地平线
Horizon Robotics

XJ3图像媒体模块
调试指南

v1.0
2020-6

免责声明

本文档信息仅用于帮助系统和软件使用人员使用地平线产品。本文档信息未以明示或暗示方式授权他人基于本文档信息设计或制造任何集成电路。

本文档中的信息如有更改，恕不另行通知。尽管本文档已尽可能确保内容的准确性，本文档中的所有声明、信息和建议均不构成任何明示或暗示的保证、陈述或担保。

本文档中的所有信息均按“原样”提供。地平线不就其产品在任何特定用途的适销性、适用性以及不侵犯任何第三方知识产权方面做出任何明示或暗示保证、陈述或担保。地平线不承担产品使用所引起的任何责任，包括但不限于直接或间接损失赔偿。

买方和正在基于地平线产品进行开发的其他方（以下统称为“用户”）理解并同意，用户在设计产品应用时应承担独立分析、评估和判断的责任。用户应对其应用（以及用于其应用的所有地平线产品）的安全性承担全部责任，并保证符合所有适用法规、法律和其它规定的要求。地平线产品简介和产品规格中提供的“典型”参数在不同应用下可能会不同，实际性能也可能随时间而变化。所有工作参数，包括“典型”参数，都必须由用户自己针对每项用户应用进行验证。

用户同意如因用户未经授权使用地平线产品或因不遵守本说明中的条款，造成任何索赔、损害、成本、损失和（或）责任，用户将为地平线及其代表提供全额赔偿。

修订记录

修订记录列出了各文档版本间发生的主要更改。下表列出了每次文档更新的技术内容。

版本	修订日期	修订说明
0.1	2020-6-23	提纲
1.0	2021-02	1.0 版本发布

1 引言

1.1 编写目的

撰写该文档主要是对于 XJ3 芯片方案中关于图像数据处理通路（VIO）（Camera）等模块的实际使用的常用场景进行说明以及对应的常见问题整理，更好的让配置使用更简单方便，让无使用基础的人在阅读本文档后能够自主使用 XJ3 开发板提供的程序进行简单的使用，为调试提供更多实际信息。

1.2 术语约定

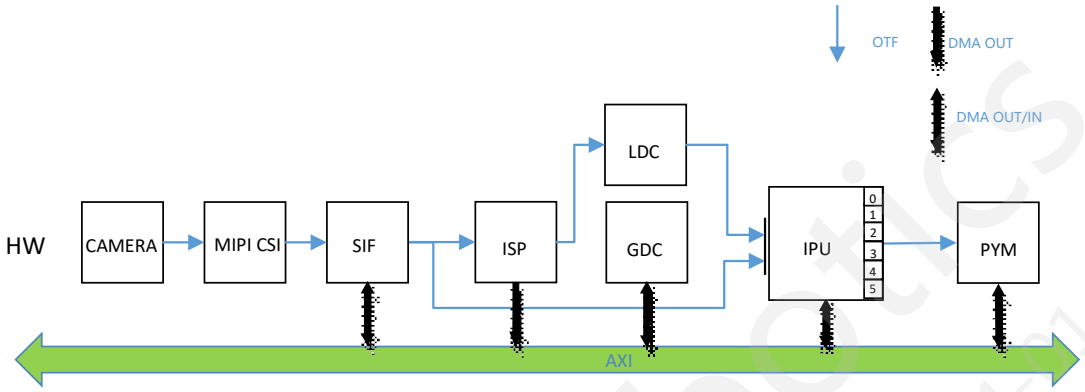
缩写	全称
VIO	Video in/out
ISP	Image Signal Processor
IPU	Image Process Unit
BPU	Brain Process Unit
HAL	Hardware Abstraction Layer
FW	Firmware
PYM	Pyramid
OTF	On The Fly
MIPI	Mobile Industry Processor Interface
CSI	Camera Serial Interface
SIF	Sensor Interface
LDC	Lens Distortion Correction
GDC	Geometric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

1.3 读者对象和阅读建议

该文档针对需要使用到 VIO 以及 Camera 的应用工程师，测试工程师以及相关工程人员等，文档提供了该部分软件使用上的各层面的信息以及大致的概念，如果涉及实际接口开发，请参考对应的软件开发手册。

2 总体概述

Camera 是图像数据的主要外部来源，VIO 部分是一个相对不透明的内部软件，主要面向提供内部应用软件提供相关的图像以及信息，XJ3 芯片内部图像处理 IP 信息大致如下：



