Hi3519AV100 自适应使用说明

文档版本 00B01

发布日期 2018-10-30

商标声明

 、HISILICON、海思和其他海思商标均为深圳市海思半导体有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受海思公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，海思公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

深圳市海思半导体有限公司

地址：深圳市龙岗区坂田华为基地华为电气生产中心 邮编：518129

网址：http://www.hisilicon.com

客户服务电话：+86-755-28788858

客户服务传真：+86-755-28357515

客户服务邮箱：support@hisilicon.com

前 言

概述

自适应模块可以根据当前环境光线的明暗变化以及用户设置的码率来调节相关图像及编码参数，以优化图像显示效果。

产品版本

与本文档相对应的产品版本如下。

产品名称	产品版本
Hi3519A	V100

读者对象

本文档（本指南）主要适用于以下工程师：

- 技术支持工程师
- 软件开发工程师

修订记录

修订记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

文档版本 00B01 (2018-10-30)

第 1 次临时版本发布。

目 录

插图目录

表格目录

1 概述

摄像机在拍摄视频时会面对外界场景的变化。对于不同的场景，芯片内部的 ISP 模块会相应的使用不同的参数，已达到采集得到的图像效果对于当前场景最优。

1.1 自适应 Sample 总体介绍

1.1.1 模块介绍

之前的自适应 sample 面临着，当适配多款 sensor 时，由于要调节的参数类型和参数值不同，所以每个 sensor 都需要维护一套自适应代码，这样代码量较多，同时一款 sensor 上面添加的参数如果要是配到其他 sensor，也要改其他的自适应代码，维护起来较为困难。

Hi3519AV100 的自适应架构相对于 Hi3559AV100 的改动在于删除 ini2source 的工具，不同的 Sensor 的参数赋值代码统一共用一套。当前主要通过模块的开关来保证不同 Sensor 的不同模式来配置 ISP 相关参数。

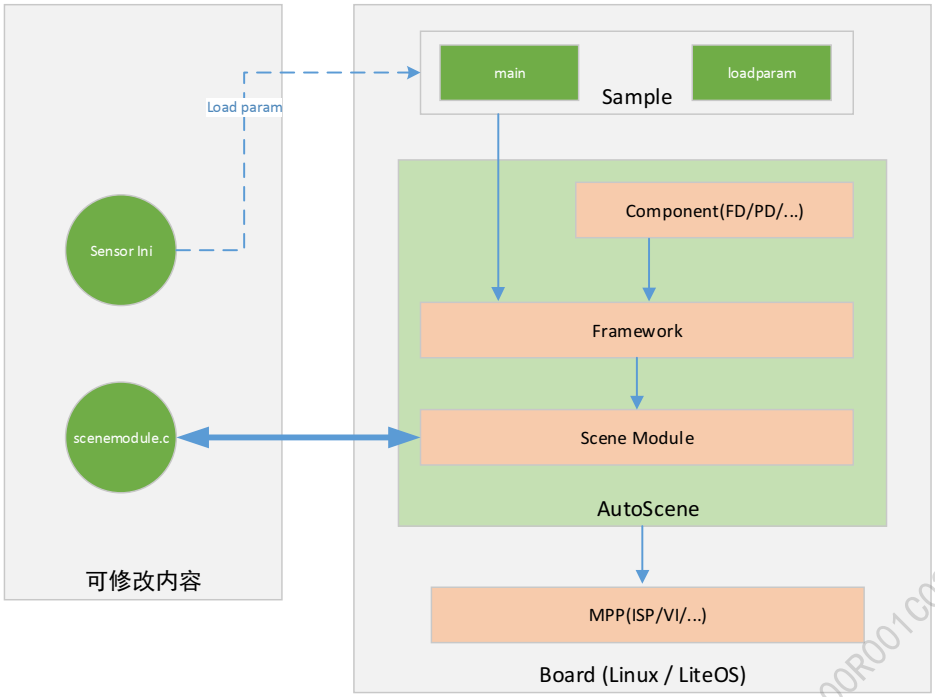
如下图 1-1 所示：其中 0 表示当前模块关闭，该参数不会配置到对应模块的接口中，为 1 时，表示当前模块开启。

