地平线
Horizon Robotics

Debugging guide for XJ3 image media module

V0.9

2020-03

This document contains the information related to the product under development. Horizon Robotics reserves the right to alter or discontinue the product without prior notice.

Disclaimer

The information in this document is only used to help system and software users to use horizon products. The information in this document does not expressly or implicitly authorize others to design or anufacture any integrated circuit based on the information in this document.

The information in this document is subject to change without notice. While the accuracy of the contents of this document has been ensured as far as possible, all statements, information and suggestions in this document do not constitute any warranty, representation or warranty, express or implied.

All information in this document is provided "as is". Horizon makes no warranty, representation or warranty, express or implied, as to the merchantability, applicability and non infringement of any third party intellectual property rights of its products for any particular purpose. Horizon shall not be liable for any liability arising from the use of the products, including but not limited to direct or indirect damages.

Buyer and other parties that are developing products based on horizon (hereinafter collectively referred to as "user") understand and agree that the user shall bear the responsibility of independent analysis, evaluation and judgment in the design of product application. Users are fully responsible for the safety of their applications (and all horizon products used in their applications) and ensure compliance with all applicable regulations, laws and other requirements. The "typical" parameters provided in horizon product brief and product specifications may vary from application to application and actual performance may vary over time. All operating parameters, including "typical" parameters, must be verified by the user himself for each user application.

User agrees to compensate horizon and its representatives in full for any claims, damages, costs, losses and / or liabilities arising from user's unauthorized use of horizon products or non-compliance with the terms of this description.

Revision History

The revision history lists the major changes that have occurred between versions of each document. The following table lists the technical content of each document update.

edition	Revision date	Revision notes
0.1	2020-6-23	outline

Table of Content

1 Introduction

1.1 Purpose of writing

This document is mainly about the Xj3 chip solution about the image data processing channel (vio) (camera) and other modules of the actual use of common scenarios and corresponding common problems, better to make the configuration and use more simple and convenient, so that people who have no use basis can use the program provided by Xj3 development board for simple use after reading this document Try to provide more practical information.

1.1.1 Terminology Convention

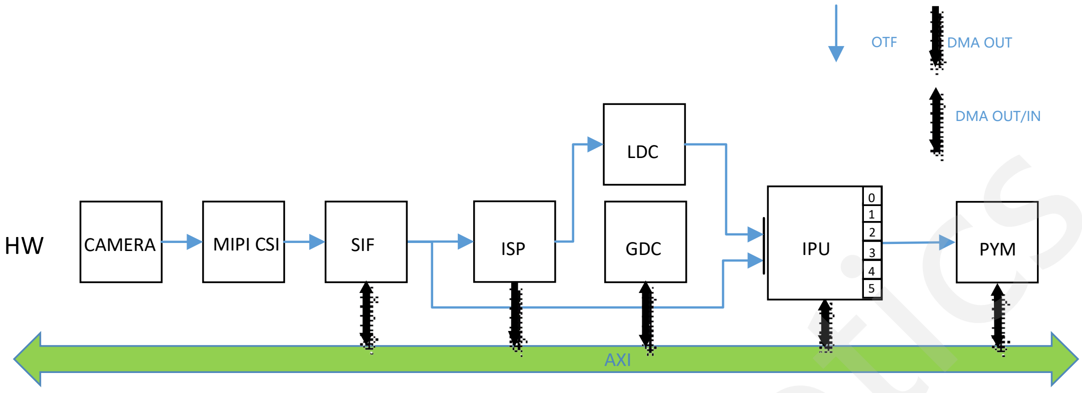
abbreviation	Full name
VIO	Video in/out
ISP	Image Signal Processor
IPU	Image Process Unit
BPU	Brain Process Unit
HAL	Hardware Abstraction Layer
FW	Firmware
PYM	Pyramid
OTF	On The Fly
MIPI	Mobile Industry Processor Interface
CSI	Camera Serial Interface
SIF	Sensor Interface
LDC	Lens Distortion Correction
GDC	Geometric Distortion Correction

1.1.2 Readers and suggestions

This document aims at the application engineers, test engineers and relevant engineering personnel who need to use vio and camera. The document provides the information and general concepts of the software at all levels. If the actual interface development is involved, please refer to the corresponding software development manual.

2 General overview

Camera is the main external source of image data, vio part of the software is a relatively opaque internal software, mainly for providing internal application software to provide relevant images and information, Xj3 chip internal image processing IP information is roughly as follows:



Input mode	IP	Output mode
OTF	MIPI	OTF
OTF/DMA	SIF	OTF/DMA
OTF	ISP	OTF/DMA
OTF	LDC	OTF
DMA	GDC	DMA
OTF/DMA	IPU	OTF/DMA
OTF/DMA	PYM	DMA

3 Camera system

3.1 Sensor debugging guide

3.1.1 Commissioning process

Please follow the process shown in Figure 1-1 for debugging

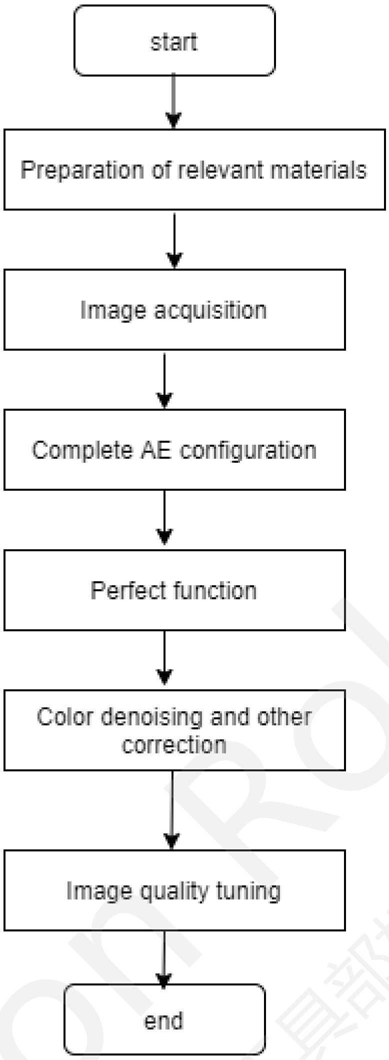

start
Preparation of relevant materials
Image acquisition
Complete AE configuration
Perfect function
Color denoising and other correction
Image quality tuning
end

Figure 1-1 sensor debugging flow chart

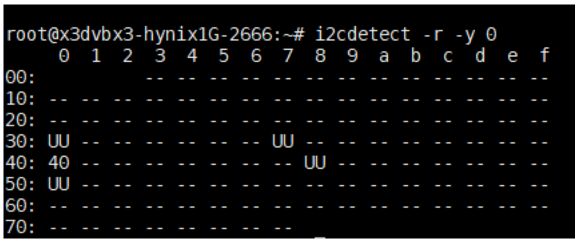
3.1.2 Preparation of materials

- sensor datasheet，It mainly refers to the registers that may need to be referred to during debugging, as well as the process of power up and power down with the sensor;
- Get the initial setting of the sensor from the manufacturer, which is the register configuration that can make the sensor work normally;
- Confirm the I2C slave address of the sensor.

3.1.3 Confirm that the hardware is normal

- Confirm that the power supply of the sensor is normal;
- Confirm which I2C bus the sensor is connected to. Suppose it is connected to I2C bus 0, then:

```
i2cdetect -r -y 0
```



```
root@x3dvbx3-hynix1G-2666:~# i2cdetect -r -y 0
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: UU -- -- -- -- -- -- UU -- -- -- -- -- -- -- --
40: 40 -- -- -- -- -- -- -- UU -- -- -- -- -- -- --
50: UU -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

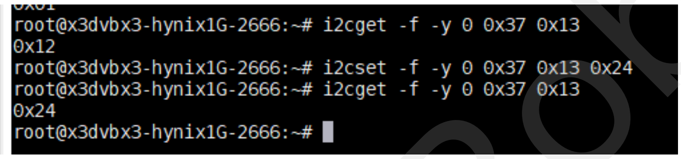
If everything is OK, the slave of the sensor_ Addr can detect;

If you cannot detect:

a. Confirm whether the bus connected to the sensor is 0；

b. Is there a problem with the I2C pull-up resistor on the sensor hardware；

- When the system can detect the sensor, confirm whether the register access to the sensor is normal through I2C tool. For example, read and write a certain register by accessing the sensor at the address of 0x37 connected to bus 0



```
0x01
root@x3dvbx3-hynix1G-2666:~# i2cget -f -y 0 0x37 0x13
0x12
root@x3dvbx3-hynix1G-2666:~# i2cset -f -y 0 0x37 0x13 0x24
root@x3dvbx3-hynix1G-2666:~# i2cget -f -y 0 0x37 0x13
0x24
root@x3dvbx3-hynix1G-2666:~# █
```

If everything is normal, the sensor can work normally。**Note that for different sensors, the values written in some registers may not be the same as those read out (confirm from the sensor datasheet)**。As long as the i2cget / set operation of reading and writing the register can be confirmed to be correct;

3.1.4 Fill template code

In the hbre / camera / utility / sensor directory of the SDK, there are various sensor codes supported by the platform for reference. Each sensor in this directory can support one or more sensor modes, and the corresponding code example can be found by searching keywords such as linear / dol2 / dol3 / PWL.

The simplest template code is as follows. Take ar0144at as an example:

```
//ar144AT_utility.c file

  #include <stdio.h>

  #include <stdlib.h>

  #include <string.h>

  #include <stdint.h>

  #include <assert.h>

  #include <getopt.h>

  #include <fcntl.h>

  #include <unistd.h>
```

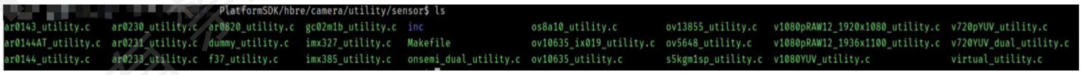
```
#include <sys/ioctl.h>
#include <linux/i2c-dev.h>

#include "logging.h"
#include "hb_cam_utility.h"
#include "inc/ar0144AT_setting.h" //initial setting Array file
#include "inc/sensor_effect_common.h"
// When debugging, the camera can be powered on by default and not controlled in the code
int sensor_poweron(sensor_info_t *sensor_info) {
  int ret = RET_OK;
  if(sensor_info->power_mode) {
                    for(gpio = 0; gpio < sensor_info->gpio_num; gpio++) {
                              if(sensor_info->gpio_pin[gpio] >= 0) {
      ...
     // Omit remaining code
 }