输入方式	IP	输出方式
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 系统

3.1 sensor 调试指南

3.1.1 调试流程

请按照如图 1-1 所示流程进行调试

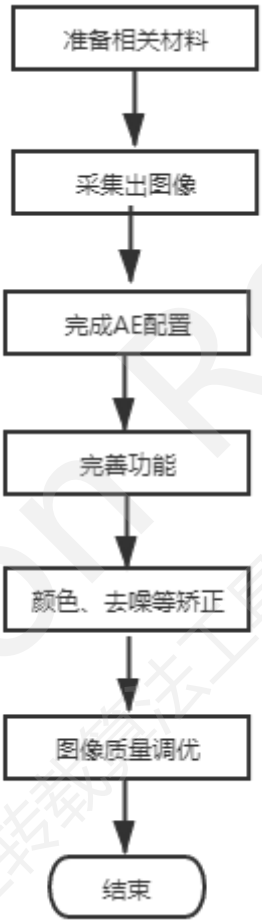


图 1-1 sensor 调试流程图

3.1.2 准备材料

- sensor datasheet，主要是调试时可能需要参考的寄存器，以及与 sensor 上下电的流程；
- 从厂商处拿到 sensor 的 initial setting，就是能让 sensor 正常的工作的寄存器配置；
- 确认一下 sensor 的 i2c slave 地址；

3.1.3 确认硬件正常

- 确认 sensor 的供电正常；
- 确认 sensor 接在哪个 i2c bus 上，假设是接在 i2c bus 0 上，那么：

  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: -- -- -- -- -- -- -- --
```

  如果一切正常，那么 sensor 的 slave_addr 能够 detect 到；

  如果不能 detect 到：

  a. 确认 sensor 的接入的 bus 是不是 0；

  b. sensor 硬件上面 i2c 上拉电阻等是否有问题；

- 在系统能够 detect 到 sensor 的情况下，通过 i2c 工具确认对 sensor 的寄存器访问是否正常。比如访问 bus 0 接的 0x37 地址的 sensor，读写某个寄存器：

```
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:~#
```

  如果一切正常，说明 sensor 能正常工作。**注意，不同的 sensor 可能某些寄存器写的值不一定和读出来的相同(从 sensor datasheet 确认)**。只要能确认对寄存器的读写的 i2cget/set 操作没有出错即可；

3.1.4 填充模板代码

在 SDK 的 hbre/camera/utility/sensor 目录下，有平台已经支持的各类 sensor 代码可以作为参考。该目录下每个 sensor 可支持一种或多种 sensor mode，可通过搜索 linear/dol2/dol3/pwl 等关键字查找对应的代码示例。

最简的模板代码如下，以 ar0144AT 这个 sensor 为例：

//ar144AT_utility.c 文件

```
#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 数组所在文件
#include "inc/sensor_effect_common.h"
//调试时，camera 电可以默认就供上，不在代码中控制
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) {
     ...
    //省略剩余代码
 }

//sensor 初始化函数，往 sensor 写 initial setting
//ar144_init_setting 中即保存了初始化参数
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);
  //通过 i2c 往 sensor 中写参数，注意函数第三个参数表示 sensor 寄存器数据宽度，1 表示
  //一个字节
  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 函数，控制 sensor 开始正常工作
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;
}
//控制 sensor 开始停止数据
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 下电函数，关闭 sensor 电源
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 回调结构体
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 相关配置，ar144AT_setting.h 文件

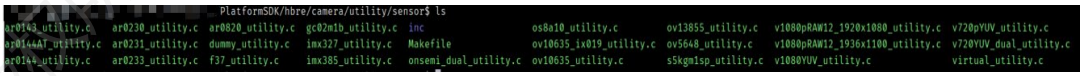
```
  #ifndef UTILITY_SENSOR_INC_AR144_SETTING_H_
  #define UTILITY_SENSOR_INC_AR144_SETTING_H_
  #ifdef __cplusplus
  extern "C" {
  #endif
  //初始化 sensor 参数，由 sensor 厂商提供
```

```
static uint32_t ar144_init_setting[] = {
   0x12, 0x60,
   0x48, 0x85,
   0x48, 0x05,
   0x0E, 0x11,
   0x0F, 0x14,
   0x10, 0x48,
   //...省略剩余参数.....
};
//stream on 参数，由 sensor 厂商提供
static uint32_t ar144_stream_on_setting[] = {
   0x12, 0x20,
  0x48, 0x85,
  0x48, 0x05,
};
//stream off 参数，由 sensor 厂商提供
static uint32_t ar144_stream_off_setting[] = {
   0x12, 0x60
};

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

3.1.5 代码说明

- sensor 代码完成后，复制到 SDK/hbre/camera/utility/sensor 下：



- 注意文件的名称需要符合规范，xxxx_utility.c，其中 xxxx 是 sensor 名称字符串，与代码中 module 名称必须对应；例如 sensor 名称定义是 ar144AT，那么代码文件必须是 ar144_utility.c，在该文件中的 module 名称必须与此一致：