图1-1 Ini 文件中模块开关示意图

```
;;;;;;;;;;;;;;module state;;;;;;;;;;;;;;
[module_state]
bStaticAE               = "1"
bDynamicAE              = "1"
bAeWeightTab            = "0"
bStaticAWB              = "0"
bStaticAWBEx            = "0"
bStaticCCM              = "0"
bStaticClut             = "0"
bStaticNr               = "0"
bDynamicNr              = "1"
bDynamicDpc             = "1"
bDynamicBLC             = "1"
bStatic3DNR             = "0"
bDyanamic3DNR           = "1"
bDynamicGamma           = "1"
bStaticWdrExposure      = "0"
bDynamicWdrExposure     = "0"
bDynamicFSWDR           = "0"
bStaticShading          = "0"
bDynamicShading         = "0"
bStaticLdci             = "0"
bDynamicLdci            = "0"
bStaticDehaze           = "0"
bDynamicDehaze          = "0"
bDynamicFalseColor      = "0"
bStaticDRC              = "0"
bDynamicDrc             = "0"
bStaticGlobalCac        = "0"
bStaticLocalCac         = "1"
bStaticSaturation       = "0"
bStaticCa               = "0"
bDynamicDis             = "0"
bStaticSharpen          = "0"
bStaticCSC              = "0"
bStaticCrosstalk        = "0"
```

1.1.2 模块框架图

图1-2 SCENE Sample 框架图



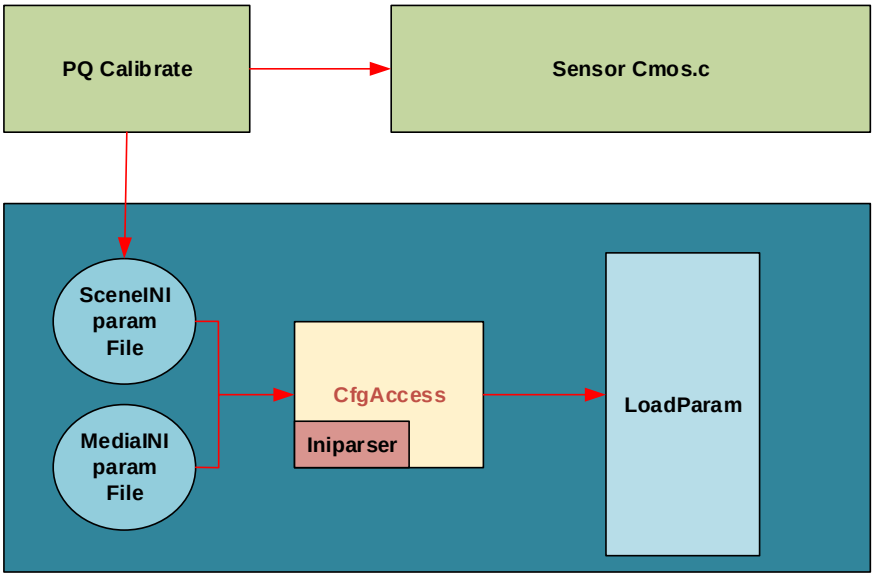
1.2 自适应 Sample 模块介绍

自适应模块按功能可以分解为如下子模块：

- 自适应参数导入
- 自适应主框架
- 自适应调节模块

1.2.1 自适应参数导入

图1-3 自适应 sample 参数获取示意图



自适应调试参数是由 PQ 离线标定获取到的，在自适应模块运行过程当中，应确保自适应调试参数已经全部齐备。

PQ 离线标定参数的存储方式可以分为两种，写在 ini 配置文件或写在源码当中。当参数写在 ini 配置文件上时，可以通过 ini 解析工具得到相应的参数，用此方式，在参数调试过程中，修改参数无需编译源码。而将参数写在源码中时，获取参数无需解析，但是缺点则是，调试时修改参数，需要重新编译代码。