//sensor Initialization function, write initial setting to sensor
//ar144_init_setting the initialization parameters are saved in
int sensor_init(sensor_info_t *sensor_info) {
  int ret = RET_OK;
  int setting_size = 0;

  setting_size = sizeof(ar144_init_setting)/sizeof(uint32_t)/2;
  pr_debug("sensor_name %s, setting_size = %d\n", sensor_info->sensor_name,
  setting_size);
  // Write parameters to sensor through I2C. Note that the third parameter of
  the function represents the data width of sensor register, and 1 represents
  the data width of sensor register
  // A byte
  ret = camera_write_array(sensor_info->bus_num, sensor_info->sensor_addr, 1,
  setting_size, ar144_init_setting);
  if (ret < 0) {
    pr_debug("%d : init %s fail\n", __LINE__, sensor_info->sensor_name);
    return ret;
  }
  return ret;
}
//sensor stream on function to control the sensor to work normally
int sensor_start(sensor_info_t *sensor_info) {
  int ret = RET_OK;
  int setting_size = 0;
  setting_size = sizeof(ar144_stream_on_setting)/sizeof(uint32_t)/2;
  printf(" start sensor_name %s, setting_size = %d\n", sensor_info->sensor_name,
  setting_size);
  ret = camera_write_array(sensor_info->bus_num, sensor_info->sensor_addr, 1,
  setting_size, ar144_stream_on_setting);
  if(ret < 0) {
    pr_debug("start %s fail\n", sensor_info->sensor_name);
    return ret;
  }
  return ret;
}
// Control sensor starts to stop data
int sensor_stop(sensor_info_t *sensor_info)
{
  int ret = RET_OK;
  int setting_size = 0;
  setting_size = sizeof(ar144_stream_off_setting)/sizeof(uint32_t)/2;
  printf(" stop sensor_name %s, setting_size = %d\n", sensor_info->sensor_name,
  setting_size);
  ret = camera_write_array(sensor_info->bus_num, sensor_info->sensor_addr, 1,
  setting_size, ar144_stream_off_setting);
  if(ret < 0) {
    pr_debug("start %s fail\n", sensor_info->sensor_name);
    return ret;
  }
  return ret;
}
//sensor power down function, turn off the sensor power
int sensor_poweroff(sensor_info_t *sensor_info){
  int ret = RET_OK;
  return ret;
}
int sensor_deinit(sensor_info_t *sensor_info) {
  int ret = RET_OK;
  return ret;
}
//sensor callback structure
sensor_module_t ar144AT = {
  .module = "ar144AT",
  .init = sensor_init,
  .start = sensor_start,
  .stop = sensor_stop,
  .deinit = sensor_deinit,
  .power_on = sensor_poweron,
  .power_off = sensor_poweroff,
};
```

// Ar144 related configuration, ar144at_ Setting. H file

```
  #ifndef UTILITY_SENSOR_INC_AR144_SETTING_H_
  #define UTILITY_SENSOR_INC_AR144_SETTING_H_
  #ifdef __cplusplus
  extern "C" {
  #endif
  // Initialize the sensor parameter, which is provided by the sensor manufacturer
  static uint32_t ar144_init_setting[] = {
     0x12, 0x60,
    0x48, 0x85,
    0x48, 0x05,
    0x0E, 0x11,
    0x0F, 0x14,
    0x10, 0x48,
    //... Omit remaining parameters.....
};
// Stream on parameter, provided by sensor manufacturer
static uint32_t ar144_stream_on_setting[] = {
    0x12, 0x20,
  0x48, 0x85,
  0x48, 0x05,
};
// Stream off parameter, provided by sensor manufacturer
static uint32_t ar144_stream_off_setting[] = {
    0x12, 0x60
};

#ifdef __cplusplus
}
#endif
#endif  //UTILITY_SENSOR_INC_AR144_SETTING_H_
```

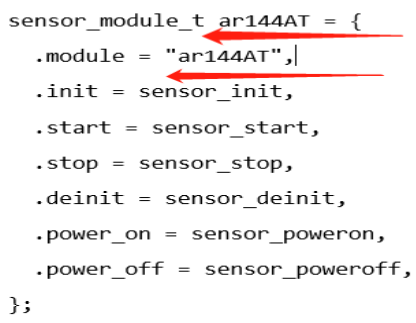
3.1.5 Code description

- After the sensor code is completed, copy it to SDK / hbre / camera / utility / sensor



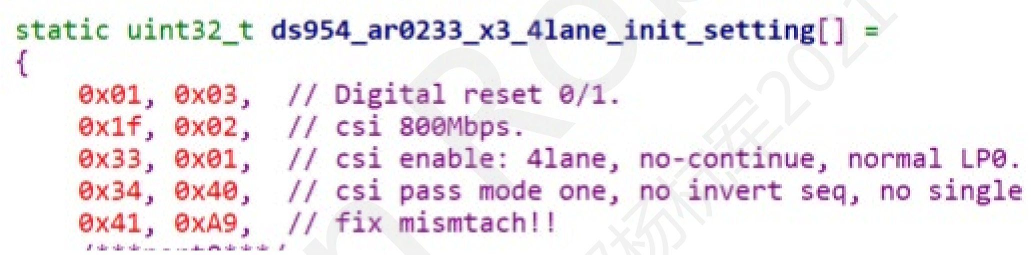
```
[redacted] PlatformSDK/hbre/camera/utility/sensor$ ls
ar0143_utility.c    ar0230_utility.c  ar0820_utility.c  gc02m1b_utility.c  inc                    os8a10_utility.c         ov13855_utility.c   v1080pRAW12_1920x1080_utility.c  v720pYUV_utility.c
ar0144AT_utility.c  ar0231_utility.c  dummy_utility.c   imx327_utility.c   Makefile               ov10635_ix019_utility.c  ov5648_utility.c    v1080pRAW12_1936x1100_utility.c  v720YUV_dual_utility.c
ar0144_utility.c    ar0233_utility.c  f37_utility.c     imx385_utility.c   onsemi_dual_utility.c  ov10635_utility.c        s5kgm1sp_utility.c  v1080YUV_utility.c               virtual_utility.c
```

Note that the name of the file needs to conform to the specification, XXX_ Utility. C, where XXX is the sensor name string, which must correspond to the module name in the code; for example, if the sensor name definition is ar144at, then the code file must be ar144_ Utility. C, the module name in the file must be consistent with this:



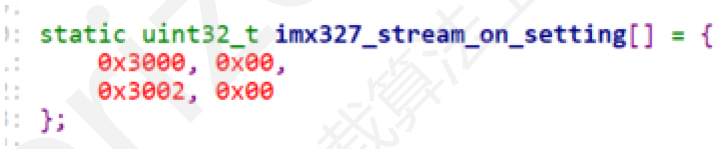
```
sensor_module_t ar144AT = {
  .module = "ar144AT",
  .init = sensor_init,
  .start = sensor_start,
  .stop = sensor_stop,
  .deinit = sensor_deinit,
  .power_on = sensor_poweron,
  .power_off = sensor_poweroff,
};
```

- After completion, the compiled version will generate a so file, such as libxxxx.so Where XXX is the name of the sensor defined in the code. For example, libar144 AT.so The file generated in the corresponding out path will also be packaged into the compiled disk.img Medium;
- After the system is started, the so file can be found under / lib / sensorlib
- The configuration meaning of parameters is as follows：



```
static uint32_t ds954_ar0233_x3_4lane_init_setting[] =
{
    0x01, 0x03,  // Digital reset 0/1.
    0x1f, 0x02,  // csi 800Mbps.
    0x33, 0x01,  // csi enable: 4lane, no-continue, normal LP0.
    0x34, 0x40,  // csi pass mode one, no invert seq, no single
    0x41, 0xA9,  // fix mismtach!!
```

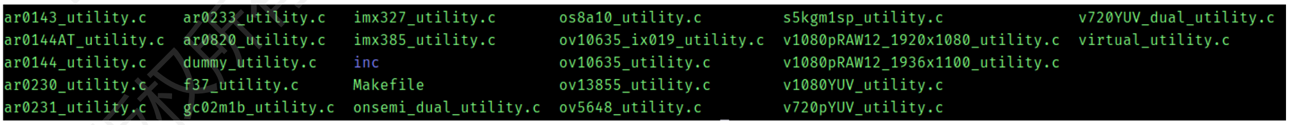
Note that the register address here is 1-byte. For some sensors, the register address can be 2-byte, which is also similar:



```
static uint32_t imx327_stream_on_setting[] = {
    0x3000, 0x00,
    0x3002, 0x00
};
```

3.2 Camera support

Some sensors supported by default (different SDKs may change) can be found under Platform SDK / hbre / camera/utility/sensor



```
ar0143_utility.c    ar0233_utility.c   imx327_utility.c        os8a10_utility.c          s5kgm1sp_utility.c                 v720YUV_dual_utility.c
ar0144AT_utility.c  ar0820_utility.c   imx385_utility.c        ov10635_ix019_utility.c   v1080pRAW12_1920x1080_utility.c   virtual_utility.c
ar0144_utility.c    dummy_utility.c    inc                     ov10635_utility.c         v1080pRAW12_1936x1100_utility.c
ar0230_utility.c    f37_utility.c      Makefile                ov13855_utility.c         v1080YUV_utility.c
ar0231_utility.c    gc02m1b_utility.c  onsemi_dual_utility.c   ov5648_utility.c          v720pYUV_utility.c
```

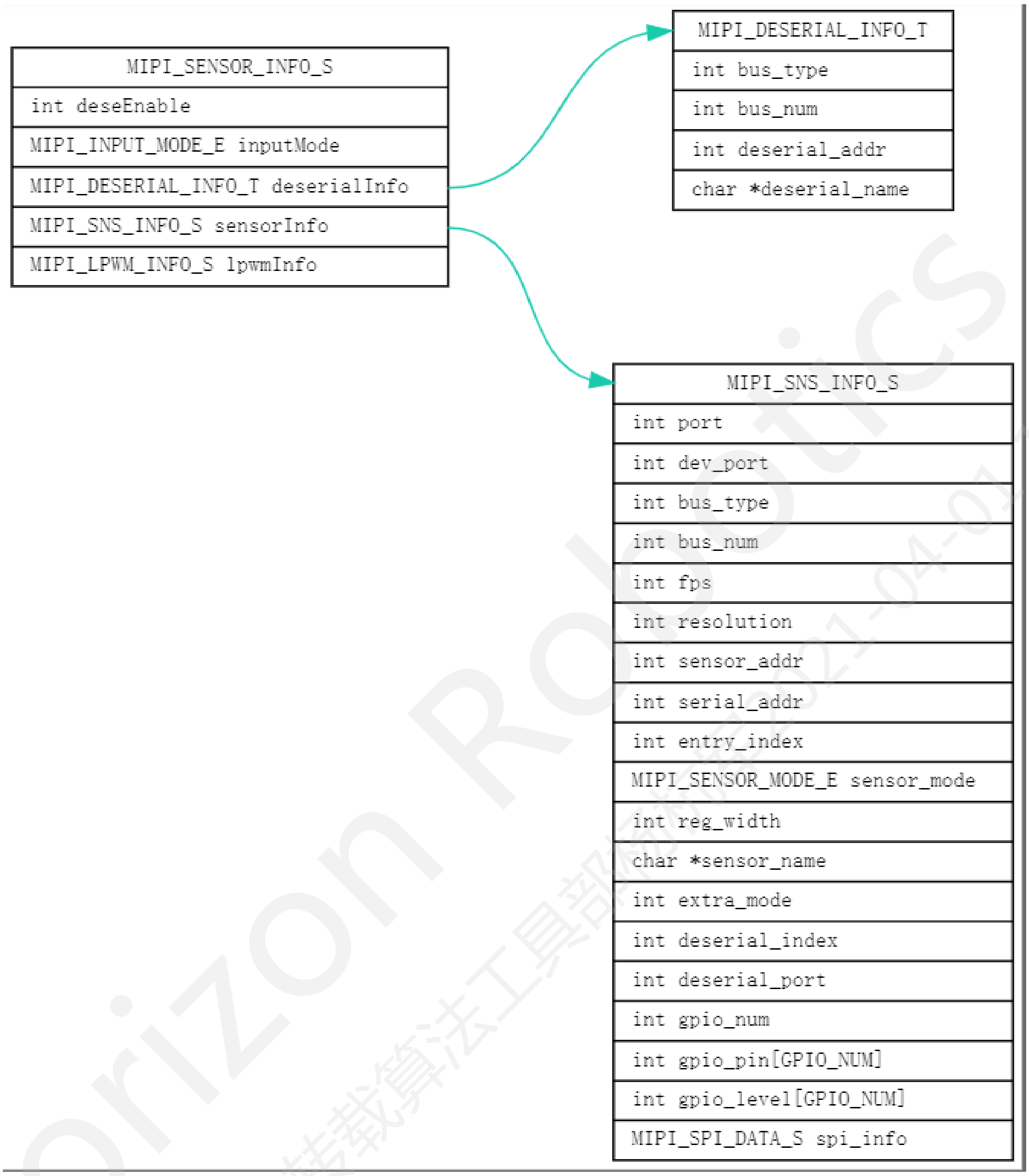
4 Vin section

The camera code generation mentioned above will be generated in / lib / sensorlib after system compilation libxxx.so Library file, waiting to be called.

4.1 Camera VIN code configuration