```
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,
};
```

- 完成之后，编译版本，会在生成一个 so 文件，比如 libxxxx.so，其中 xxxx 就是 sensor 的在代码中定义的名称。比如 libar144AT.so，文件生成在对应的 out 路径下，也会打包进编译出的 disk.img 中；
- 在系统启动之后，可以在系统的/lib/sensorlib 下找到该 so 文件
- 参数的配置意义如下：
-

```
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!!
```

注意这里寄存器地址是 1 字节的，有些 sensor，寄存器地址可以是 2 字节的，同样也是类似：

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

3.2 camera 支持

默认支持的部分 sensor(不同 SDK 可能会有变化)可以在 PlatformSDK/hbre/camera/utility/sensor 下找到：

```
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
```

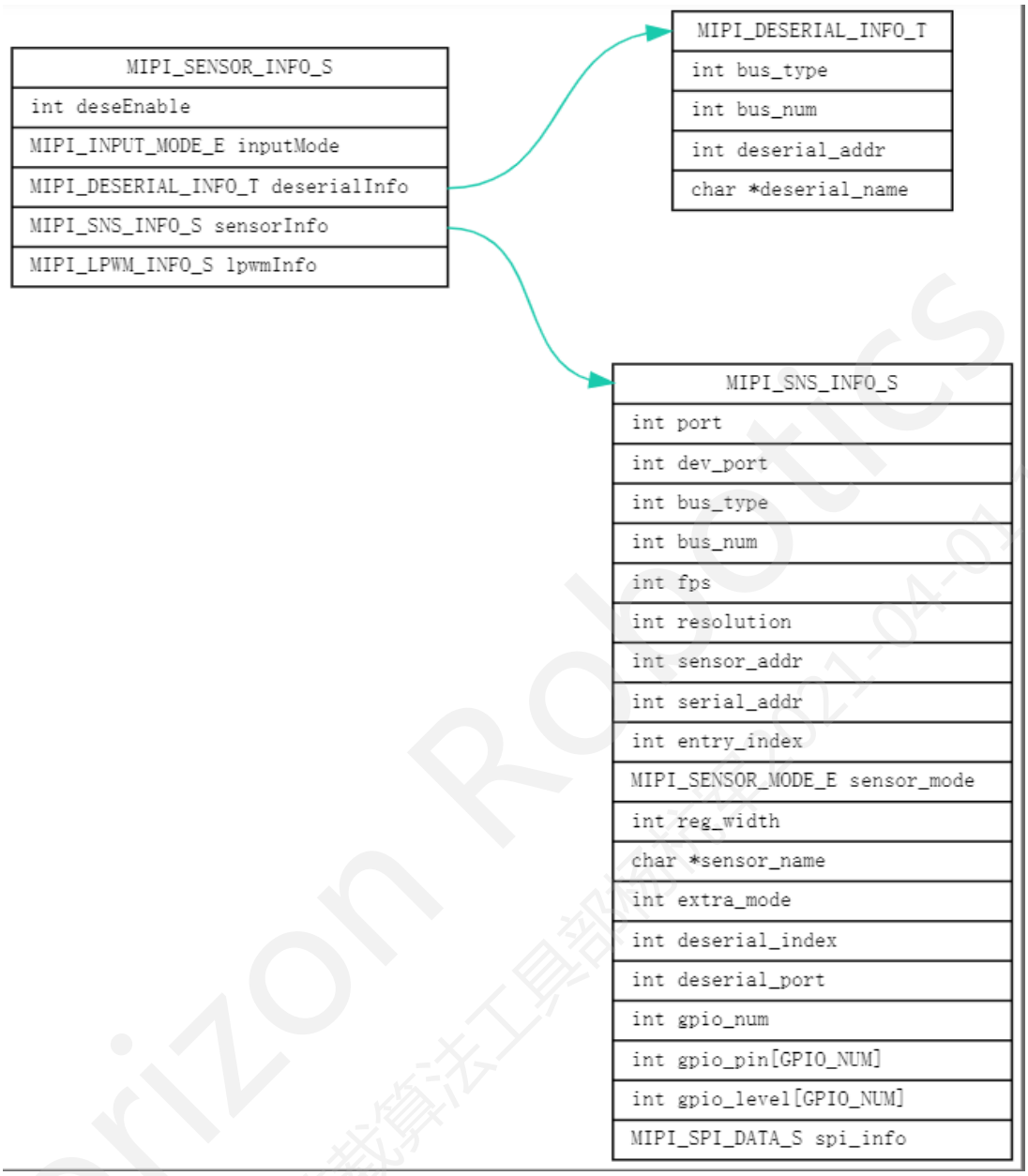
4 VIN 部分

前文提到的 camera 代码生成，经过系统编译后，最终会形成在**/lib/sensorlib** 下的 libxxx.so 库文件，等待被调用。

4.1 camera VIN 代码配置