说明

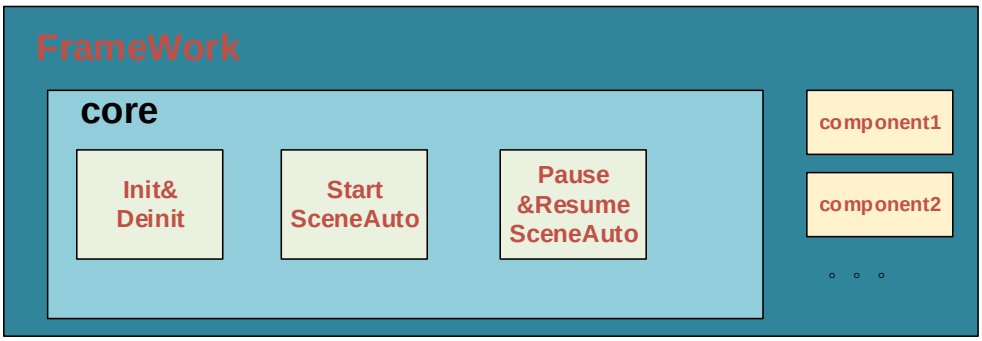
- 自适应调试参数中还有一种类型是由厂测时产线标定获取到的，如 Mesh-Shading 的 table 表以及静态坏点的 table 表。
- 这种调试参数的存储方式只能是写在 ini 配置文件中，因为通过此方式，无需重新编译源码。

1.2.2 自适应主框架

自适应主框架是自适应模块对产品提供功能函数接口的子模块，包括自适应模块的初始化、去初始化，以及根据当前的媒体通路刷新自适应参数并启动自适应参数调节功能等。

该模块可以适用于多种产品形态，包括但不限于 IPC、MobileCam，对于不同的产品形态提供上述提到的基础功能。

图1-4 自适应框架代码示意图



说明

Component 为预留部分，待后续补充自适应新增规格。

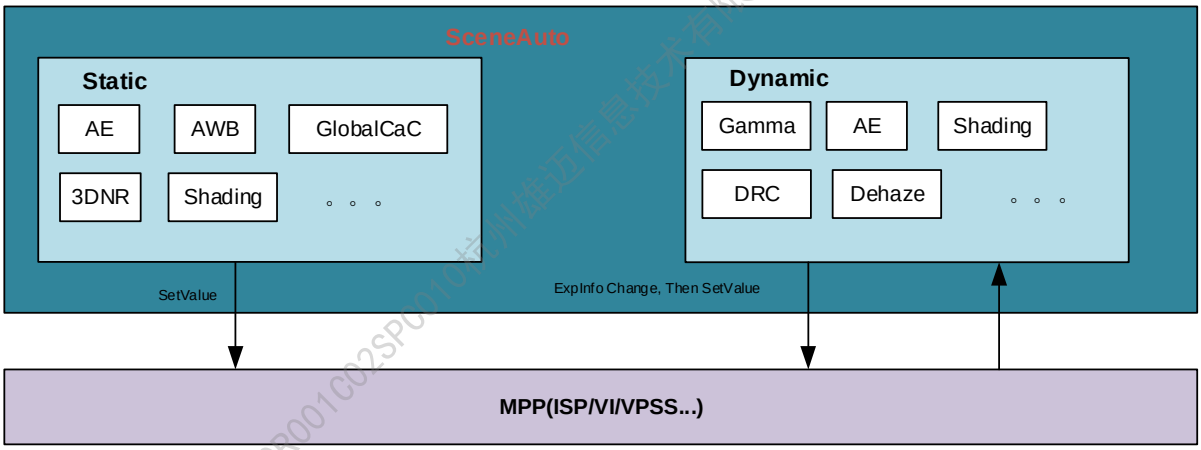
1.2.3 自适应调节模块

自适应参数可以按照静态和动态分成两个大类：

- 静态参数是在参数生效过程中只需要配置一次；
- 动态参数配置则是随着当前场景的变化而变化，每个大类又根据 ISP 模块分成各个子类。

对于不同的产品形态，或者对于相同产品形态不同的 sensor，需要调试的自适应参数是有可能不同的。因此将自适应调试参数赋值单独剥离成一个模块，这样对于自适应整个大模块来说，面对参数调试多样性的需求，只有该子模块是变化的，降低代码维护复杂度。