```
MIPI_SENSOR_INFO_S SENSOR_4LANE_AR0144_30FPS_12BIT_720P_954_INFO =
{
    .deseEnable = 1,
        .inputMode = INPUT_MODE_MIPI,
        .deserialInfo = {
                       .bus_type = 0,
                       .bus_num = 4,
                       .deserial_addr = 0x3d,
                       .deserial_name = "s954",
                       },
        .sensorInfo = {
                     .port = 0,
                     .dev_port = 0,
                     .bus_type = 0,
                     .bus_num = 4,
                     .fps = 30,
                     .resolution = 720,
                     .sensor_addr = 0x10,
                     .serial_addr = 0x18,
                     .entry_index = 1,
                     .sensor_mode = NORMAL_M,
                     .reg_width = 16,
                     .sensor_name = "ar0144AT",
                     .deserial_index = 0,
                     .deserial_port = 0}
};
```

The structure is described in the < x3j3 platform aiot media system interface manual >, which actually indicates a series of information needed for camera control, such as the name of the sensor (used to match so files), the I2C address, which I2C bus is connected to, and so on. Corresponding structure:



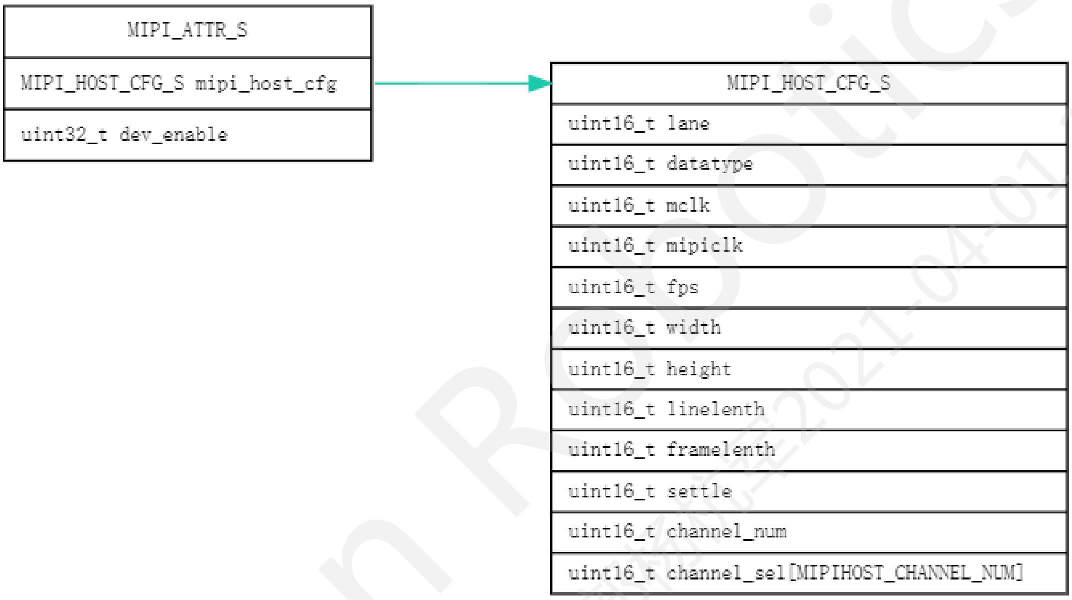
MIPI_SENSOR_INFO_S
- int deseEnable
- MIPI_INPUT_MODE_E inputMode
- MIPI_DESERIAL_INFO_T deserialInfo
- MIPI_SNS_INFO_S sensorInfo
- MIPI_LPWM_INFO_S lpwmInfo

MIPI_DESERIAL_INFO_T
- int bus_type
- int bus_num
- int deserial_addr
- char *deserial_name

MIPI_SNS_INFO_S
- int port
- int dev_port
- int bus_type
- int bus_num
- int fps
- int resolution
- int sensor_addr
- int serial_addr
- int entry_index
- MIPI_SENSOR_MODE_E sensor_mode
- int reg_width
- char *sensor_name
- int extra_mode
- int deserial_index
- int deserial_port
- int gpio_num
- int gpio_pin[GPIO_NUM]
- int gpio_level[GPIO_NUM]
- MIPI_SPI_DATA_S spi_info

4.2 Mipi configuration corresponding to camera

```
MIPI_ATTR_S MIPI_4LANE_SENSOR_AR0144_30FPS_12BIT_720P_954_ATTR = {
        .mipi_host_cfg = {
                        4,    /* lane */
                        0x2c, /* datatype */
                        24,   /* mclk       */
                        1600, /* mipiclk */
                        30,   /* fps */
                        1280, /* width      */
                        720,  /*height */
                        1488, /* linlength */
                        1600, /* framelength */
                        30,   /* settle */
                        4,
                        {0, 1, 2, 3}
                        },
          .dev_enable = 0         /*  mipi dev enable */
};
The corresponding structures are as follows:
```



```
MIPI_ATTR_S
MIPI_HOST_CFG_S mipi_host_cfg
uint32_t dev_enable

MIPI_HOST_CFG_S
uint16_t lane
uint16_t datatype
uint16_t mclk
uint16_t mipiclk
uint16_t fps
uint16_t width
uint16_t height
uint16_t linelenth
uint16_t framelenth
uint16_t settle
uint16_t channel_num
uint16_t channel_sel[MIPIHOST_CHANNEL_NUM]
```

4.3 Reference code and process description

MIPI_SENSOR_INFO_S   snsInfo;

MIPI_ATTR_S   mipiAttr;

int DevId = 0, mipiIdx = 1;

int bus = 1, port = 0, serdes_index = 0, serdes_port = 0;

int ExtraMode= 0;

memset(snsInfo, 0, sizeof(MIPI_SENSOR_INFO_S));

memset(mipiAttr, 0, sizeof(MIPI_ATTR_S));

snsInfo.sensorInfo.bus_num = 0;

snsInfo.sensorInfo.bus_type = 0;

snsInfo.sensorInfo.entry_num = 0;

snsInfo.sensorInfo.sensor_name = "ar144";

```
snsInfo.sensorInfo.reg_width = 16;
snsInfo.sensorInfo.sensor_mode = NORMAL_M;
snsInfo.sensorInfo.sensor_addr = 0x36;

mipiAttr.dev_enable = 1;
mipiAttr.mipi_host_cfg.lane = 4;
mipiAttr.mipi_host_cfg.datatype = 0x2c;
mipiAttr.mipi_host_cfg.mclk = 24;
mipiAttr.mipi_host_cfg.mipiclk = 891;
mipiAttr.mipi_host_cfg.fps = 25;
mipiAttr.mipi_host_cfg.width = 1952;
mipiAttr.mipi_host_cfg.height = 1097;
mipiAttr.mipi_host_cfg->linelenth = 2475;
mipiAttr.mipi_host_cfg->framelenth = 1200;
mipiAttr.mipi_host_cfg->settle = 20;

HB_MIPI_SetBus(snsInfo, bus);
HB_MIPI_SetPort(snsinfo, port);
HB_MIPI_SensorBindSerdes(snsinfo, sedres_index, sedres_port);
HB_MIPI_SensorBindMipi(snsinfo,   mipiIdx);
HB_MIPI_SetExtraMode (snsinfo,   ExtraMode);
ret = HB_MIPI_InitSensor(DevId, snsInfo);
if(ret < 0) {
    printf("HB_MIPI_InitSensor error!\n");
    return ret;
}
ret = HB_MIPI_SetMipiAttr(mipiIdx, mipiAttr);
if(ret < 0) {
    printf("HB_MIPI_SetMipiAttr error! do sensorDeinit\n");
    HB_MIPI_SensorDeinit(DevId);
    return ret;
}
```

