闭鼠标图标

Table 60: 表 12-6

功能	为指定的图层设置 Colorkey
命令	echo GUI_SET_CLRKEY [LayerID] [Red] [Green] [Blue] > /proc/mi_modules/mi_fb0
参数说明	[LayerID] 图层号
	[Red] 红色分量 以 16 进制,ARGB888 格式表示
	[Green] 绿色分量 以 16 进制,ARGB888 格式表示
	[Blue] 蓝色分量 以 16 进制,ARGB888 格式表示
举例	echo GUI_SET_CLRKEY 0 ff 00 00 > /proc/mi_modules/mi_fb0 设置 fb0 的 Colorkey 为红色

Table 61: 表 12-7

功能	Dump 当前正在显示的鼠标图标
命令	echo CURSOR_DUMP [Path] >/proc/mi_modules/mi_fb0
参数说明	[Path] dump 的路径
举例	echo CURSOR_DUMP /mnt > /proc/mi_modules/mi_fb0 dump cursor Icon 到/mnt 路径，文件名为 CursorData.raw，用 7yuv 打开，宽高设置为 128,128，格式按照 cursor icon 的格式设置即可显示 Rawdata

Table 62: 表 12-8

功能	分别为 ARGB1555 格式的 alpha bit0,alpha bi1 设置 pixel alpha 数值。因为目前 msr620 不支持这个 feature，暂时未实现
命令	echo GUI_SETALPHA_ARGB1555 [LayerID] [Alpha0] [Alpha1]> /proc/mi_modules/mi_fb0

参数说明	[LayerID]　图层号
	[Alpha0]　alpha0 的数值
	[Alpha1]　alpha1 的数值
举例	无

14. GFX

14.1. cat

【调试信息】

cat /proc/mi_modules/mi_gfx/mi_gfx0

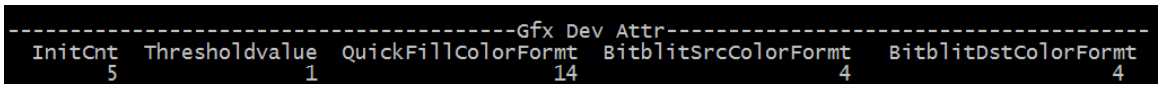

Figure 8: 图 13-1

【调试信息分析】

记录当前 GFX 的使用状况以及 device 属性可以动态地获取到这些信息，方便调试和测试。

【参数说明】

Table 63: 表 13-1

参数		描述
device info	InitCnt	设备被 Open 的次数，由于 GFX 会有多个模块进行调用，并且不能确定是哪一个 module 会先调用 GFX 的初始化，所以设备允许多次 Open，会记录 Open 次数来在 Close 的时候做对应的计数判断是否需要真的 Close Device。
	Thresholdvalue	ARGB1555 时，GFX 内部的 alpha 判定阈值，如大于 N 则 alpha 为 1，反之 alpha 为 0. 在 ARGB8888 模式下无效。默认值为 1.
	QuickFillColorFormt	最后一次做 QuickFill 的颜色格式 取值参考：GFX_Buffer_Format
	BitblitSrcColorFormt	最后一次做 BitBlit 的 Src Surface 的颜色格式
	BitblitDstColorFormt	最后一次做 BitBlit 的 Dst Surface 的颜色格式

15. REGION

15.1. cat

【调试信息】

cat /proc/mi_modules/mi_rgn0

```
/ # cat /proc/mi_modules/mi_rgn/mi_rgn0

----------- Start dump region capability info ----------
Module name : REGION

Limitation :
Osd supported format : fmt   0 Support
Osd supported format : fmt   1 Support
Osd supported format : fmt   2 Support
Osd supported format : fmt   3 Support
Osd supported format : fmt   4 Support
Osd supported format : fmt   5 Support
Osd supported format : fmt   6 Support
Region handle : 1024 (0 ~ 1023)
Vpe channel   :   64 (0 ~   63)
Vpe port      :    4 (0 ~    3)
Divp channel  :   64 (0 ~   63)
Divp port     :    1 (0 ~    0)
Osd attach    :    8 (each channel port)
Cover attach  :    4 (each channel port)

Osd support :
  Width   : 1 ~ 3840
  Height  : 1 ~ 2160
  HW mode : 2 frontbuffs
  Color key value : 0x2323
  X pos overlap : Support
  Overlap : Support

Cover support :
Size :
  Width   : 1 ~ 8192
  Height  : 1 ~ 8192
  Overlap : Support
----------End dump region capability info ----------
```

Figure 9: 图 14-1

```
------------------------------------------ Start dump region attr ------------------------------------------
 Handle        Type       Width      Height      Stride      Format     VirAddr     PhyAddr
      0         OSD         128          96         256           0           0     3db0000
      1         OSD         200         301         400           0           0     3dc0000
      2         OSD         300         360         608           0           0     7355000
      3         OSD         400         266         800           0           0     738b000
      4         OSD         200         356         400           0           0     73bf000
      5         OSD         200         131         400           0           0     3de0000
      6         OSD         300         224         608           0           0     73e2000
      7         OSD         200         133         400           0           0     3df0000
      8       COVER
      9       COVER
     10       COVER
     11       COVER
------------------------------------------ End dump region attr ------------------------------------------
```

Figure 10: 图 14-2

```
------------------------------------ Start dump Vpe Channel 0 port 0 info ------------------------------------
---------------------------------------- Scaling Info ----------------------------------------
    CanvasWidth     CanvasHeight     ScreenWidth     ScreenHeight
           1920             1080            1920             1080
---------------------------------------- frontbuff info ----------------------------------------
  Index    bShow  OffsetX  OffsetY    Width   Height   Stride   Format  VirAddr  PhyAddr AlphaMode AlphaVal  BgAlpha  FgAlpha
      0        1        0        0      952      866     1904        0        0  5633000     Pixel                0      255
      1        0        0        0        0        0        0        0        0        0     Pixel                0        0
---------------------------------------- region info ----------------------------------------
 Handle   Type  bShow  Layer   Color  Width Height Stride PositionX PositionY  Format  VirAddr  PhyAddr AlphaMode BgAlpha FgAlpha
      0    OSD      1    323          128     96    256         0         0       0        0  1a18000     Pixel       0     255
      1    OSD      1     33          200    301    400       400         0       0        0  1a20000     Pixel       0     255
      2    OSD      1     22          300    360    608       100       200       0        0  398c000     Pixel       0     255
      3    OSD      1    300          400    266    800       400       400       0        0  4dd3000     Pixel       0     255
      4    OSD      1    222          200    356    400         0       510       0        0  4d75000     Pixel       0     255
      5    OSD      1    333          200    131    400       150         0       0        0  1a40000     Pixel       0     255
      6    OSD      1    420          300    224    608       650         0       0        0  4e07000     Pixel       0     255
      7    OSD      1    678          200    133    400       650       400       0        0  1a50000     Pixel       0     255
      8  COVER      1      0  6b1dff   1000     1000              1200      1200
      9  COVER      1      1  15952b   1000     1000              1800      1800
     10  COVER      1      2  ff4c54   1000     1000              2400      2400
     11  COVER      1      3  7d038f   1000     1000              3000      3000
```