图1-5 自适应调节模块示意图

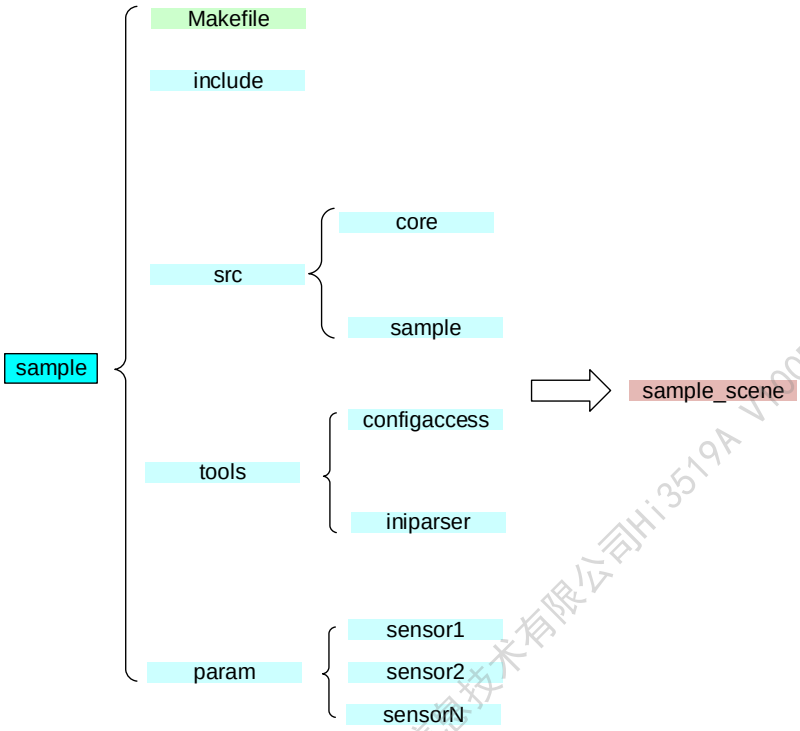


1.3 文件组织

自适应模块的文件组织结构如图 1-6 所示，主要分为 ini 解析组件 configaccess 和 iniparser 工具，各 sensor 不同的 param、source 和 lib，以及整个模块的公共代码。

- Tools 内部基于 iniparser 封装的 cfgaccess 工具，用作 ini 文件的解析；
- Src 内部主要有 core：自适应框架代码；sample: 自适应 sample 的支撑代码；Include 内部主要是自适应 sample 和自适应框架代码所需头文件。
- param 目录主要是放置 N 个 Sensor 参数 ini 文件。

图1-6 SCENE 模块文件组织

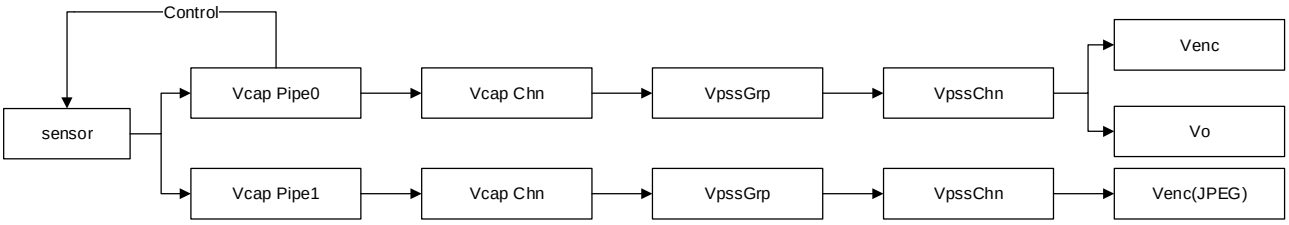


1.4 重要概念

下面以双 pipe 抓拍（当处于 8M 抓拍，前端 sensor 时序为 4K30）为例，介绍本模块如何调节图像参数，并讲解一些 API 数据类型中的概念，方便用户配置。

单 sensor 预览抓拍通路，如图 1-7 所示。

图1-7 单 sensor 预览抓拍通路示意图



如图 1-7 所示当前媒体通路中两个 Pipe 的数据来源于同一个 sensor，且 Pipe0 中的 ISP 可以控制 sensor，因此 Pipe0 和 pipe1 的 MainPipeHdl 都为 0。