```
ret = HB_MIPI_ResetSensor(DevId);
if(ret < 0) {
    printf("HB_MIPI_ResetSensor error! do mipi deinit\n");
    HB_MIPI_DeinitSensor(DevId);
    HB_MIPI_Clear(mipiIdx);
    return ret;
}
ret = HB_MIPI_ResetMipi(mipiIdx);
if(ret < 0) {
    printf("HB_MIPI_ResetMipi error!\n");
    HB_MIPI_UnresetSensor(DevId);
    HB_MIPI_DeinitSensor(DevId);
    HB_MIPI_Clear(mipiIdx);
    return ret;
}
HB_MIPI_UnresetSensor(DevId);
HB_MIPI_UnresetMipi(mipiIdx);
HB_MIPI_DeinitSensor(DevId);
HB_MIPI_Clear(mipiIdx);
```

Two VIN related configurations are involved in this sample code, Mipi_ SENSOR_ INFO_ S is the sensor related configuration.

For the sensor, through the sensor_ Name, the framework code of sensor can find the corresponding so file under / lib / sensorlib /. By loading the library, the information such as the address / resolution / I2C bus / register width of the sensor is passed into the code of the sensor when calling the library. After that, the sensor can use the information and initialization parameters in the code to initialize the sensor.

The Mipi host corresponding to the sensor is configured through Mipi_ ATTR_ S. here, the resolution / frame rate / number of lanes used / data format of corresponding access sensors of Mipi host are configured. The configuration of Mipi host is based on the sensor configuration, such as resolution / data format / actual number of connected physical lanes;

4.4 Driving interaction with Hal

In VIN, all control related to camera is completed in user mode, while control of Mipi / ISP is completed in driver. The configuration related to user mode is transferred to the driver, which is used by the driver to set the hardware. The relevant device nodes are as follows:

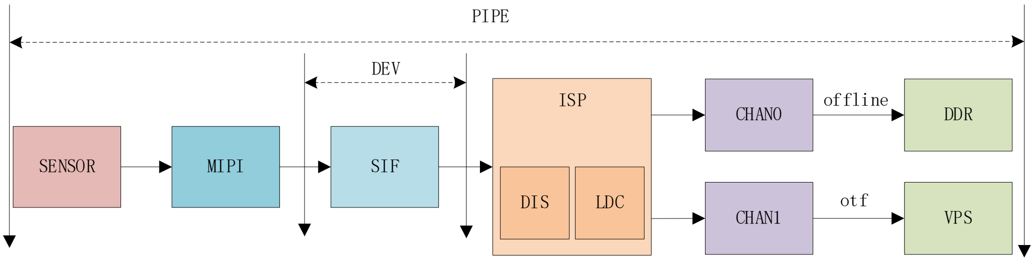


Figure 4-1 Vin software block diagram

- mipi_dev0: The device node is enabled in the configuration, Configure mipi_ dev output。
- mipi_host0~4: Mipi host configuration node，It mainly completes the initialization of Mipi。
- mipi_dphy: Dphy related nodes。
- SIF has two nodes：

  sif_capture: Set the SIF related attribute information, initialize the SIF module, and dump the image from the SIF module。

  sif_ddrin: Set the attribute information/size/format of ddrin node, which is only used in SIF offline ISP scenario, and is responsible for reading memory data to ISP.
- ISP related nodes：

  ac_calib: Calibration effect library settings。

  ac_isp: ISP effect adjustment interface use。

  ac_isp4uf0~7: ISP driven algorithm library sends command to use。

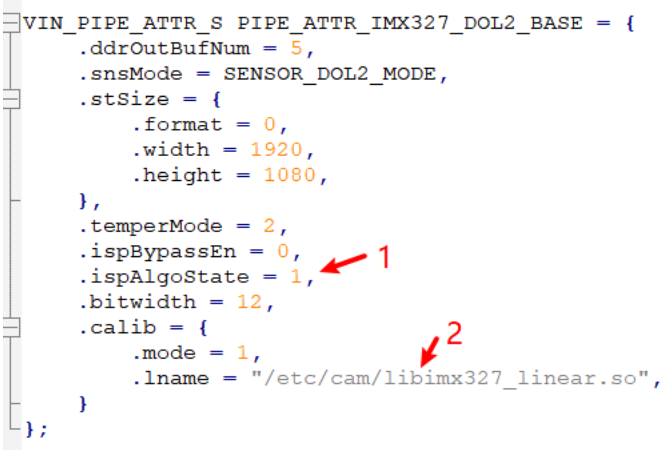
  ac_sbuf0~7: The algorithm library synchronizes some algorithm data with ISP driver through the device node.

  video0~7: ISP v4l2 device node, set size / format / size, Memory mapping interacts with devices through this node.

In VIN, the function of Mipi / SIF is relatively simple. For Mipi, it is actually several nodes abstracted from hardware, which are used for user configuration parameters, so as to set Mipi host to corresponding state, and can accept Mipi data input from sensor;

SIF processes the data received by Mipi host, such as saving the data from different sensors to different DDR addresses;

ISP's function is relatively the most complex. It needs to interact with the sensor / load the corresponding algorithm library / load the corresponding effect library. In the configuration code:



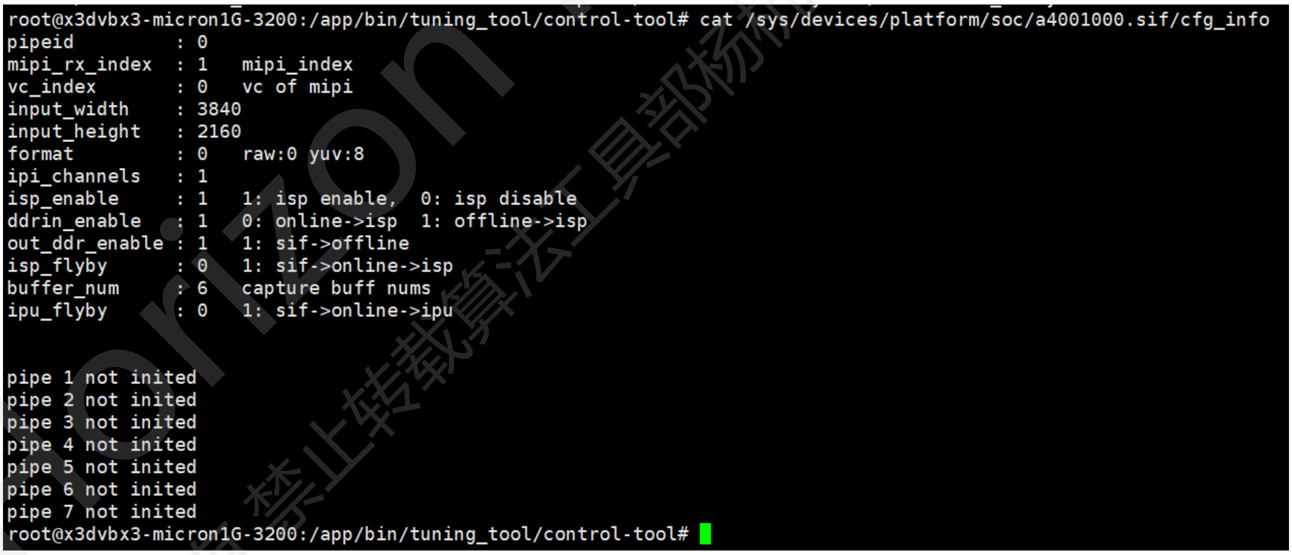
```
VIN_PIPE_ATTR_S PIPE_ATTR_IMX327_DOL2_BASE = {
    .ddrOutBufNum = 5,
    .snsMode = SENSOR_DOL2_MODE,
    .stSize = {
        .format = 0,
        .width = 1920,
        .height = 1080,
    },
    .temperMode = 2,
    .ispBypassEn = 0,
    .ispAlgoState = 1,
    .bitwidth = 12,
    .calib = {
        .mode = 1,
        .lname = "/etc/cam/libimx327_linear.so",
    }
};
```

This tag indicates that if the 3A algorithm is used, lib will be used_ algo.so Second, this is the effect library configured by different sensors, which is used to adjust the sensor effect;

5 VIO debug information

5.1 SIF debug information

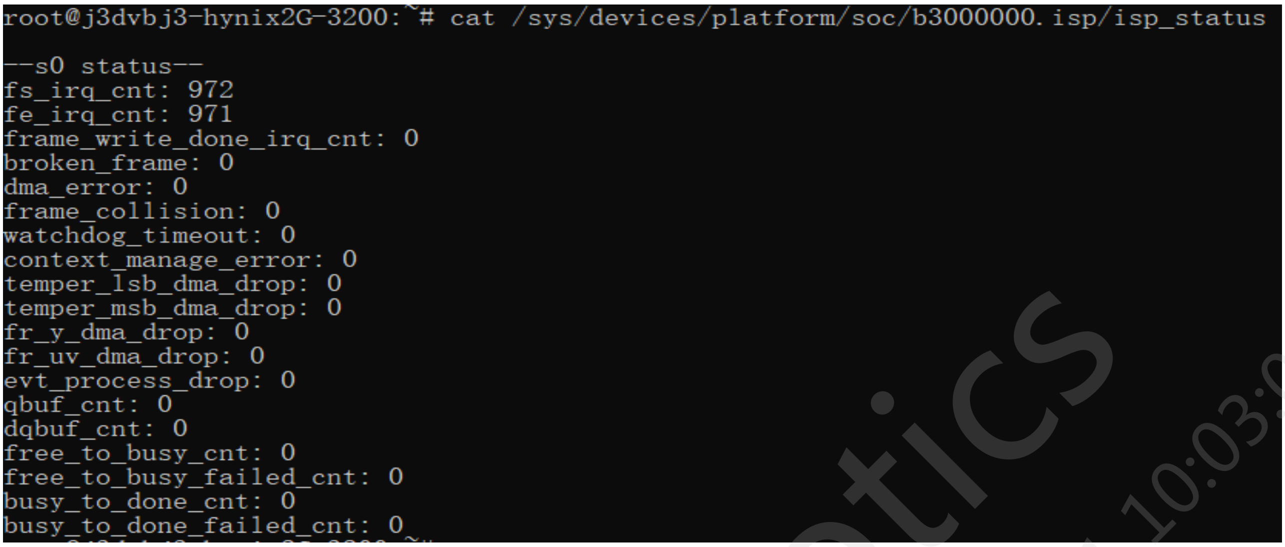
View SIF debugging information：**cat /sys/devices/platform/soc/a4001000.sif/cfg_info**



```
root@x3dvbx3-micron1G-3200:/app/bin/tuning_tool/control-tool# cat /sys/devices/platform/soc/a4001000.sif/cfg_info
pipeid          : 0
mipi_rx_index   : 1   mipi_index
vc_index        : 0   vc of mipi
input_width     : 3840
input_height    : 2160
format          : 0   raw:0 yuv:8
ipi_channels    : 1
isp_enable      : 1   1: isp enable,  0: isp disable
ddrin_enable    : 1   0: online->isp  1: offline->isp
out_ddr_enable  : 1   1: sif->offline
isp_flyby       : 0   1: sif->online->isp
buffer_num      : 6   capture buff nums
ipu_flyby       : 0   1: sif->online->ipu


pipe 1 not inited
pipe 2 not inited
pipe 3 not inited
pipe 4 not inited
pipe 5 not inited
pipe 6 not inited
pipe 7 not inited
root@x3dvbx3-micron1G-3200:/app/bin/tuning_tool/control-tool#
```

5.2 ISP debug information

View ISP debugging information：**cat /sys/devices/platform/soc/b3000000.isp/isp_status**



```
root@j3dvbj3-hynix2G-3200:~# cat /sys/devices/platform/soc/b3000000.isp/isp_status

--s0 status--
fs_irq_cnt: 972
fe_irq_cnt: 971
frame_write_done_irq_cnt: 0
broken_frame: 0
dma_error: 0
frame_collision: 0
watchdog_timeout: 0
context_manage_error: 0
temper_lsb_dma_drop: 0
temper_msb_dma_drop: 0
fr_y_dma_drop: 0
fr_uv_dma_drop: 0
evt_process_drop: 0
qbuf_cnt: 0
dqbuf_cnt: 0
free_to_busy_cnt: 0
free_to_busy_failed_cnt: 0
busy_to_done_cnt: 0
busy_to_done_failed_cnt: 0
```

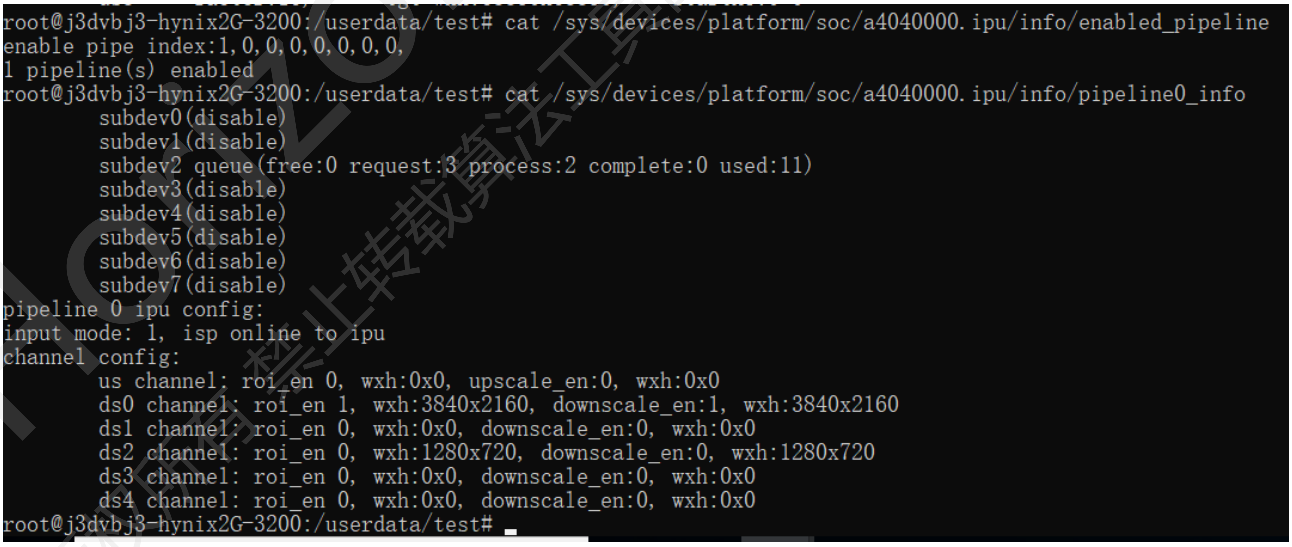
5.3 IPU debug information

To view which pipe enabled currently:

cat /sys/devices/platform/soc/a4040000.ipu/info/enabled_pipeline

Check the configuration of each pipe：

cat /sys/devices/platform/soc/a4040000.ipu/info/pipelinex_info （x Value 0-7）