Figure 11: 图 14-3

```
------------------------------------------ Start dump color invert info ------------------------------------------
 Handle  OsdLayer  bShow  AreaId  BColorInv  PosX  PosY  Width  Height  InvMode  Threshold  Th1  Th2  Th3  DivW  DivH
      0       323      1       0          0     0     0      0       0        0          0  128  128    0     0     0
      1        33      1       1          0     0     0      0       0        0          0  128  128    0     0     0
      2        22      1       2          0     0     0      0       0        0          0  128  128    0     0     0
      3       300      1       3          0     0     0      0       0        0          0  128  128    0     0     0
      4       222      1       4          0     0     0      0       0        0          0  128  128    0     0     0
      5       333      1       5          0     0     0      0       0        0          0  128  128    0     0     0
      6       420      1       6          0     0     0      0       0        0          0  128  128    0     0     0
      7       678      1       7          0     0     0      0       0        0          0  128  128    0     0     0
      0       323      1       8          0     0     0      0       0        0          0  128  128    0     0     0
      1        33      1       9          0     0     0      0       0        0          0  128  128    0     0     0
      2        22      1      10          0     0     0      0       0        0          0  128  128    0     0     0
      3       300      1      11          0     0     0      0       0        0          0  128  128    0     0     0
      4       222      1      12          0     0     0      0       0        0          0  128  128    0     0     0
      5       333      1      13          0     0     0      0       0        0          0  128  128    0     0     0
      6       420      1      14          0     0     0      0       0        0          0  128  128    0     0     0
      7       678      1      15          0     0     0      0       0        0          0  128  128    0     0     0
------------------------------------------ End dump color invert info ------------------------------------------
```

Figure 12: 图 14-4

```
---------------------------------
Memory used:
kmalloc    : 5040 bytes
mma malloc : 2609712 bytes
---------------------------------
```

Figure 12: 图 14-5

【调试信息分析】

RGN 的调试信息按照上图所示，分四个部分：

图 14-1 显示的是 Region 的 device 信息，展现的是 Region 模块硬件的能力值以及一些参数的边界。

图 14-2 显示的是当前创建的 region 信息。

图 14-3 显示的是当前 region attach 到通道上相关的信息，从这些信息中看出 Region attach 的内存分布为 background buffer 部分和 front buffer 两部分，front buffer 是硬件真正使用的内存。

图 14-4 显示的是 APP 的反色设定。

图 14-5 显示的是 region 内部统计 MMA 的 buffer 使用的大小，以及 Kmalloc 使用的大小。

【参数说明】

Table 64: 表 14-1

参数		描述
Region capability	Osd support fmt	支持的 osd 格式: 0：ARGB1555 1：ARGB4444 2：I2 3: I4 4: I8 5: RGB565 6: ARGB8888
	Region handle	创建 region 的最大数量为 1024 个，句柄取值为[0~1023]
	Vpe channel	VpeR 最大通道数量为 64 个，通道取值为[0~63]
	Vpe port	Vpe 最大输出端口数量为 4 个，端口取值为[0~3]
	Divp channel	Divp 最大通道数量为 64 个，通道取值为[0~63]
	Divp port	Divp 最大输出端口数量为 1 个，端口取值为 0
	Osd attach	每个输出端口每个通道能绑定的 Osd 数量上限
	Cover attach	每个输出端口每个通道能绑定的 Cover 数量上限
	Osd support	Width: Osd 宽取值范围。 Height: Osd 高取值范围。 HW mode: OSD 支持硬件的 layer 个数。 Color key value：OSD 所用的 Color key 数值，内部固定写死，不支持 APP 自行设定。 X pos overlap： OSD 是否支持 x 方向重叠。 X overlap： OSD 是否支持重叠。 注：硬件不支持重叠的情况，region 内部会自动通过软件的方式实现重叠。
	Cover support	Width: Cover 宽取值为[1~8192] Height: Cover 高取值为[1~8192] Overlap：是否支持范围重叠

Table 65: 表 14-2

参数		描述
	Handle	句柄

Region attr	Type	类型，Osd 或是 Cover
	Width	宽
	Height	高
	Stride	宽补齐
	Format	OSD 颜色格式: 0：ARGB1555 1：ARGB4444 2：I2 3: I4 4: I8 5: RGB565 6: ARGB8888
	VirAddr	Canvas 的虚拟地址
	PhyAddr	Canvas 的物理地址

Table 66: 表 14-3

参数			描述
Region attr	Frontbuffer info	Index	Frontbuffer 索引
		bShow	是否显示
		OffsetX	X 偏移
		OffsetY	Y 偏移
		Width	宽
		Height	高
		Stride	宽补齐
		Format	颜色格式
		VirAddr	虚拟地址
		PhyAddr	物理地址
		AlphaMode	Apha 显示模式
		BgAlpha	Argb1555 背景 Alpha
		FgAlpha	Argb1555 前景 Alpha
	Attach info	Handle	绑定 Region 句柄
		Type	绑定 Region 类型
		bShow	是否显示
		Layer	绑定 Cover 的层级
		Color	绑定 Cover 的颜色值
		Width	宽
		Height	高

		Stride	宽补齐
		PositionX	X 偏移
		PositionY	Y 偏移
		Format	绑定 Osd 的颜色格式
		VirAddr	虚拟地址
		PhyAddr	物理地址
		AlphaMode	Apha 显示模式
		BgAlpha	Argb1555 背景 Alpha
		FgAlpha	Argb1555 前景 Alpha

Table 67: 表 14-4

参数		描述
Buffer info	kmalloc	Region 模块内部申请内存字节数。
	mma alloc	Region 模块通过 Sys 模块申请内存字节数。