- **MainPipeHdl** 指可以控制 sensor 的 ISP 所在的 pipe 数量和 sensor 的数量一致。当两个 Pipe 的数据来源于同一个 sensor，它们的 MainPipeHdl 相同。
- 单 sensor 预览抓拍通路中，需要调节图像质量，则需要配置位于 Pipe 通路上的 ISP、位于 vpss 上的 3DNR、位于 VENC 上的编码属性。
- **Pipe 通路**指以 Vcap Pipe 为起点，包括和其绑定的 Vpss 以及后续的 Venc、Vo。所以图 1-7 中有两条 pipe 通路。

  场景自适应模块将当前的媒体通路分解为一条或多条 pipe 通路，每条 pipe 通路配置一组自适应参数，每一组自适应参数由一个图像参数调节文件维护，请参考 4 数据类型。当前 pipe 通路需要配置的自适应参数由 u8PipeParamIndex 来指定。u8PipeParamIndex 具体含义请参考 HI_SCENE_PIPE_ATTR_S。

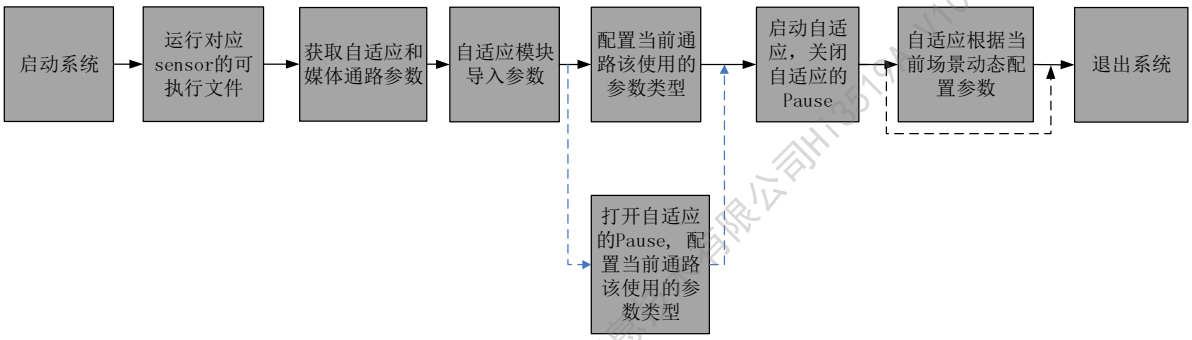
2 开发指引

2.1 使用流程

2.1.1 使用概述

自适应模块相关 API 和参数说明可参考本文档下述部分。具体实现可参考自适应的 SDK sample，自适应使用流程如图 2-1 所示。

图2-1 SCENE 模块使用流程图



2.1.2 运行对应 Sensorini 文件

场景说明

在 SDK sample 中，不同的 sensor 分属于不同的文件夹且各自独立，每一个 sensor 维护自己的参数。

工作流程

运行过程如下：

步骤 1. 先在 Sceneauto 的目录下，执行 Makefile 生成对应 sample_scene。

步骤 2. 利用 sample_scene 加载 param 目录下对应的 sensor 的 ini 文件。

----结束

注意事项

- Sceneauto 文件下的 param 文件中的参数配置说明参考本开发参考的后续章节(2.2)。
- Src/core 文件下的 source 文件中的源码用于自适应参数的赋值。

2.1.3 获取和导入自适应和媒体通路参数

场景说明

启动场景自适应功能所需要的参数分为两部分，自适应调试参数和媒体通路参数。在 Sensor 的文件中，自适应调试参数的 ini 文件一般有多个，如启动自适应，媒体通路上每一个正在运行中的 pipe 都应当赋值一组自适应调试参数，各 pipe 上赋值的参数可以相同也可以不同。

在 sample 中，自适应调试参数和媒体通路参数都被存储在 ini 配置文件中，首先需要在运行过程中解析 ini 文件将参数写入到特定内存中的结构体，再导入到自适应模块当中。