```
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}
};
```

该结构体的说明在<X3J3 平台 AIOT 媒体系统接口手册>有说明，这里实际上指明了 sensor 的名称(用于匹配 so 文件)、i2c 地址、接在哪个 i2c bus 上等一系列 camera 控制需要的信息。对应结构体：

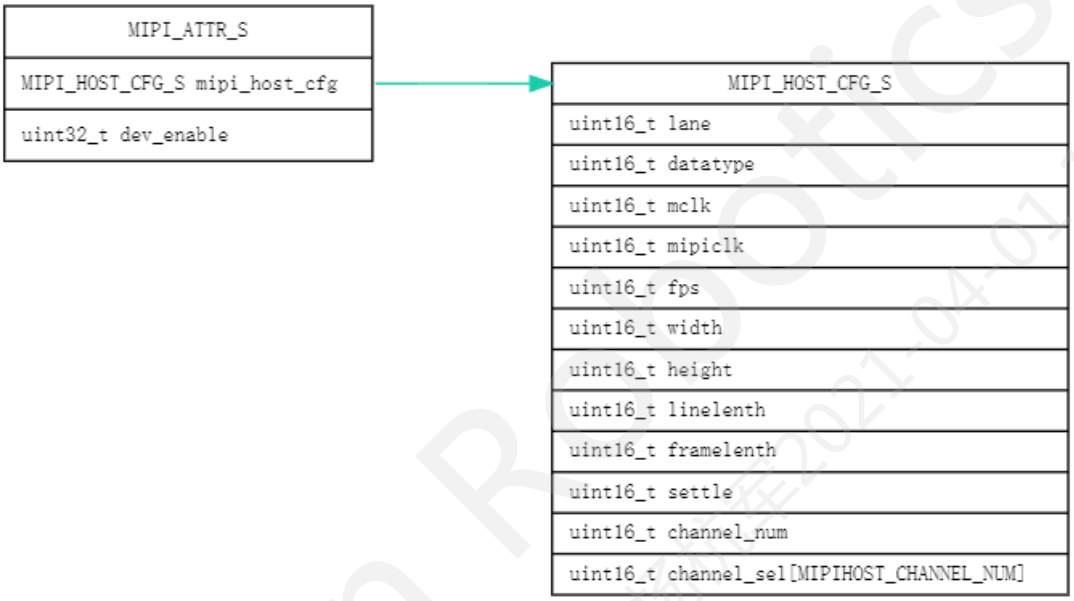


4.2 camera 对应 MIPI 配置

```
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 */
};
```

对应结构体如下：



4.3 参考代码及流程说明