15.2. echo

Table 68: 表 14-5

功能	Dump 指定 Region 的 buffer
命令	echo dumpRgnBuf [Handle] [Path] > /proc/mi_modules/mi_rgn/mi_rgn0
参数说明	[Handle] region 句柄
	[Path] 保存 dump 数据的路径。保存内容为 region 的 canvas 内容
举例	echo dumpRgnBuf 0 /mnt > /proc/mi_modules/mi_rgn/mi_rgn0 在/mnt 下产生文件 Rgn0_canvasInfo_fmt0_128X96_stride256 表示 dump 句柄为 0，颜色格式为 ARGB1555，宽为 128，高为 96，宽对齐为 256 字节的 region 的 canvas 数据 文件格式为: Rgn[Handle]_canvasInfo_fmt[Format]_[Height]X[Height]_stride[Stride] [Handle]: region 句柄。 [Format]：颜色格式，ARGB1555 为 0，ARGB4444 为 1，I2 为 2，I4 为 3。 [Width]：宽 [Height]：高 [Stride]：宽补齐

Table 69: 表 14-6

功能	Dump 指定 channel 和 port 的 frontbuffer
命令	echo dumpFrontBuf [ModId] [ChnID] [PortID] [Path] > /proc/mi_modules/mi_rgn/mi_rgn0
参数说明	[ModId] 端口类型，Vpe 为 0，Divp 为 1。

	[ChnID] 通道号 [0 ~ 63]
	[PortID] 端口号 [0~3]
	[Path] 保存 dump 数据的路径。会根据当前实际使用的 frontbuffer 数量生成 0 ~ 2 个文件
举例	echo dumpFrontBuf 0 0 0 /mnt > /proc/mi_modules/mi_rgn/mi_rgn0 在/mnt 下产生 Vpe_Chn0_Port0_frontBuf0_fmt0_952X866_stride1904，表示 dump 通道号为 0，输出端口为 Vpe0 号端口，索引为 0 的 frontbuffer 的数据，颜色格式为 ARGB1555，宽为 952，高为 866，宽对齐为 1904 字节。 文件格式 [Mod]_Chn[Channel]_Port[Port]_frontbuf[Index]_fmt[Format]_[Width]X[Height]_stride[Stride] [Mod] 输出端口类型，Vpe 时为 Vpe，Divp 时为 Divp。 [Channel 通道号。 [Port] 输出端口。 [Index] frontbuffer 索引号。 [Format] 颜色格式，ARGB1555 为 0，ARGB4444 为 1，I2 为 2，I4 为 3。 [Width] 宽 [Height] 高 [Stride] 宽补齐

Table 70: 表 14-7

功能	获取 region 能力级信息
命令	echo getcap > /proc/mi_modules/mi_rgn/mi_rgn0
参数说明	无
举例	无

Table 71: 表 14-8

功能	获取已创建 region 信息
命令	echo dumprgn > /proc/mi_modules/mi_rgn/mi_rgn0
参数说明	无
举例	无

Table 72: 表 14-9

功能	获取 channel 和 port 的信息
命令	echo dumpchport > /proc/mi_modules/mi_rgn/mi_rgn0
参数说明	无
举例	无

Table 73: 表 14-10

功能	获取内存使用信息
命令	echo bufcnt > /proc/mi_modules/mi_rgn/mi_rgn0

参数说明	无
举例	无

Table 74: 表 14-11

功能	获取调色板设定
命令	echo dumpPalette> /proc/mi_modules/mi_rgn/mi_rgn0
参数说明	无
举例	无

Table 75: 表 14-12

功能	获取反色设定
命令	echo dumpColorInvert> /proc/mi_modules/mi_rgn/mi_rgn0
参数说明	无
举例	无

Table 76: 表 14-13

功能	设定 OSD 显示开关
命令	echo setDispOnOff [ModId] [ChnID] [PortID] [Index] [OnOff]> /proc/mi_modules/mi_rgn/mi_rgn0
参数说明	[ModId] 端口类型，Vpe 为 0，Divp 为 1。
	[ChnID] 通道号 [0 ~ 63]
	[PortID] 端口号 [0~3]
	[Index] Frontbuffer 索引
	[OnOff] 开关 1：表示开，0：表示关。
举例	echo setDispOnOff 0 0 0 1> /proc/mi_modules/mi_rgn/mi_rgn0 表示 VPE 的 channel 0 port 0 上的 index 0 所对应 frontbuffer 的内容不显示。

Table 76: 表 14-13

功能	设定一个 16 位数值，对当前的 Color key 值进行“或“操作
命令	echo setcolorkeymask [MASK]> /proc/mi_modules/mi_rgn/mi_rgn0
参数说明	[MASK] 支持十六进位表示，也支持十进位表示。
举例	echo setcolorkeymask 0xFFFF > /proc/mi_modules/mi_rgn/mi_rgn0 Color key mask 可以使 Colorkey 失效，可以很直观地看出 osd 的显示区域的内存分配范围。

16. VDEC

16.1. cat

【调试信息】

cat /proc/mi_modules/mi_vdec/mi_vdec0

```
==================================Private Vdec0 Info====================================
-----------------------------------CHN ATTR Info----------------------------------------
ChnID  CodecType  Width  Height   BufSize  Priority  VideoMode  JpegFormat  RefFrmNum
    0         0    720     576   1048576         0          1           4          4
    1         0    720     576   1048576         0          1           4          4
    2         0    720     576   1048576         0          1           4          4
    3         0    720     576   1048576         0          1           4          4

-----------------------------------CHN PARAM Info---------------------------------------
ChnID  DecMode  OutputOrder  VideoFormat  DisplayMode
    0       0            0            0            0
    1       0            0            0            0
    2       0            0            0            0
    3       0            0            0            0

-----------------------------------CHN STATE--------------------------------------------
ChnID  bStart  CodecType  LeftStrmBytes  LeftStrmFrm     LeftPic  RecvStrmFrm  DecStrmFrm
    0       1          0              0            0         396          396         396
    1       1          0         116265            1         396          397         396
    2       1          0              0            0         396          396         396
    3       1          0              0            0         397          398         398

ChnID      ErrCnt     DropCnt     SkipCnt
    0           0           0           0
    1           0           0           0
    2           0           0           0
    3           1           0           0
```

Figure 11: 图 15-1

【调试信息分析】

打印分为两部分，使用 Private Vdec0 Info 分隔开。上半部分为 common 信息，下半部分为 vdec 模块信息。主要记录了解码通道的使用情况及配置属性，可用于检查属性配置和当前通道的工作状态，便于 debug。