工作流程

获取和导入模块所需参数的过程如下：

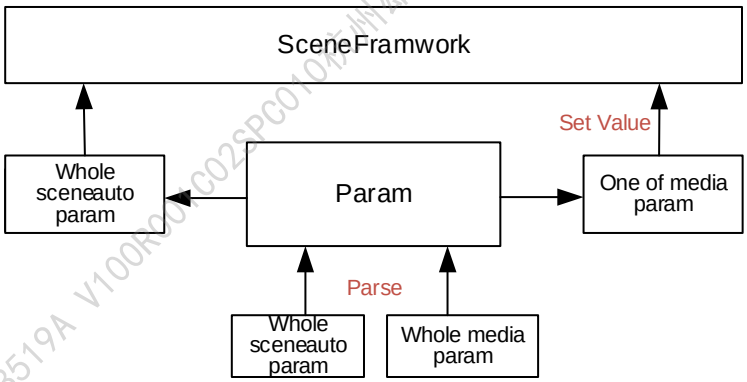
步骤 1. Configaccess 工具解析对应 sensor 路径下的 config_cfgaccess_hd.ini。

步骤 2. Configaccess 工具分别解析自适应调试参数和媒体通路参数，并将这两类参数分别存储在一个数组当中。

步骤 3. Sample 调用自适应接口将自适应调试参数数组导入自适应模块中，并选取一种和当前媒体通路匹配的参数配置到模块中。

----结束

图2-2 SCENE 模块使用流程图



注意事项

- 媒体通路参数在后续的数据结构详解中有详细说明，媒体通路参数主要是当前媒体通路的使能关系，指示各个使能的媒体 pipe 上该配置哪一套自适应调试参数。
- 自适应调试参数的集合是一个数组，媒体通路参数中各 pipe 上要配置的自适应调试参数为该数组的下标，需提前指定。
- 各自适应调试参数在数组中的位置在 config_cfgaccess_hd.ini 中指定。

2.1.4 启动场景自适应

场景说明

场景自适应的启动主要分成两个步骤。配置各 pipe 上的静态参数，静态参数表示该参数只需要配置一次。之后再配置各 pipe 上的动态参数，动态参数表示该参数会随着当前的场景变化而变化。

说明

当前版本动态参数跟随的场景的变化而变化，主要取决于当前场景下的曝光量、ISO 或者曝光比。

工作流程

启动场景自适应的过程如下：

步骤 1. 调用自适应相应的接口，配置媒体通路参数，调用 HI_SCENE_SetSceneMode()。

步骤 2. 自适应启动，并跟随当前场景动态调试某些参数。

----结束

2.1.5 切换自适应配置

场景说明

在场景自适应的过程中，会伴随着媒体通路变化这样的典型行为。当媒体通路发生变化时，如线性模式切换到 WDR 模式，运动 DV 多个分辨率之间的切换等，各个通路 pipe 上的要配置的自适应参数一般也会需要相应的发生变化，此时需要切换自适应的配置。

工作流程

切换自适应配置的过程如下：

步骤 1. 首先将自适应 Pause，调用 HI_SCENE_Pause(HI_TRUE)。

步骤 2. 再切换媒体通路。

步骤 3. 获取当前媒体通路下自适应模块针对此通路的配置参数。

步骤 4. 启动自适应，调用 HI_SCENE_SetSceneMode()。

步骤 5. 将自适应 Resume，调用 HI_SCENE_Pause(HI_FALSE)。

----结束

注意事项

- 当媒体通路关闭时，需要调用自适应接口 HI_SCENE_Pause(HI_TRUE)，来防止调用 ISP 接口。
- 当媒体通路开启时，需要调用自适应接口 HI_SCENE_Pause(HI_FALSE)，来使能调用 ISP 接口。

2.1.6 Ini 解析工具介绍

当前自适应 sample 中，解析 ini 参数主要依赖于开源的 ini 文件解析 c 库 iniparser，以及在此基础上海思封装的 cfgaccess 工具。