```
root@j3dvbj3-hynix2G-3200:/userdata/test# cat /sys/devices/platform/soc/a4040000.ipu/info/enabled_pipeline
enable pipe index:1,0,0,0,0,0,0,0,
1 pipeline(s) enabled
root@j3dvbj3-hynix2G-3200:/userdata/test# cat /sys/devices/platform/soc/a4040000.ipu/info/pipeline0_info
        subdev0(disable)
        subdev1(disable)
        subdev2 queue(free:0 request:3 process:2 complete:0 used:11)
        subdev3(disable)
        subdev4(disable)
        subdev5(disable)
        subdev6(disable)
        subdev7(disable)
pipeline 0 ipu config:
input mode: 1, isp online to ipu
channel config:
        us channel: roi_en 0, wxh:0x0, upscale_en:0, wxh:0x0
        ds0 channel: roi_en 1, wxh:3840x2160, downscale_en:1, wxh:3840x2160
        ds1 channel: roi_en 0, wxh:0x0, downscale_en:0, wxh:0x0
        ds2 channel: roi_en 0, wxh:1280x720, downscale_en:0, wxh:1280x720
        ds3 channel: roi_en 0, wxh:0x0, downscale_en:0, wxh:0x0
        ds4 channel: roi_en 0, wxh:0x0, downscale_en:0, wxh:0x0
root@j3dvbj3-hynix2G-3200:/userdata/test# _
```

explain:

Subdev0 corresponds to IPU SRC, and sbudev1 ~ 6 corresponds to IPU US / DS0 ~ DS4. The information in brackets after subdev indicates the number of buffers in each state of the channel.

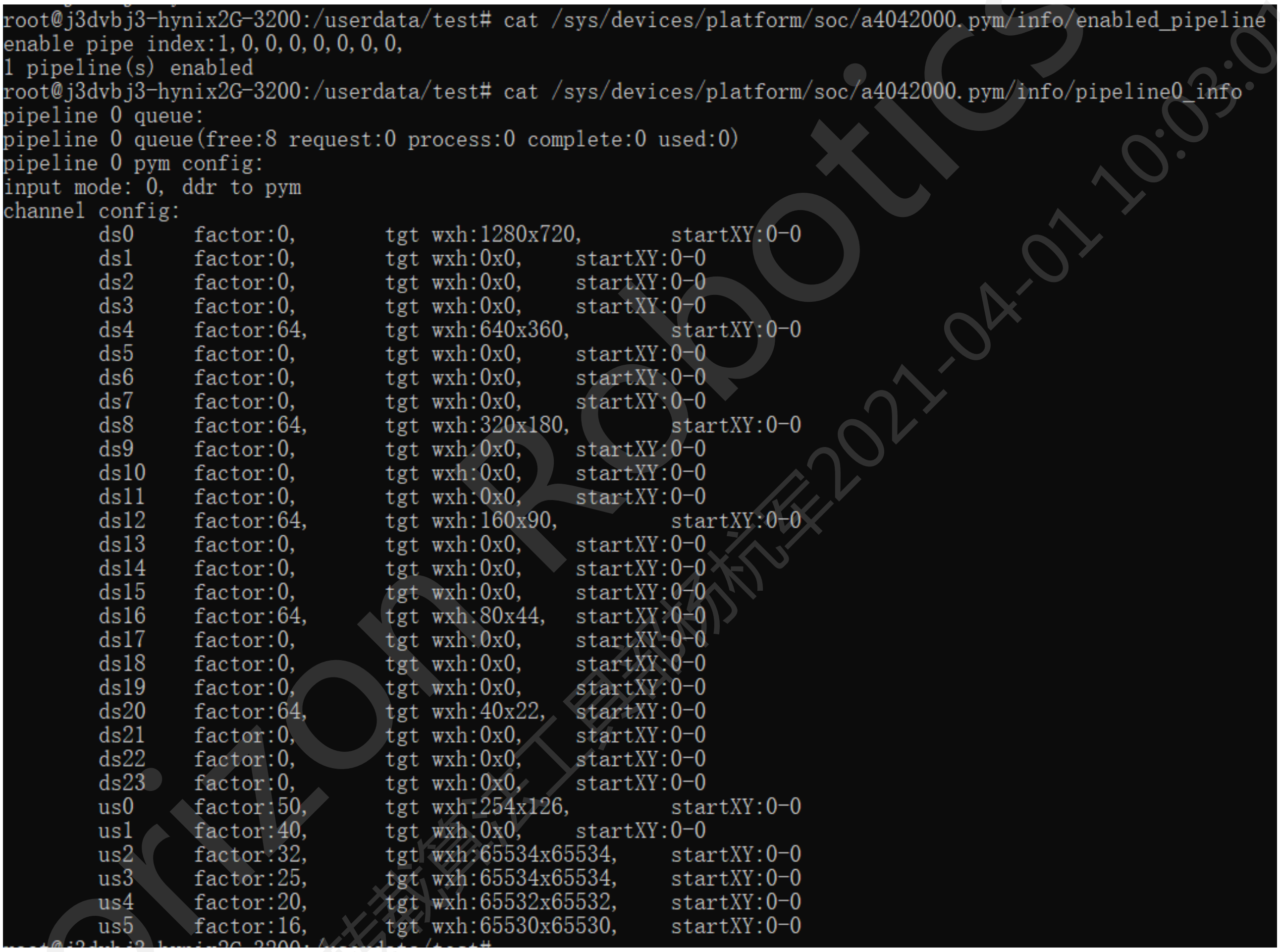
5.4 PYM debug information

To view which pipe enabled currently:

cat /sys/devices/platform/soc/a4042000.pym/info/enabled_pipeline

Check the configuration of each pipe：

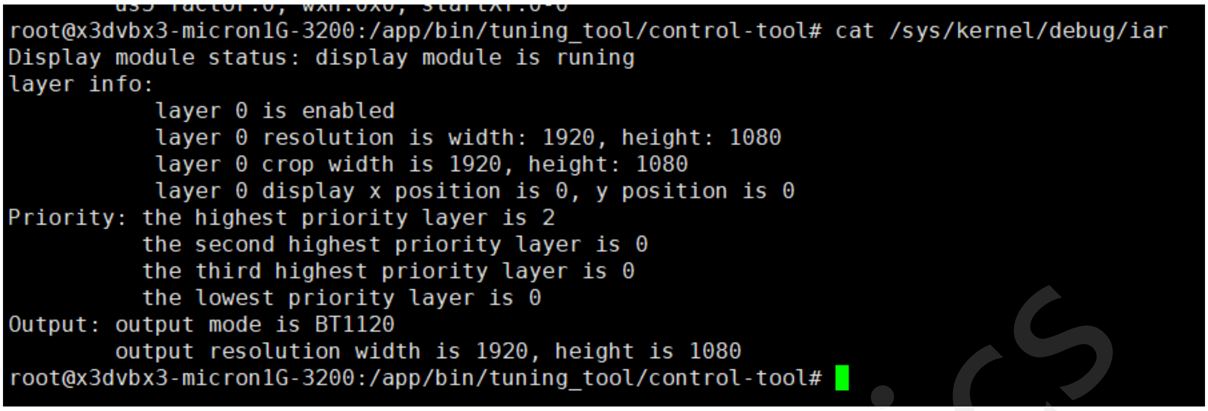
cat /sys/devices/platform/soc/a4042000.pym/info/pipelinex_info （x value 0-7）



```
root@j3dvbj3-hynix2G-3200:/userdata/test# cat /sys/devices/platform/soc/a4042000.pym/info/enabled_pipeline
enable pipe index:1,0,0,0,0,0,0,0,
1 pipeline(s) enabled
root@j3dvbj3-hynix2G-3200:/userdata/test# cat /sys/devices/platform/soc/a4042000.pym/info/pipeline0_info
pipeline 0 queue:
pipeline 0 queue(free:8 request:0 process:0 complete:0 used:0)
pipeline 0 pym config:
input mode: 0, ddr to pym
channel config:
        ds0     factor:0,       tgt wxh:1280x720,       startXY:0-0
        ds1     factor:0,       tgt wxh:0x0,    startXY:0-0
        ds2     factor:0,       tgt wxh:0x0,    startXY:0-0
        ds3     factor:0,       tgt wxh:0x0,    startXY:0-0
        ds4     factor:64,      tgt wxh:640x360,        startXY:0-0
        ds5     factor:0,       tgt wxh:0x0,    startXY:0-0
        ds6     factor:0,       tgt wxh:0x0,    startXY:0-0
        ds7     factor:0,       tgt wxh:0x0,    startXY:0-0
        ds8     factor:64,      tgt wxh:320x180,        startXY:0-0
        ds9     factor:0,       tgt wxh:0x0,    startXY:0-0
        ds10    factor:0,       tgt wxh:0x0,    startXY:0-0
        ds11    factor:0,       tgt wxh:0x0,    startXY:0-0
        ds12    factor:64,      tgt wxh:160x90,         startXY:0-0
        ds13    factor:0,       tgt wxh:0x0,    startXY:0-0
        ds14    factor:0,       tgt wxh:0x0,    startXY:0-0
        ds15    factor:0,       tgt wxh:0x0,    startXY:0-0
        ds16    factor:64,      tgt wxh:80x44,  startXY:0-0
        ds17    factor:0,       tgt wxh:0x0,    startXY:0-0
        ds18    factor:0,       tgt wxh:0x0,    startXY:0-0
        ds19    factor:0,       tgt wxh:0x0,    startXY:0-0
        ds20    factor:64,      tgt wxh:40x22,  startXY:0-0
        ds21    factor:0,       tgt wxh:0x0,    startXY:0-0
        ds22    factor:0,       tgt wxh:0x0,    startXY:0-0
        ds23    factor:0,       tgt wxh:0x0,    startXY:0-0
        us0     factor:50,      tgt wxh:254x126,        startXY:0-0
        us1     factor:40,      tgt wxh:0x0,    startXY:0-0
        us2     factor:32,      tgt wxh:65534x65534,    startXY:0-0
        us3     factor:25,      tgt wxh:65534x65534,    startXY:0-0
        us4     factor:20,      tgt wxh:65532x65532,    startXY:0-0
        us5     factor:16,      tgt wxh:65530x65530,    startXY:0-0
root@j3dvbj3-hynix2G-3200:/userdata/test#
```

5.5 IAR debug information

View IAR debugging information：**cat /sys/kernel/debug/iar**

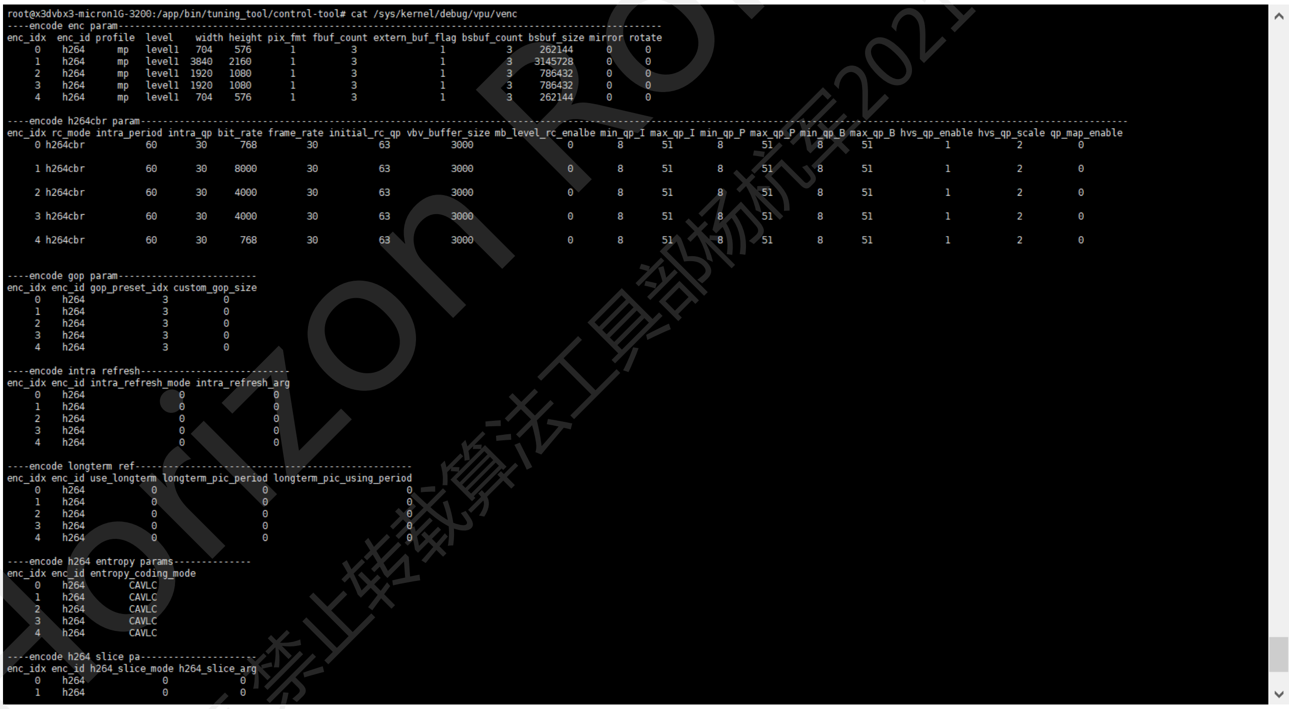