【参数说明】

Table 74: 表 15-1

参数		描述
CHN ATTR Info	ChnID	通道号。
	CodecType	解码协议类型 0：H264; 1：H265; 2：JPEG。
	Width	解码图像最大宽度。

	Height	解码图像最大高度。
	BufSize	VDEC 码流 buffer 大小，单位：byte。
	Priority	解码通道优先级。
	VideoMode	发送码流方式。 0：按流发送； 1：按帧发送。
	JpegFormat	Jpeg 图片的存储格式 0：YCbCr400； 1：YCbCr420； 2：YCbCr422； 3：YCbCr444。
	RefFrmNum	最大参考帧个数，JPEG 通道无效。
CHN PARAM Info	ChnID	通道号。
	DecMode	解码模式。 0：解码 IPB 数据帧模式； 1：只解 I 帧模式； 2：只解 IP 帧，跳过 B 帧。
	OutputOrder	解码图像输出顺序。 0：按显示顺序输出数据帧； 1：按解帧顺序输出数据帧。
	VideoFormat	解码图像数据格式。 0：TILE 数据格式； 1：数据帧压缩模式，减少数据帧内存使用量。
	DisplayMode	显示模式。 0：预览模式，不参考 PTS 输出。 1：回放模式，参考 PTS 数值输出。
CHN STATE	ChnID	通道号。
	bStart	解码器是否启动接收码流。
	CodecType	解码协议类型 0：H264; 1：H265; 2：JPEG
	LeftStrmBytes	码流 buffer 中待解码的 byte 数。
	LeftStrmFrm	码流 buffer 中待解码的帧数。–1 表示无效。 仅按帧发送时有效。
	LeftPics	图像 buffer 中剩余的 pic 数目。
	RecvStrmFrm	码流 buffer 中已接收码流帧数。-1 表示无效。 仅按帧发送时有效。
	DecStrmFrm	码流 buffer 中已解码帧数。

16.2. echo

Table 75: 表 15-2

功能	设置解码模式
命令	echo setDecMode [ChnID] [ModID] > /proc/mi_modules/mi_vdec/mi_vdec0
参数说明	[ChnID] 通道号 [0~32]
	[ModID] 解码模式 0->H264 1->H265 2->JPEG
举例	echo setDecMode 0 0 > /proc/mi_modules/mi_vdec/mi_vdec0 通道 0，解码模式为 H264

Table 76: 表 15-3

功能	设置解码图像输出顺序
命令	echo setOutputOrder [ChnID] [ModID] > /proc/mi_modules/mi_vdec/mi_vdec0
参数说明	ChnID 通道号 [0~32]
	ModID 0 按显示顺序输出数据帧 1 按解码顺序输出数据帧
举例	echo setOutputOrder 0 0 > /proc/mi_modules/mi_vdec/mi_vdec0 通道 0，按显示顺序输出

Table 77: 表 15-4

功能	设置解码图像数据格式，暂不支持设置
命令	echo setVideoFormat [ChnID] [ModID] > /proc/mi_modules/mi_vdec/mi_vdec0
参数说明	[ChnID] 通道号 [0~32]
	[ModID] 解码数据格式 0 TILE 格式 1 数据帧压缩模式，减少数据帧内存使用量
举例	echo setVideoFormat 0 0 > /proc/mi_modules/mi_vdec/mi_vdec0 通道 0，TILE 格式输出

Table 78: 表 15-5

功能	设置显示模式
命令	echo setDisplayMode [ChnID] [ModID] > /proc/mi_modules/mi_vdec/mi_vdec0
参数说明	[ChnID] 通道号 [0~32]
	[ModID] 0：预览模式，不参考 PTS 输出。

	1：回放模式，参考 PTS 数值输出。
举例	echo setDisplayMode 0 0 > /proc/mi_modules/mi_vdec/mi_vdec0 通道 0 预览模式

Table 79: 表 15-6

功能	打开/关闭统计打印码流输入时间间隔，每隔一秒打印一次
命令	echo statInputTimeIntvl [ChnID] [Status] >/proc/mi_modules/mi_vdec/mi_vdec0
参数说明	[ChnID] 通道号 [0~32]
	[Status] on 打开 off 关闭
举例	echo statInputTimeIntvl 0 on >/proc/mi_modules/mi_vdec/mi_vdec0 [MI VDEC PROCFS]:Input ChnID: 0, CurTime: 5959100469, TotalFrmCnt: 19, AvgTimeIntvl: 49856, MaxTimeIntvl: 50047, MinTimeIntvl: 47252 [Input ChnID] 通道号 [CurTime] 当前时间 [TotalFrmCnt] 输入帧次数 [AvgTimeIntvl 每秒内所有帧输入平均时间间隔 [MaxTimeIntvl] 每秒内两帧输入最大时间间隔 [MinTimeIntvl] 每秒内两帧输入最小时间间隔。

Table 80: 表 15-7

功能	打开/关闭统计打印图像输出时间间隔
命令	echo statOutputTimeIntvl[ChnID][Status]>/proc/mi_modules/mi_vdec/mi_vdec0
参数说明	[ChnID] 通道号 [0~32]
	[Status] on 打开 off 关闭
举例	echo statOutputTimeIntvl 0 on> /proc/mi_modules/mi_vdec/mi_vdec0 MI VDEC PROCFS]:Output ChnID: 0, CurTime: 6413830139, TotalFrmCnt: 19, AvgTimeIntvl: 49999, MaxTimeIntvl: 50021, MinTimeIntvl: 49985 [Output ChnID] 通道号 [CurTime] 当前时间 [TotalFrmCnt] 输入帧次数 [AvgTimeIntvl 每秒内所有帧输入平均时间间隔 [MaxTimeIntvl] 每秒内两帧输入最大时间间隔 [MinTimeIntvl] 每秒内两帧输入最小时间间隔。

Table 81: 表 15-8

功能	打开/关闭统计打印输入码流时间戳
命令	echo ChkInputFrmPts [ChnID] [Status] > /proc/mi_modules/mi_vdec/mi_vdec0
参数说明	[ChnID] 通道号 [0~32]
	[Status] on 打开 off 关闭
举例	echo ChkInputFrmPts 0 on > /proc/mi_modules/mi_vdec/mi_vdec0 [MI VDEC PROCFS]:Input ChnID: 0, TotalFrmCnt: 400, PTS: 3964389 [Input ChnID] 通道号； [TotalFrmCnt] 输入帧次数； [PTS] 码流时间戳。