```
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);
```

这段示例代码中涉及到了两个与 VIN 相关的配置，其中 MIPI_SENSOR_INFO_S 是与 sensor 相关配置。

对于 sensor 来说通过 sensor_name，sensor 的框架代码可以找到/lib/sensorlib/下对应的 so 文件。通过库加载的方式，将 sensor 的地址/分辨率/所在 i2c 总线/寄存器宽度等信息在调用库时传入 sensor 的代码中。之后 sensor 可以利用这些信息以及代码中本身有的初始化参数，对 sensor 进行初始化。

sensor 对应的 mipi host 配置，通过 MIPI_ATTR_S 来设置，这里实际配置了 mipi host 的对应接入 sensor 的分辨率/帧率/使用的 lane 数量/数据格式等。mipi host 的配置是基于 sensor 配置的，比如分辨率/数据格式/实际接的物理 lane 数；

4.4 驱动与 HAL 交互

在 VIN 中，camera 相关的控制全部在用户态完成，而 mipi/ISP 的控制则在驱动中完成，通过将用户态相关的配置传递给驱动，由驱动使用，对硬件进行设置，各个相关的设备节点如下：

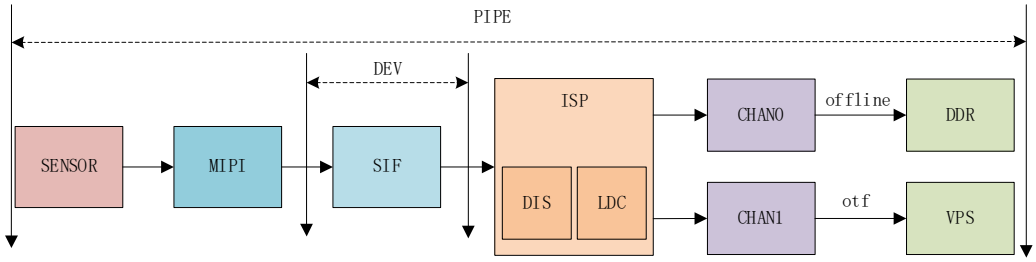


图 4-1 VIN 软件框图

mipi_dev0：该设备节点会在配置中使能，配置 mipi_dev 输出。
mipi_host0~4：mipi host 配置节点，主要完成 mipi_host 的初始化。
mipi_dphy：dphy 相关节点。
SIF 共有两个节点：
sif_capture：设置 sif 相关属性信息，对 sif 模块初始化，可以 dump sif 模块出来的图像。
sif_ddrin：设置 ddrin 节点的属性信息/尺寸/格式等，sif-offline-isp 场景才使用，负责读内存数据给 isp。
ISP 相关节点：
ac_calib：calibration 效果库设置。
ac_isp：isp 效果调节接口使用。
ac_isp4uf0~7：isp 驱动算法库发 command 使用。
ac_sbuf0~7：算法库通过该设备节点与 isp 驱动同步一些算法数据。
video0~7：isp v4l2 设备节点，设置尺寸/格式/大小，内存映射通过该节点与设备交互。

VIN 中，MIPI/SIF 的功能相对简单，对于 MIPI 实际上就是硬件上抽象出来的几个节点，用于用户配置参数，从而设置 MIPI HOST 到对应的状态，能够接受 sensor 的 MIPI 数据输入；

而 SIF 则是将 MIPI HOST 收到的数据再进行一定的处理，比如将不同 sensor 的数据保存到不同的 ddr 地址等；

ISP 的功能相对来说是最复杂的，它需要和 sensor 交互/需要加载对应的算法库/需要加载对应的效果库，在配置代码中：

```
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",
    }
};
```

1 这个标签表示使用 3A 算法，将会使用 lib_algo.so 库的算法；2 这个则是不同 sensor 配置出来的效果库，用于调整 sensor 效果；

5 VIO 调试信息

5.1 SIF 调试信息

查看 SIF 调试信息：**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 调试信息

查看 ISP 调试信息：**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 调试信息

查看当前哪些 pipe 使能：

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

查看各 pipe 配置情况：

cat /sys/devices/platform/soc/a4040000.ipu/info/pipelinex_info （x 取值 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#
```

说明：

subdev0 对应 ipu src，sbudev1~6 对应 ipu us/ds0~ds4。subdev 后面括号里的信息表示这个通道的 buffer 在各个状态的数量。

5.4 PYM 调试信息

查看当前哪些 pipe 使能：

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

查看各 pipe 配置情况：

cat /sys/devices/platform/soc/a4042000.pym/info/pipelinex_info （x 取值 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
```

5.5 IAR 调试信息

查看 IAR 调试信息：**cat /sys/kernel/debug/iar**