```
        dss factor:0, wxh:0x0, startxy:0-0
root@x3dvbx3-micron1G-3200:/app/bin/tuning_tool/control-tool# cat /sys/kernel/debug/iar
Display module status: display module is runing
layer info:
           layer 0 is enabled
           layer 0 resolution is width: 1920, height: 1080
           layer 0 crop width is 1920, height: 1080
           layer 0 display x position is 0, y position is 0
Priority: the highest priority layer is 2
          the second highest priority layer is 0
          the third highest priority layer is 0
          the lowest priority layer is 0
Output: output mode is BT1120
        output resolution width is 1920, height is 1080
root@x3dvbx3-micron1G-3200:/app/bin/tuning_tool/control-tool# 
```

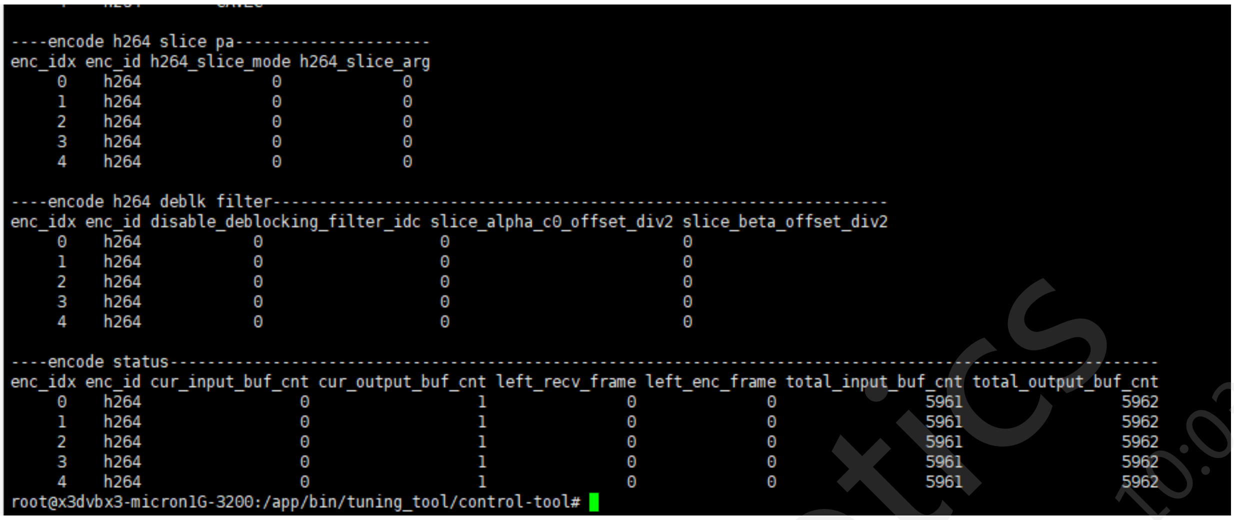
6 VPU debug information

6.1 VENC debug information

View coding information：**cat /sys/kernel/debug/vpu/venc**



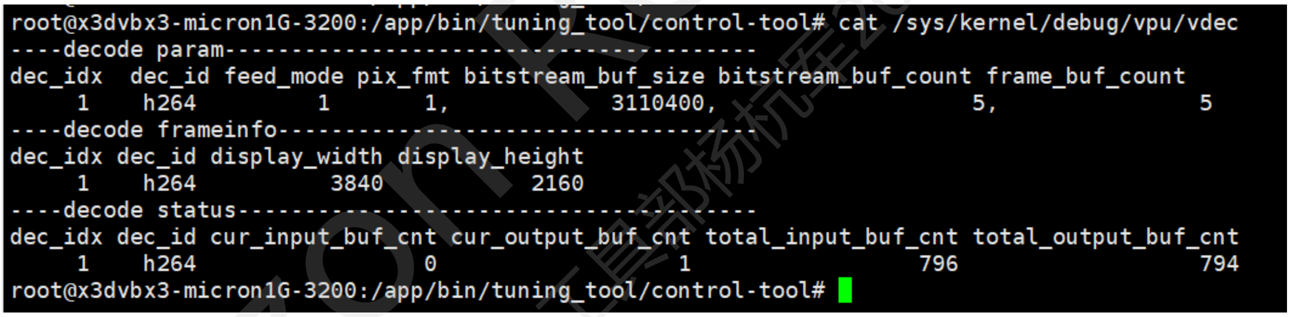
```
root@x3dvbx3-micron1G-3200:/app/bin/tuning_tool/control-tool# cat /sys/kernel/debug/vpu/venc
----encode enc param-------------------------------------------------------------------------------------------
enc_idx  enc_id profile  level    width height pix_fmt fbuf_count extern_buf_flag bsbuf_count bsbuf_size mirror rotate
     0    h264      mp   level1   704    576       1          3               1           3     262144      0      0
     1    h264      mp   level1  3840   2160       1          3               1           3    3145728      0      0
     2    h264      mp   level1  1920   1080       1          3               1           3     786432      0      0
     3    h264      mp   level1  1920   1080       1          3               1           3     786432      0      0
     4    h264      mp   level1   704    576       1          3               1           3     262144      0      0

----encode h264cbr param----------------------------------------------------------------------------------------------------------------------------------------------------------------
enc_idx rc_mode intra_period intra_qp bit_rate frame_rate initial_rc_qp vbv_buffer_size mb_level_rc_enalbe min_qp_I max_qp_I min_qp_P max_qp_P min_qp_B max_qp_B hvs_qp_enable hvs_qp_scale qp_map_enable
     0 h264cbr           60       30      768         30            63            3000                  0        8       51        8       51        8       51             1            2             0

     1 h264cbr           60       30     8000         30            63            3000                  0        8       51        8       51        8       51             1            2             0

     2 h264cbr           60       30     4000         30            63            3000                  0        8       51        8       51        8       51             1            2             0

     3 h264cbr           60       30     4000         30            63            3000                  0        8       51        8       51        8       51             1            2             0

     4 h264cbr           60       30      768         30            63            3000                  0        8       51        8       51        8       51             1            2             0


----encode gop param-------------------------
enc_idx enc_id gop_preset_idx custom_gop_size
     0    h264              3          0
     1    h264              3          0
     2    h264              3          0
     3    h264              3          0
     4    h264              3          0

----encode intra refresh--------------------------
enc_idx enc_id intra_refresh_mode intra_refresh_arg
     0    h264                  0                0
     1    h264                  0                0
     2    h264                  0                0
     3    h264                  0                0
     4    h264                  0                0

----encode longterm ref---------------------------------------------------
enc_idx enc_id use_longterm longterm_pic_period longterm_pic_using_period
     0    h264            0                   0                         0
     1    h264            0                   0                         0
     2    h264            0                   0                         0
     3    h264            0                   0                         0
     4    h264            0                   0                         0

----encode h264 entropy params-------------
enc_idx enc_id entropy_coding_mode
     0    h264          CAVLC
     1    h264          CAVLC
     2    h264          CAVLC
     3    h264          CAVLC
     4    h264          CAVLC

----encode h264 slice pa--------------------
enc_idx enc_id h264_slice_mode h264_slice_arg
     0    h264               0             0
     1    h264               0             0
```



```
----encode h264 slice pa--------------------
enc_idx enc_id h264_slice_mode h264_slice_arg
     0    h264               0              0
     1    h264               0              0
     2    h264               0              0
     3    h264               0              0
     4    h264               0              0

----encode h264 deblk filter------------------------------------------------------------
enc_idx enc_id disable_deblocking_filter_idc slice_alpha_c0_offset_div2 slice_beta_offset_div2
     0    h264             0                           0                          0
     1    h264             0                           0                          0
     2    h264             0                           0                          0
     3    h264             0                           0                          0
     4    h264             0                           0                          0