Table 82: 表 15-9

功能	打开/关闭统计打印输出码流时间戳
命令	echo ChkOutputFrmPts [ChnID] [Status] > /proc/mi_modules/mi_vdec/mi_vdec0
参数说明	[ChnID] 通道号 [0~32]
	[Status] on 打开 off 关闭
举例	echo ChkOutputFrmPts 0 on > /proc/mi_modules/mi_vdec/mi_vdec0 [MI VDEC PROCFS]:Output ChnID: 0, TotalFrmCnt: 154, PTS: 4039893000 [Output ChnID] 通道号； [TotalFrmCnt] 输入帧次数； [PTS] 码流时间戳。

Table 83: 表 15-10

功能	将写入码流 buffer 的码流同时存文件
命令	echo DumpBS [Path] [MoreChnID] > /proc/mi_modules/mi_vdec/mi_vdec0
参数说明	[Path] 存储路径
	[MoreChnID] 通道号 [0~32] 可以设置多路，如 0 1 2 3
举例	echo DumpBS /mnt 0 > /proc/mi_modules/mi_vdec/mi_vdec0 [MI WRN]: _MI_VDEC_IMPL_InjectBuffer[737]: Chn(0) Dump Es Buffer, Size:5765, (0x00 0x00 0x00 0x01 0x61) 在/mnt 下产生 chn_0_h264_dump_vdec.es 文件，为 es 流文件

Table 84: 表 15-11

功能	将解码出的图像同时存文件，存 3 帧
命令	echo DumpFB [Path] [MoreChnID] > /proc/mi_modules/mi_vdec/mi_vdec0

参数说明	[Path] 存储路径
	[MoreChnID] 通道号 [0~32] 可以设置多路，如 0 1 2 3
举例	echo DumpFB 0 /mnt > /proc/mi_modules/mi_vdec/mi_vdec0 [MI WRN]: _MI_VDEC_IMPL_DebugWriteFile[3494]: dump file(/mnt/chn_0_h264_dump_vdec[720_576_736]_0.yuv) v1 ok .....[len:635904] 在 /mnt 下产生 chn_0_h264_dump_vdec[720_576_736]_0.yuv（已 detile）， chn_0_dump_vdec[720_576_736]_0_luma.yuv， chn_0_dump_vdec[720_576_736]_0_chrome.yuv 各 3 帧，共 9 帧

Table 85: 表 15-12

功能	将当前码流 buffer 中的所有数据存储到文件中
命令	echo DumpCurBS [Path] [ChnID] > /proc/mi_modules/mi_vdec/mi_vdec0
参数说明	[Path] 存储路径
	[ChnID] 通道号 [0~32]
举例	echo DumpCurBS /mnt 0 > /proc/mi_modules/mi_vdec/mi_vdec0 Dump Chn(0) ES Data Now 在/mnt 下 产生 chn_0_h264_dump_now_vdec.es

17. WARP

17.1. cat

【调试信息】

cat /proc/mi_modules/mi_warp/mi_warp0

```
/ # cat /proc/mi_modules/mi_warp/mi_wrap0

----------------------------------------Common info for mi_wrap0----------------------------------------
  ChnNum  EnChnNum  InPortNum  OutPortNum  CollectSize
       1         1          1           1            0

-----------------------------Common info for mi_wrap0 only dump enabled chn-----------------------------
  ChnId  EnInPNum  EnOutPNum  MMAHeapName
      0        1          1       (null)

-------------------------Input port common info for mi_wrap0  only dump enabled port--------------------
  ChnId  PortId  SrcFrmrate  DstFrmrate  user_buf_quota  UsrInjectQ_cnt  UsrInjectQ_size  BindInQ_cnt  BindInQ_size  Worki
ngQ_cnt  WorkingQ_size  usrLockedInjectCnt
      0       0          30          30               4               0                0            0             0
      1       12441600                   0

  ChnId  PortId  bind_module_id  bind_module_name  bind_ChnId  bind_PortId
      0       0               7            mi_vpe           0            0

  ChnId  PortId   LowLatencyDelayMs    scheduledDelayTaskCnt     LastStaticDelayAveMS

----------------------Output port common info for mi_wrap0  only for enabled port----------------------
  ChnId  PortId usrDepth  BufCntQuota  usrLockedCnt  totalOutPortInUsed   DrvBkRefFifoQ_cnt  DrvBkRefFifoQ_size
      0       0        2            4             0                   3                  0                   0
  ChnId  PortId  UsrGetFifoQ_cnt UsrGetFifoQ_size  UsrGetFifoQ_seqnum  UsrGetFifoQ_discardnum  WorkingQ_cnt  WorkingQ_size
      0       0                2         24883200                 777                     775             1       12441600

----------------------------------------BindPeerInputPortList----------------------------------------
  ChnId  PortId  bind_module_id  bind_module_name  bind_ChnId  bind_PortId
      0       0               2           mi_venc           0            0

---------------------------------- Dump Warp Dev0 InputPort Info ----------------------------------
  ChnId      Format  Width Height    GetInput    FinishInput    RewindInput         Fps   StatDepth
      0   YUV420_SP    3840   2160         778            777              0        28.6          20

-------------------------------- End Dump Warp Dev0 InputPort Info --------------------------------

---------------------------------- Dump Warp Dev0 OutputPort Info ----------------------------------
  ChnId      Format  Width Height   GetOutput   FinishOutput    RewindInput         Fps   StatDepth
      0   YUV420_SP    3840   2160         778            777              0        28.6          20

-------------------------------- End Dump Warp Dev0 OutputPort Info --------------------------------

-------------------------------- Dump Warp Dev0 Hal Info --------------------------------
  ChnId AvgTime(us) MaxTime(us) MinTime(us)   TotalTrigger  FinishTrigger
      0       12674       86100        9566            778            777

------------------------------ End Dump Warp Dev0 Hal Info ------------------------------
```