```
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 调试信息

6.1 VENC 调试信息

查看编码信息：**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 调试信息

查看解码信息：**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 调试信息

7.1 JENC 调试信息

查看编码信息：**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 调试信息

查看解码信息：**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 常见问题及解决方法

8.1 Camera init 失败

现象：I2C 配置失败，sensor 初始化失败
原因分析：
检测总线下能否检测到设备地址，i2cdect －r －y [bus]
解决办法：
检查下硬件，有可能是排线问题，有可能是硬件连接问题。确保总线下看到设备地址后，才能正确配置 i2c。

8.2 MIPI 错误

现象一：mipi check error
原因分析：
mipi 没有收到数据。
解决办法：
- 检查 sensor 所用的 mipi 索引是不是和 check 的 mipi 是同一个，即 mipi_init 和 mipi_start 传的是不是同一个 mipi 索引。通过 log 也可以看到。
- 确实是没有收到 sensor 数据，检查下 sensor 是否配置成功，平台有要求，sensor_init 序列配置后，不能有数据流输出，如果有，会报错。所以 sensor 初始化完保证数据流是关闭的。
- 检查软件逻辑，有没有打开 sensor 数据流，如果没有打开数据流，也是收不到数据的。

现象二：mipi phy fatal
原因分析：
mipiclk, lane 配置错误
解决办法：
- 修改 settle 值，settle 配置错误，会导致 phy fatal 错误，这个值可以试错，范围 0~127。

现象三：mipi INT_ST_FRAME_FATAL
原因分析：
- 整帧数据中有 CRC 错误，报出 ErrFrameData:
- SOF/EOF 没有配对，比如连续收到两次同一个 VC 的 SOF 没有收到 EOF，报出 ErrFrameSync
- Frame Number 顺序错误，比如前一帧 SOF 的 frame number 是 1， 下一次收到的 frame number 变成了 4。

解决办法：
检查 PHY Clock 以及 settle 的配置， 如参数配置确认无误，确认 Camera/Host 初始化时序

现象四：mipi INT_ST_IPI
原因分析：

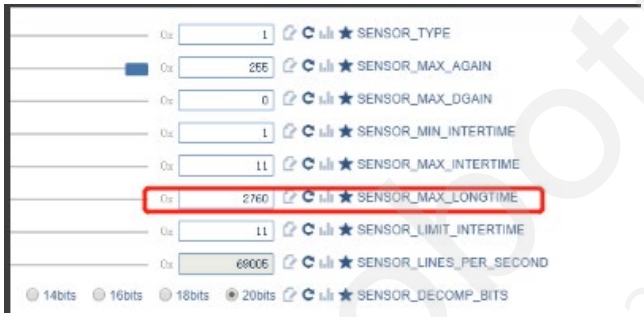
FIFO overflow: FIFO 溢出。IPI Clock 过慢，或者 IPI 取数慢了，比如 HSD 过大或者在收到 Hsync 后过了太久（超过根据 blanking 计算出的 HSD 时间太多），导致 FIFO 已经被塞满后 IPI 还没开始取数或者 IPI 取数速度低于 PPI 填充 FIFO 的速度导致 FIFO 被塞满。

解决办法：

检查 width, height, linelenth, framelenth, fps 的配置是否正确。

8.3 HDR 模式下 AE 异常

现象：sensor_max_longtime == 0

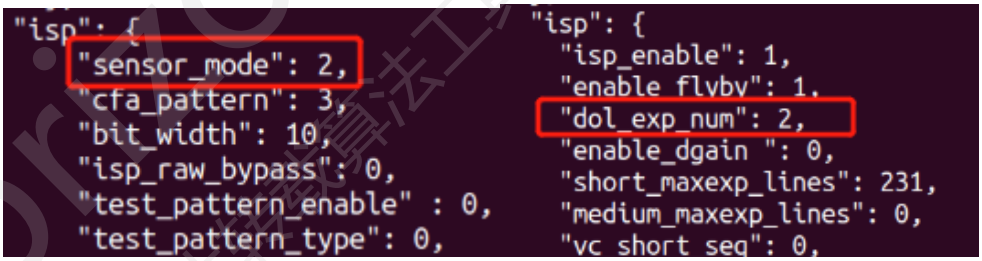


原因分析：

模式配置错误；

解决方法：

修改配置文件为当前模式。



8.4 line/gain 无法跑到 datasheet 最大值

现象：

在较暗的场景下，ae 的 line/gain 无法跑到最大值

原因分析：

sensor 驱动限幅值有误或者 tuning 限幅值有误。

解决方法：

修改 sensor 驱动。

检查 gain 限幅值：

```
turning_data.sensor_data.gain_max = 120 * 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;
```