----encode status--------------------------------------------------------------------------------------------------------------
enc_idx enc_id cur_input_buf_cnt cur_output_buf_cnt left_recv_frame left_enc_frame total_input_buf_cnt total_output_buf_cnt
     0    h264                 0                  1               0              0                5961                 5962
     1    h264                 0                  1               0              0                5961                 5962
     2    h264                 0                  1               0              0                5961                 5962
     3    h264                 0                  1               0              0                5961                 5962
     4    h264                 0                  1               0              0                5961                 5962
root@x3dvbx3-micron1G-3200:/app/bin/tuning_tool/control-tool#
```

6.2 VDEC debug information

View decoding information：**cat /sys/kernel/debug/vpu/vdec**

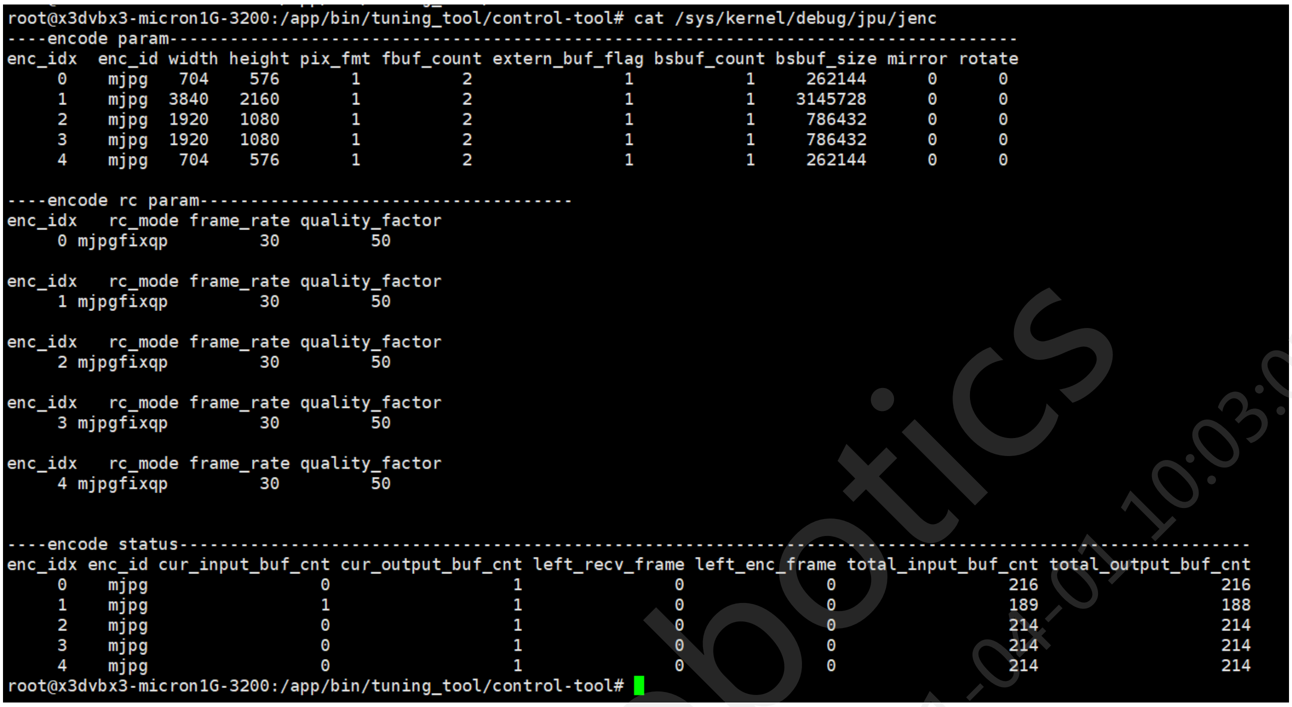


```
root@x3dvbx3-micron1G-3200:/app/bin/tuning_tool/control-tool# cat /sys/kernel/debug/vpu/vdec
----decode param-----------------------------------------
dec_idx  dec_id feed_mode pix_fmt bitstream_buf_size bitstream_buf_count frame_buf_count
     1    h264         1       1,           3110400,                  5,              5
----decode frameinfo-------------------------------------
dec_idx dec_id display_width display_height
     1    h264          3840           2160
----decode status----------------------------------------
dec_idx dec_id cur_input_buf_cnt cur_output_buf_cnt total_input_buf_cnt total_output_buf_cnt
     1    h264                 0                  1                 796                  794
root@x3dvbx3-micron1G-3200:/app/bin/tuning_tool/control-tool#
```

7 JPU debug information

7.1 JENC debug information

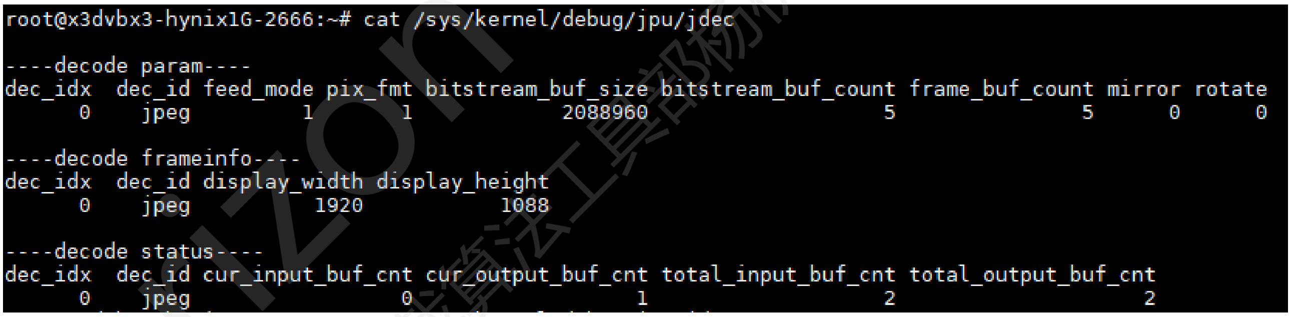
View coding information：**cat /sys/kernel/debug/jpu/jenc**



```
root@x3dvbx3-micron1G-3200:/app/bin/tuning_tool/control-tool# cat /sys/kernel/debug/jpu/jenc
----encode param------------------------------------------------------------------------------------
enc_idx  enc_id width height pix_fmt fbuf_count extern_buf_flag bsbuf_count bsbuf_size mirror rotate
     0    mjpg   704    576       1          2               1           1     262144      0      0
     1    mjpg  3840   2160       1          2               1           1    3145728      0      0
     2    mjpg  1920   1080       1          2               1           1     786432      0      0
     3    mjpg  1920   1080       1          2               1           1     786432      0      0
     4    mjpg   704    576       1          2               1           1     262144      0      0

----encode rc param-----------------------------------
enc_idx   rc_mode frame_rate quality_factor
     0 mjpgfixqp         30          50

enc_idx   rc_mode frame_rate quality_factor
     1 mjpgfixqp         30          50

enc_idx   rc_mode frame_rate quality_factor
     2 mjpgfixqp         30          50

enc_idx   rc_mode frame_rate quality_factor
     3 mjpgfixqp         30          50

enc_idx   rc_mode frame_rate quality_factor
     4 mjpgfixqp         30          50


----encode status-----------------------------------------------------------------------------------------------------------
enc_idx enc_id cur_input_buf_cnt cur_output_buf_cnt left_recv_frame left_enc_frame total_input_buf_cnt total_output_buf_cnt
     0    mjpg                 0                  1               0              0                 216                   216
     1    mjpg                 1                  1               0              0                 189                   188
     2    mjpg                 0                  1               0              0                 214                   214
     3    mjpg                 0                  1               0              0                 214                   214
     4    mjpg                 0                  1               0              0                 214                   214
root@x3dvbx3-micron1G-3200:/app/bin/tuning_tool/control-tool#
```

7.2 JDEC debug information

View decoding information：cat /sys/kernel/debug/jpu/jdec



```
root@x3dvbx3-hynix1G-2666:~# cat /sys/kernel/debug/jpu/jdec

----decode param----
dec_idx  dec_id feed_mode pix_fmt bitstream_buf_size bitstream_buf_count frame_buf_count mirror rotate
      0    jpeg         1       1            2088960                   5               5      0      0

----decode frameinfo----
dec_idx  dec_id display_width display_height
      0    jpeg          1920           1088

----decode status----
dec_idx  dec_id cur_input_buf_cnt cur_output_buf_cnt total_input_buf_cnt total_output_buf_cnt
      0    jpeg                 0                  1                   2                    2
```

8 Common problems and Solutions

8.1 Camera init fail

Phenomenon: I2C configuration failed, sensor initialization failed

Cause analysis:

Test whether the device address can be detected under the bus, i2cdect – R – y [bus]

terms of settlement:

Check the hardware. It may be the wiring problem or the hardware connection problem. Make sure that the device address is seen under the bus before I2C can be configured correctly.

8.2 MIPI error

Phenomenon 1: Mipi check error

Cause analysis:

Mipi did not receive data.

terms of settlement:

- Check whether the Mipi index used by the sensor is the same as the Mipi of check, that is, mipi_init and mipi_start is the same mipi index . It can also be seen from the log.
- It is true that the sensor data is not received. Check whether the sensor configuration is successful. If the platform has requirements, After the sensor_init sequence is configured, no data stream can be output. If there is, an error will be reported. Therefore, the initialization of sensor ensures that the data stream is closed。
- Check the software logic to see if the sensor data stream is opened. If the data stream is not opened, the data will not be received。

Phenomenon 2: Mipi PHY fatal

Cause analysis:

Mipiclk, Lane configuration error

terms of settlement:

- If you modify the Sette value, it will cause PHY fatal error if the setting is wrong. This value can be trial and error, ranging from 0 to 127。

Phenomenon 3: Mipi INT_ST_FRAME_FATAL

Cause analysis:

- There is CRC error in the whole frame data, and errframedata is reported
- The SOF/ EOF is not paired. For example, SOF of the same vc are received twice in a row, but EOF are not received
- The frame number sequence is wrong, for example, the frame number of the previous frame sof is 1, and the frame number received next time becomes 4。

terms of settlement:

Check the configuration of phy clock and Sette. If the parameter configuration is correct, confirm the camera / host initialization timing

Phenomenon 4: Mipi INT_ST_IPI

Cause analysis:

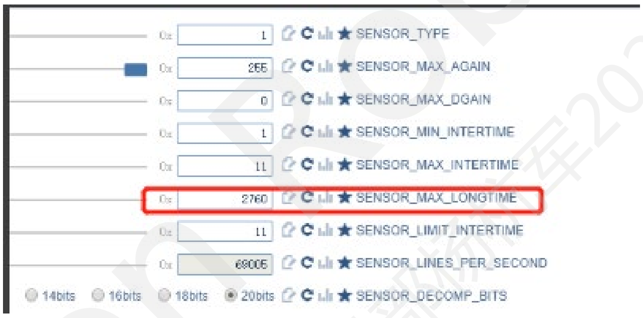
FIFO overflow: FIFO overflow. IPI clock is too slow, or IPI data retrieval is slow. For example, HSD is too large or too long after Hsync is received (too much longer than HSD time calculated according to blanking), which results in IPI not starting data retrieval after FIFO has been filled, or IPI data retrieval speed is lower than that of PPI filling FIFO, resulting in FIFO being filled.

terms of settlement:

Check whether the configuration of width, height, linelenth, framelenth and FPS is correct。

8.3 AE abnormal in HDR mode

phenomenon：sensor_max_longtime == 0



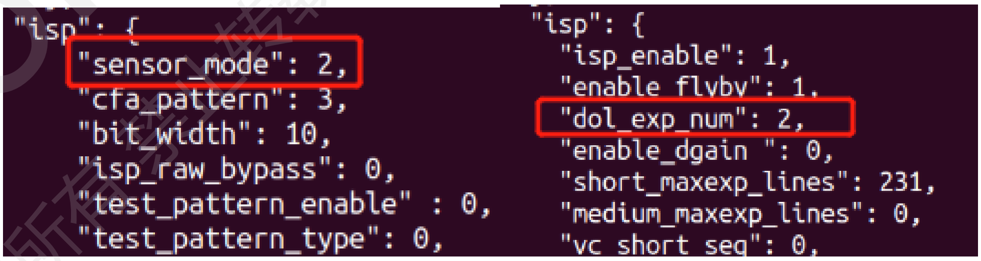
Value	Parameter
1	SENSOR_TYPE
255	SENSOR_MAX_AGAIN
0	SENSOR_MAX_DGAIN
1	SENSOR_MIN_INTERTIME
11	SENSOR_MAX_INTERTIME
2760	SENSOR_MAX_LONGTIME
11	SENSOR_LIMIT_INTERTIME
69005	SENSOR_LINES_PER_SECOND
○ 14bits ○ 16bits ○ 18bits ● 20bits	SENSOR_DECOMP_BITS

Cause analysis:

Mode configuration error;

resolvent:

Modify the configuration file to the current mode.