Figure 12: 图 16-1

【调试信息分析】

打印分为两部分，common 信息和 warp 模块信息。

主要记录了 warp 设备各通道的使用情况及端口状态，可用于检查属性配置和当前通道的工作状态，便于 debug。

【参数说明】

Table 86: 表 16-1

参数		描述
Dev InputPort Info	ChnId	通道号。
	Format	Pixel format: YUV422_YUYV; YUV420_SP; Unknown。
	Width	帧宽度。
	Height	帧高度。
	GetInput	InputPort 接收到的 buffer 数量。
	FinishInput	InputPort 结束的 buffer 数量，包括正常处理完成和丢弃的帧数。
	RewindInput	InputPort 放回的 buffer 数量。
	Fps	InputPort 端的帧率。根据两帧时间间隔进行采样，先计算平均两帧时间间隔，再计算帧率。
	StatDepth	采样样本大小。取值范围为（1~100）。
Dev OutputPort Info	ChnId	通道号。
	Format	Pixel format: YUV422_YUYV; YUV420_SP; Unknown。
	Width	帧宽度。
	Height	帧高度。
	GetOutput	OutputPort 接收到的 buffer 数量。
	FinishOutput	OutputPort 正常处理完成的 buffer 数量。
	RewindOutput	OutputPort 放回的 buffer 数量。
	Fps	OutputPort 端的帧率。根据两帧时间间隔进行采样，先计算平均两帧时间间隔，再计算帧率。
	StatDepth	采样样本大小。取值范围为（1~100），默认值为 20。
Dev Hal Info	ChnId	通道号。
	AvgTime	Hal 执行单次 trigger 平均耗时。单位 us。
	MaxTime	Hal 执行单次 trigger 最大耗时。单位 us。
	MinTime	Hal 执行单次 trigger 最小耗时。单位 us。
	TotalTrigger	Hal 层总共 trigger 数量。
	FinishTrigger	Hal 层执行成功的 trigger 数量。

17.2. echo

Table 87: 表 16-2

功能	Dump 模块配置文件。
命令	echo dump_table2file [ChnID] [BinType] [BinPath] > /proc/mi_modules/mi_warp/mi_warp0
参数说明	[ChnID] 通道号 0
	[BinType] Bin 文件类型 0 -> Bounding Box Table dump 文件名称为 Warp_BbTable.bin 1 -> Displayment Table dump 文件名称为 Disp_absolute.bin 或 Disp_relative.bin （根据 warpConfig 中 disp_table 类型不同，生成不同文件）。
	[BinPath] Bin 文件路径
举例	echo 0 0 /mnt/warp > /proc/mi_modules/mi_vdec/mi_vdec0 通道 0，Bin 文件类型为 Bounding Box，路径为/mnt/warp。最后在/mnt/warp 目录生成文件 Warp_BbTable.bin。

Table 88: 表 16-3

功能	Dump 输入/输出端口帧率。
命令	echo dump_fps > /proc/mi_modules/mi_warp/mi_warp0
参数说明	无。
举例	无。

Table 89: 表 16-4

功能	统计输入/输出端口数据帧状态。
命令	echo dump_frameCnt > /proc/mi_modules/mi_warp/mi_warp0
参数说明	无。
举例	无。

Table 90: 表 16-5

功能	统计 hal 时间消耗。
命令	echo dump_halTimeConsume > /proc/mi_modules/mi_warp/mi_warp0
参数说明	无。
举例	无。

Table 91: 表 16-6

功能	设置统计样本大小。
命令	echo dump_SetStatDepth [Depth] > /proc/mi_modules/mi_warp/mi_warp0
参数说明	[Depth] 样本大小。
举例	echo dump_SetStatDepth 30 /mnt/warp > /proc/mi_modules/mi_vdec/mi_vdec0

	设置统计样本大小为 30。会统计两帧间的 pts 差值，放入样本集合中，最多可放置 30 个，不足 30 时，根据实际样本数量计算 fps；达到 30 时，按照 30 个 pts 差值计算 fps。

18. VDF

18.1. cat

【调试信息】

cat /proc/mi_modules/mi_shadow/mi_vdf0

```
----------------------------------------Common info for mi_vdf0----------------------------------------
 ChnNum  EnChnNum  InPortNum  OutPortNum  CollectSize
     64         2          1           1            0

-----------------------------Common info for mi_vdf0 only dump enabled chn-----------------------------
 ChnId  EnInPNum  EnOutPNum  MMAHeapName
     0        1          1       (null)
     1        1          1       (null)

------------------------Input port common info for mi_vdf0  only dump enabled port--------------------
 ChnId  PortId  SrcFrmrate  DstFrmrate  user_buf_quota  UsrInjectQ_cnt  UsrInjectQ_size  BindInQ_cnt  BindInQ_size  WorkingQ_cnt  WorkingQ_size  usrLockedInjectCnt
     0       0         30           5               4               0                0            0             0             0              0                   0
     1       0         30           5               4               0                0            0             0             0              0                   0

 ChnId  PortId  bind_module_id  bind_module_name  bind_ChnId  bind_PortId
     0       0              12           mi_divp           0            0
     1       0              12           mi_divp           0            0

 ChnId  PortId   LowLatencyDelayMs    scheduledDelayTaskCnt     LastStaticDelayAveMS

----------------------Output port common info for mi_vdf0  only for enabled port----------------------
 ChnId  PortId usrDepth  BufCntQuota  usrLockedCnt  totalOutPortInUsed   DrvBkRefFifoQ_cnt  DrvBkRefFifoQ_size
     0       0        4            6             0                   0                  0                  0
     1       0        4            6             0                   0                  0                  0
 ChnId  PortId  UsrGetFifoQ_cnt UsrGetFifoQ_size  UsrGetFifoQ_seqnum  UsrGetFifoQ_discardnum  WorkingQ_cnt  WorkingQ_size
     0       0                0                0                  83                       0             0              0
     1       0                0                0                  76                       0             0              0

-------------------------------------BindPeerInputPortList-------------------------------------
 ChnId  PortId  bind_module_id  bind_module_name  bind_ChnId  bind_PortId
```