检查 line 限幅值：

```
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 图像出现横纹/异常色

现象：

yuv 画面异常。

原因分析：

ldc 的 linebuf 不够或者 iram 不够。

解决方法：

检查 line_buf 数目和 iram 地址。

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

8.6 图像分屏异常（hdr 模式下 sif-otf-isp 场景）

现象：

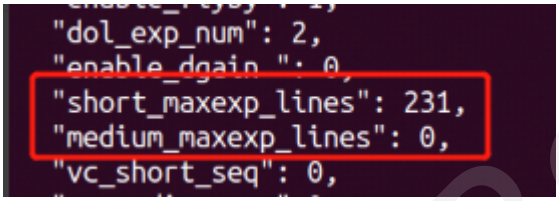
yuv 画面异常。

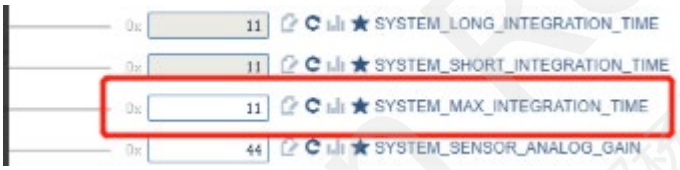
原因分析：

dol2 模式下 sif 的 iram 不够缓存短帧图像。

解决方法：

检查 sif 的 iram 缓存空间，检查 ae 短帧曝光时间。system_max_integration_time 应小于 short_maxexp_lines，否则会可能异常。





8.7 YUV 底下有几行异常

现象：

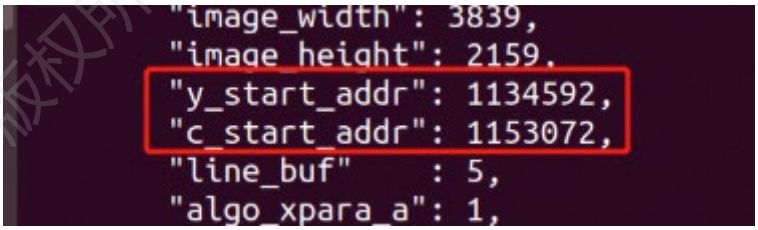
yuv 画面异常。

原因分析：

ldc 的 iram 缓存空间不足。

解决方法：

检查 ldc 的 iram 缓存空间。



9 媒体模块日志查看

9.1 日志级别

控制台输出日志和 logcat 查看日志是二选一的关系，通过环境变量 LOGLEVEL 来控制。
如 export LOGLEVEL=14，即把比 Debug 级别高的日志（<=14）全部输出到 Console。
如果想通过 logcat 查看 Debug 及更高级别的日志，需要 export LOGLEVEL=4。

控制台输出日志		通过 logcat 查看日志	
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 日志标签

媒体模块内部定义了一些 LOG_TAG，所有 TAG 如下：

vio-core
vio-devop
ipu
sif
dwe
gdc
pym
vin
isp
rgn
mipi
vp
vps
venc
vdec
audio
vot
vio-bufmgr
ffmedia
multimedia

注意：
没有标签的日志不能过滤，在满足 LOG LEVEL 级别的情况下会被打印出来（一般见于应用程序或没加 TAG 的模块）。

如果想给应用程序加上 TAG：
1. 可以在文件最开头定义#define LOG_TAG “APP”
2. 包含相关头文件 #include “logging.h”
3. 应用程序中的日志打印用 logging.h 头文件中的 pr_xxx 开关的宏定义

9.3 日志过滤

各模块日志均可通过 logcat 来过滤查看，这里介绍下模块相关如何过滤。logcat 是开源的命令，其他参数可自行探索。

例如只想打印 vps 部分且日志级别高于 Debug 的日志，并输出到文件可以这样操作：
logcat vps:D -f log.txt

想查看多个模块的日志可以在后面追加过滤如查看 vps/vin 模块且级别高于 Debug 的日志：
logcat vps:D vin:D -f log.txt

9.4 日志存储

内核日志会保存在/userdata/log/kernel/目录；
LOGLEVEL 设置为 4，上层日志会保存在/userdata/log/usr/目录；