```
"isp": {
    "sensor_mode": 2,
    "cfa_pattern": 3,
    "bit_width": 10,
    "isp_raw_bypass": 0,
    "test_pattern_enable" : 0,
    "test_pattern_type": 0,
```

```
"isp": {
  "isp_enable": 1,
  "enable_flyby": 1,
  "dol_exp_num": 2,
  "enable_dgain ": 0,
  "short_maxexp_lines": 231,
  "medium_maxexp_lines": 0,
  "vc_short_seq": 0,
```

8.4 Line / gain cannot run to datasheet maximum

Phenomenon:

In the dark scene, the line / gain of AE cannot reach the maximum value

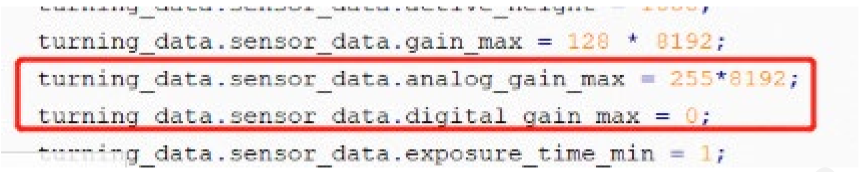
Cause analysis:

The sensor driving limiting amplitude is wrong or tuning limiting amplitude is wrong.

resolvent:

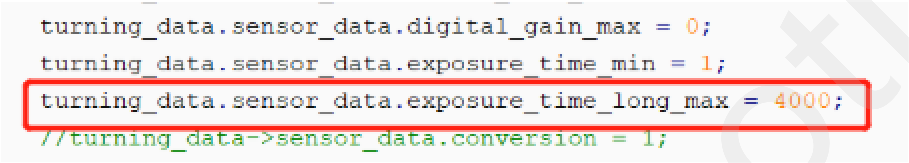
Modify the sensor driver.

Check gain limit amplitude:

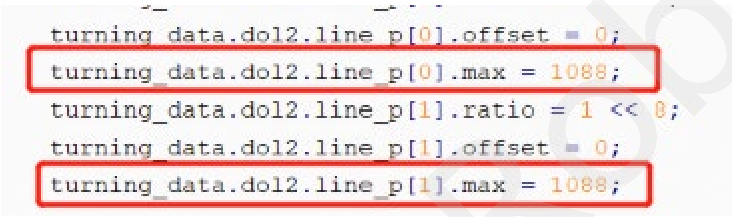


```
turning_data.sensor_data.gain_max = 128 * 8192;
turning_data.sensor_data.analog_gain_max = 255*8192;
turning_data.sensor_data.digital_gain_max = 0;
turning_data.sensor_data.exposure_time_min = 1;
```

Check line limiting amplitude:



```
turning_data.sensor_data.digital_gain_max = 0;
turning_data.sensor_data.exposure_time_min = 1;
turning_data.sensor_data.exposure_time_long_max = 4000;
//turning_data->sensor_data.conversion = 1;
```

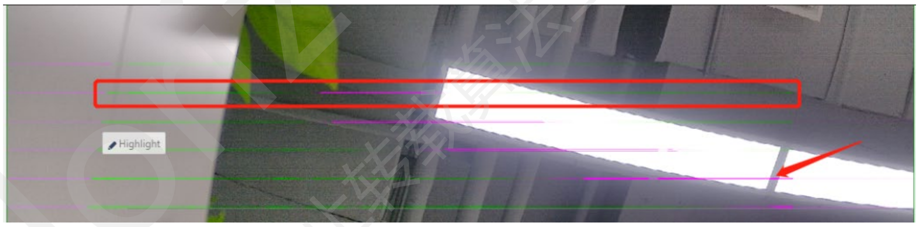


```
turning_data.dol2.line_p[0].offset = 0;
turning_data.dol2.line_p[0].max = 1088;
turning_data.dol2.line_p[1].ratio = 1 << 8;
turning_data.dol2.line_p[1].offset = 0;
turning_data.dol2.line_p[1].max = 1088;
```

8.5 The image is streaked / abnormal color

Phenomenon:

YUV screen is abnormal.

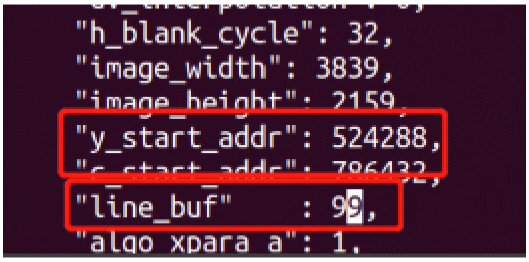


Cause analysis:

LDC does not have enough linebuf or IRAM.

resolvent:

Check line_buf number and IRAM address.

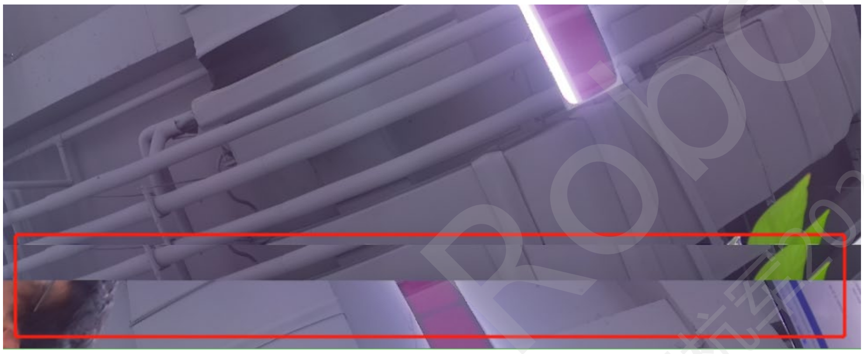


```
"h_blank_cycle": 32,
"image_width": 3839,
"image_height": 2159,
"y_start_addr": 524288,
"c_start_addr": 786432,
"line_buf"     : 99,
"algo_xpara_a": 1,
```

8.6 Abnormal image split (SIF OTF ISP scene in HDR mode)

Phenomenon:

YUV screen is abnormal.

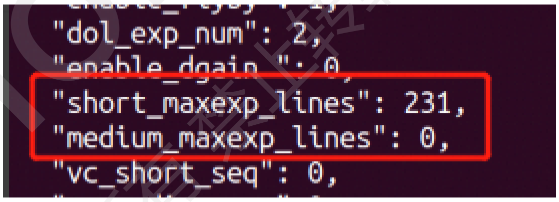


Cause analysis:

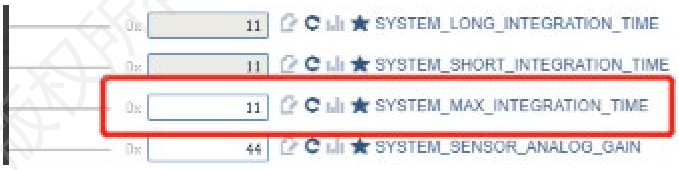
In dol2 mode, the IRAM of SIF is not enough to cache short frame images.

resolvent:

Check the RAM buffer space of SIF, and check the AE short frame exposure time. system_ max_ integration_ Time should be less than short_ maxexp_ Otherwise, an exception may occur.



```
"dol_exp_num": 2,
"enable_dgain": 0,
"short_maxexp_lines": 231,
"medium_maxexp_lines": 0,
"vc_short_seq": 0,
```

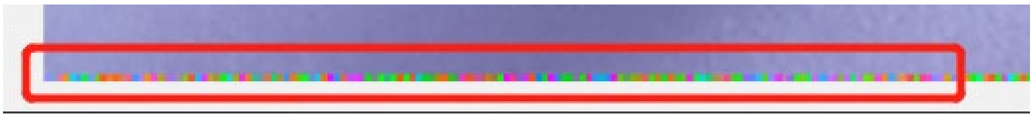


```
0x  11  SYSTEM_LONG_INTEGRATION_TIME
0x  11  SYSTEM_SHORT_INTEGRATION_TIME
0x  11  SYSTEM_MAX_INTEGRATION_TIME
0x  44  SYSTEM_SENSOR_ANALOG_GAIN
```

8.7 There are several abnormal lines under YUV

Phenomenon:

YUV screen is abnormal.

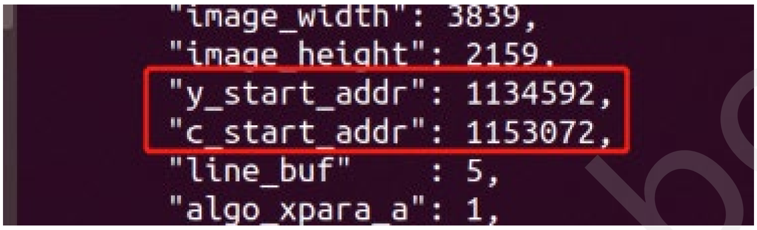


Cause analysis:

The IRAM cache space of LDC is insufficient.

resolvent:

Check the IRAM cache space of LDC.



```
"image_width": 3839,
"image_height": 2159,
"y_start_addr": 1134592,
"c_start_addr": 1153072,
"line_buf"     : 5,
"algo_xpara_a": 1,
```

9 Media module log view

9.1 log level

The relationship between console output log and logcat view log is one-of-a-kind, which is controlled by environment variable loglevel. If export loglevel = 14, all logs higher than debug level (< = 14) will be output to the console. If you want to view debug and higher level logs through logcat, you need to export loglevel = 4.

Console output log		View log through logcat	
CONSOLE_DEBUG_LEVEL	14	ALOG_DEBUG_LEVEL	4
CONSOLE_INFO_LEVEL	13	ALOG_INFO_LEVEL	3
CONSOLE_WARNING_LEVEL	12	ALOG_WARNING_LEVEL	2
CONSOLE_ERROR_LEVEL	11	ALOG_ERROR_LEVEL	1

9.2 Log label

Some log tags are defined in the media module, all tags are as follows:

```
vio-core
vio-devop
ipu
sif
dwe
```

```
gdc
pym
vin
isp
rgn
mipi
vp
vps
venc
vdec
audio
vot
vio-bufmgr
ffmedia
multimedia
```

be careful:

Logs without tags cannot be filtered and will be printed when they meet the log level level level (generally seen in applications or modules without tag).

If you want to tag an application:

1) It can be defined at the beginning of the file #define LOG_TAG “APP”

2) Include related header file #include "logging. h"

3) The prs in the logging. h header file is used for the log printing in the application program_ Macro definition of XXX switch

9.3 Log filtering

Log of each module can be filtered and viewed through logcat. Here is how to filter module related logs. Logcat is an open source command, and other parameters can be explored by yourself.

For example, if you only want to print the VPS part and the log level is higher than debug, and output it to a file, you can do this:

logcat vps:D -f log.txt

If you want to view the logs of multiple modules, you can add filtering at the end. For example, you can view the logs of VPS / VIN modules with a level higher than debug

logcat vps:D vin :D -f log.txt

9.4 Log storage

The kernel log will be saved in the / UserData / log / kernel / directory;

When loglevel is set to 4, the upper level log will be saved in the / UserData / log / usr / directory;