Figure 13: 图 18-2

```
------------------------------ ** Dump all private info for VDF ** ---------------------------------
 VDF Verison:   [Dec 21 2018 12:13:36]

------------------------------ Dump DEV info for VDF ----------------------------------------
   bInited       MDEnable       ODEnable       VGEnable    YuvTaskExit     MDTaskExit     ODTaskExit     VGTaskExit       DbgLevel
         1              1              1              0              0              0              0              0              2

--------------------- CHN common info for VDF only dump created chn -------------------------
     ChnId        eStatus        phandle       FrameCnt       FrameInv       YImgSize       ImageQCnt
         0              2     0xb24e3008              0              0         110592              0
     ChnId        eStatus        phandle       FrameCnt       FrameInv       YImgSize       ImageQCnt
         1              2     0xb2417008              0              0         110592              0

--------------------- Input Port info for VDF only dump created chn -------------------------
     ChnId       InGetCnt   InGetFailCnt      InDropCnt      InDoneCnt          InFps
         0             83            610              0             83           5.33
     ChnId       InGetCnt   InGetFailCnt      InDropCnt      InDoneCnt          InFps
         1             76            564              0             76           5.33

--------------------- Output Port info for VDF only dump created chn ------------------------
     ChnId      OutGetCnt  OutGetFailCnt     OutDropCnt     OutDoneCnt      RstGetCnt  RstGetFailCnt     RstDropCnt      RstPutCnt         OutFps
         0             83              0              0             83             83            956              0             83           5.33
     ChnId      OutGetCnt  OutGetFailCnt     OutDropCnt     OutDoneCnt      RstGetCnt  RstGetFailCnt     RstDropCnt      RstPutCnt         OutFps
         1             76              0              0             76             76            927              0             76           5.00

------------- MD Attr info for VDF only dump created chn [Ver:  20181207] -------------------
[====chn:0====]
[c_param]:         Enable       MdBufCnt       VDFIntvl      RstBufSize   SubRstObjLen   SubRstSadLen   SubRstStsLen   ccl_InitArea       ccl_Step
                        1              4              0            6592           3060           1728           1728             8              2
[s_param]:          width         height         stride          color        mb_size    sad_out_ctrl     roi_md_num    md_alg_mode
                      384            288            384              1              1              1              4              1
                roi_md_num      (pnt[0].x      pnt[0].y)      (pnt[1].x      pnt[1].y)      (pnt[2].x      pnt[2].y)      (pnt[3].x      pnt[3].y)
                        4              0              0            383              0            383            287              0            287
[d_param]:     sensitivity     learn_rate         md_thr     obj_num_max
                       80           2000             16              0

------------- OD Attr info for VDF only dump created chn [Ver:  20180717] -------------------
[====chn:1====]
[c_param]:         Enable       OdBufCnt       VDFIntvl      RstBufSize
                        1              4              0              32
[s_param]:          inImgW         inImgH     inImgStride       nClrType          alpha            div      roi_od_num              M   MotionSensit
                      384            288            384              1              2              2              4            120              0
                roi_od_num      (pnt[0].x      pnt[0].y)      (pnt[1].x      pnt[1].y)      (pnt[2].x      pnt[2].y)      (pnt[3].x      pnt[3].y)
                        4              0              0            383              0            383            287              0            287
[d_param]:      thd_tamper tamper_blk_thd   min_duration
                        3              1             15

------------- VG Attr info for VDF only dump created chn [Ver:        0] -------------------
```

Figure 14: 图 18-1

【调试信息分析】

打印分为两部分，common 信息和 vdf 模块信息。

主要记录了 vdf 设备各通道的使用情况及端口状态，可用于检查属性配置和当前通道的工作状态，参数设置，便于 debug。Common 信息的说明请参考 sys 模块说明一章，vdf 模块信息说明请参考下表。

【参数说明】

Table 92 表 18-1

参数		描述
DEV info for VDF	VDF Verison	版本信息
	bInited	Vdf 模块是否初始化。（0-否，1-是）
	MDEnable	MD 是否使能状态。（0-否，1-是）
	ODEnable	OD 是否使能状态。（0-否，1-是）
	VGEnable	VG 是否使能状态。（0-否，1-是）
	YuvTaskExit	获取 yuv 数据的线程是否退出状态。（0-否，1-是）
	MDTaskExit	MD 运算线程是否退出状态。（0-否，1-是）
	ODTaskExit	OD 运算线程是否退出状态。（0-否，1-是）
	VGTaskExit	VG 运算线程是否退出状态。（0-否，1-是）
	DbgLevel	Vdf 模块的 debug 等级。 参数说明： 0- disable 1- error 2- warn 3- info 4- debug 5- trace
CHN common info for VDF **(only dump created chn)**	ChnId	通道号。
	eStatus	通道运行状态。 参数说明： 0- 销毁 1- 创建 2- 运行 3- 停止
	phandle	当前通断对应的 handle
	FrameCnt	暂不使用。
	FrameInv	暂不使用。
	YImgSize	该通道处理的一张 yImage 数据大小（height*stride）
	ImageQCnt	当前队列里有几张未处理的 yImage 数据
Input Port info for VDF **(only dump created chn)**	ChnId	通道号。
	InGetCnt	获取 yImage 数据成功次数。
	InGetFailCnt	获取 yImage 数据失败次数。
	InDropCnt	丢弃不处理的 yImage 数据次数。

	InDoneCnt	存入待处理队列的 yImage 数据次数。	
	InFps	获取 yImage 数据的帧率统计	
Output Port info for VDF **(only dump created chn)**	ChnId	通道号。	
	OutGetCnt	申请内存（用于存放运算结果）成功次数。	
	OutGetFailCnt	获取内存（用于存放运算结果）失败次数。	
	OutDropCnt	丢弃不处理的内存（用于存放运算结果的）次数。	
	OutDoneCnt	写入运算结果到内存（用于存放运算结果的）成功次数。	
	RstGetCnt	用户获取运算结果成功次数。	
	RstGetFailCnt	用户获取运算结果失败次数。	
	RstDropCnt	丢弃（被新数据覆盖）的运算结果次数。	
	RstPutCnt	用户使用完成，归还运算结果使用内存的次数。	
	OutFps	获取运算结果的帧率统计	
MD Attr info for VDF **(only dump created chn)**	ChnId	通道号。	
	[c_param] 常规参数	Enable	当前通道是否使能。
		MdBufCnt	设置的 Md 存放结果的缓冲队列深度。
		VDFIntvl	暂不使用。
		RstBufSize	一次返回结果的大小。
		SubRstObjLen	CCL 输出结果的大小。
		SubRstSadLen	SAD 输出结果的大小。
		SubRstStsLen	检查结果状态的大小。
		ccl_InitArea	CCL 区面积的门槛值
		ccl_Step	CCL 每提高一次门槛值的提升值
	[s_param] 静态参数	width	MD 输入 Image 宽
		height	MD 输入 Image 高
		stride	MD 输入 Image stride
		color	MD 输入 Image 的类型
		mb_size	MD 宏块大小 0- MB_4x4 1- MB_8x8 2- MB_16x16
		sad_out_ctrl	MD 的 SAD 输出格式 0- 16BIT_SAD 1- 8BIT_SAD
		md_alg_mode	CCL 联通区域的运算模式 0- FG 模式

			1- SAD 模式
		roi_md_num	侦测区域边界点数量
		Pnt[i].x	侦测区域每个边界点的 x 坐标
		Pnt[i].y	侦测区域每个边界点的 y 坐标
	[d_param] 动态参数	sensitivity	算法灵敏度，范围[10,20,30,.....100]，值越大越灵敏，输入的灵敏
		learn_rate	单位毫秒，范围[1000,30000]，用于控制前端物体停止运动多久时，才作为背景画面
		md_thr	判断移动的门坎值，随不同模式而有不同设定标准
		obj_num_max	CCL 的连通区域数量限制值
OD Attr info for VDF **(only dump created chn)**	ChnId	通道号。	
	[c_param] 常规参数	Enable	当前通道是否使能。
		OdBufCnt	设置的 Od 存放结果的缓冲队列深度。
		VDFIntvl	暂不使用。
		RstBufSize	一次返回结果的大小。
	[s_param] 静态参数	inImgW	OD 输入 Image 宽
		inImgH	OD 输入 Image 高
		inImgStride	OD 输入 Image stride
		nClrType	OD 輸入影像的類型 2- y 数据
		alpha	控制产生参考图像的学习速率
		div	OD 检测窗口的类型 0- 1 个窗口 1- 4 个窗口 2- 9 个窗口
		M	多少张影像更新一次参考图像
		MotionSensit	移动敏感度设置
		roi_od_num	侦测区域边界点数量
		pnt[i].x	侦测区域每个边界点的 x 坐标
		pnt[i].y	侦测区域每个边界点的 y 坐标
	[d_param] 动态参数	thd_tamper	图像差异比例门坎值
		tamper_blk_thd	图像被遮挡区域数量门坎值
		min_duration	图像差异持续时间门坎值
VG Attr info for VDF **(only dump**	ChnId	通道号。	
	[c_param]	Enable	当前通道是否使能。

created chn)	常规参数	VgBufCnt	设置的 Vg 存放结果的缓冲队列深度。
		VDFIntvl	暂不使用。
		RstBufSize	一次返回结果的大小。
	[s_param] 静态参数	width	VG 输入 Image 宽
		height	VG 输入 Image 高
		stride	VG 输入 Image stride
		obj_size_thd	滤除物体占感兴趣区域的百分比大小
		indoor	相机架设区域 0- 室外 1- 室内
		func_state	侦测模式 2- VG_VIRTUAL_GATE，表示模式为虚拟线段。 3- VG_REGION_INVASION，表示模式区域入侵。
		line_number	虚拟线段的数目，支持 1 到 4 条虚拟线段 (侦测模式为 VG_VIRTUAL_GATE 起效)
		p[i].x	线段 n 的 x 坐标 (侦测模式为 VG_VIRTUAL_GATE 起效)
		p[i].y	线段 n 的 y 坐标 (侦测模式为 VG_VIRTUAL_GATE 起效)
		pd[i].x	线段 n 的第一个方向点 (侦测模式为 VG_VIRTUAL_GATE 起效)
		pd[i].y	线段 n 的第二个方向点 (侦测模式为 VG_VIRTUAL_GATE 起效)
		region_dir	设定区域入侵的方向，目前有进入、离开及穿越三种方向可以选择。 0- REGION_ENTER，进入警报区域触发警报。 1- REGION_LEAVING，离开警报区域触发警报。 2- REGION_CROSS，表示穿越警报区域触发警报。 (侦测模式为 VG_REGION_INVASION 起效)
		Pnt[i].x	区域的第 i 个点的 x 坐标 (侦测模式为 VG_REGION_INVASION 起效)
		Pnt[i].x	区域的第 i 个点的 y 坐标 (侦测模式为 VG_REGION_INVASION 起效)

18.2. echo

Table 93 表 18-2

功能	显示 vdf 支持的 echo 命令。

命令	echo help > /proc/mi_modules/mi_shadow/mi_vdf0
参数说明	无。
举例	echo help > /proc/mi_modules/mi_shadow/mi_vdf0

Table 94: 表 18-3

功能	设置 vdf 日志等级
命令	echo log_level [level] > /proc/mi_modules/mi_shadow/mi_vdf0
参数说明	[level]: 0-CLOSE 1ERROR 2WARN 3INFO 4-DEBUG 5-TRACE
举例	echo log_level 4 > /proc/mi_modules/mi_shadow/mi_vdf0

Table 95: 表 18-4

功能	线程状态检查
命令	echo sem_check [bool] > /proc/mi_modules/mi_shadow/mi_vdf0
参数说明	[bool]: 0-disable 1-enable
举例	echo sem_check 1 > /proc/mi_modules/mi_shadow/mi_vdf0

Table 96: 表 18-5

功能	使能工作模式
命令	echo en_workmode [mode] [bool] > /proc/mi_modules/mi_shadow/mi_vdf0
参数说明	[mode]: md/od/vg [bool]: 0-disable 1-enable
举例	echo en_workmode md 1 > /proc/mi_modules/mi_shadow/mi_vdf0

Table 97: 表 18-6

功能	使能通道口
命令	echo en_chn [chnId] [bool] > /proc/mi_modules/mi_shadow/mi_vdf0
参数说明	[chnId]: 0-63 [bool]: 0-disable 1-enable
举例	echo en_chn 0 1 > /proc/mi_modules/mi_shadow/mi_vdf0

Table 98: 表 18-7

功能	打开自动检查功能
命令	echo auto_check [bool] > /proc/mi_modules/mi_shadow/mi_vdf0
参数说明	[bool]: 0-disable 1-enable
举例	echo auto_check 1 > /proc/mi_modules/mi_shadow/mi_vdf0

Table 99: 表 17-8

功能	打开所有 debug 信息
命令	echo dump_all [bool] > /proc/mi_modules/mi_shadow/mi_vdf0
参数说明	[bool]: 0-disable 1-enable
举例	echo dump_all 1 > /proc/mi_modules/mi_shadow/mi_vdf0

Table 100: 表 18-9

功能	Dump yImage 数据
命令	echo dump_image [chnId] [dumpNum] [dumpPath] > /proc/mi_modules/mi_shadow/mi_vdf0
参数说明	[chnId]: 0-63 [dumpNum]: 0-xxxx [dumpPath] : "yourPath"
举例	echo dump_image 0 10 /mnt > /proc/mi_modules/mi_shadow/mi_vdf0