Iniparser 主要功能就是将 ini 配置文件中的字符串解析成 bool 值、int 值、long 值等特定的数据类型，而 cfgaccess 工具是在此基础上做了一层封装，可以通过一个主 ini 配置文件对其余 ini 配置文件做到很好的管理，在自适应 sample 中，该主 ini 配置文件为各 sensor 下 param 文件中 config_cfgaccess_hd.ini，如图 2-3 所示。

图2-3 Cfgaccess 工具所需要的 ini 配置管理文件

```
[module]
module_num          = "3"
module1             = "video_mode_0"
module2             = "scene_mode_0"
module3             = "scene_mode_1"

[video_mode_0]
path                = "./imx117/param/"
cfg_filename        = "config_videomode_0.ini"

[scene_mode_0]
path                = "./imx117/param/"
cfg_filename        = "config_product_scene_1080p30_linear.ini"

[scene_mode_1]
path                = "./imx117/param/"
cfg_filename        = "config_product_scene_1080p30_3t1_wdr.ini"
```

video_mode_0 所对应的媒体参数配 ini 文件只有一个，多种媒体参数配置放在该文件中。而 scene_mode_n 所对应的自适应配置参数有多个，每个 ini 配置文件对应一组参数。

通过该文件，ini 解析工具可以一次解析多个 ini 配置文件，并且将 ini 配置文件生成的参数有序的存储。

同时通过 cfgaccess 工具，可以方便快捷的知道 ini 配置文件中是否有对应的项，具体的使用请参考 sample 代码。

2.1.7 自适应配置参数介绍

自适应配置参数包括动态和静态两大类，其中各大类还可以按模块细分成子类。典型的如静态 AE、动态 3DNR 以及动态 AE 等，动态 AE 每种类型的 ini 配置如图 2-4 所示。

图2-4 动态 AE 的 ini 配置文件

```
;;;;;;;;;;;;;;;;;dynamic_AE_parameter;;;;;;;;;;;;;;;;;
[dynamic_ae]
aeExpCount                = "5"
aeExpLtoHThresh           = "10000,  40000,  300000,  1000000,  10000000"
aeExpHtoLThresh           = "5000,   7000,   30000,   200000,   800000"
AutoCompesation           = "30,     28,     24,      24,       24"
AutoHistOffset            = "20,     20,     20,      20,       20"
```

其中 aeExpCount 代表数组的大小，而后面的 ExpLtoH 以及 ExpHtoL 则表示曝光量的数组，Compesation 以及 HistOffset 分别和 ISP 中配置曝光量中的某项参数对应。

通过 dynamic_ae 这样的模块命名，可以知道这个模块是配置与 AE 相关的动态参数，同时通过 ExpThresh 可以知道动态配置 AE 参数是由当前环境的曝光量变化决定的，同时动态 AE 调整的值为 Compesation 和 HistOffset 对应的 ISP 参数。

说明

- 一个自适应配置 ini 文件可以通过参数解析填充到自适应参数数组中，每个 pipe 中可以配置自适应参数数组中的一项。
- Sample 里一个 sensor 的自适应配置 ini 文件的数量有上限，注意不要超过上限导致数组越界

2.1.8 媒体通路配置参数介绍

媒体通路配置参数，需要作为一个入参传入到自适应框架代码的接口中，启动自适应。图 2-5 是一组媒体通路配置参数的典型赋值。

从通路配置参数中，可以知道当前的模式是线性还是 WDR，所有 pipe 是开还是关，以及该 pipe 上应该配置哪一组自适应参数，通过 PipeParamIndex 来指定。

说明

- ini 配置中的 PipeParamIndex 是指自适应配置参数数组的下标。
- 自适应配置参数数组的排列是在如图 2-3 中的 hd 文件中指定，scene_mode_0 对应数组下标 0，scene_mode_1 对应数组下标 1，依次排列。

多组媒体通路配置存在同一个 ini 配置文件中，在调用自适应函数接口时，只需要使用其中一种配置，在自适应过程中，可以动态切换配置。

图2-5 媒体通路参数的 ini 配置文件

```
[pipe_comm_0]
SCENE_MODE              = "0"               ;;0- linear, 1-wdr, 2-hdr

[pipe_0_0]
Enable                  = "1"               ;;0 - disable, 1 - able
VcapPipeHdl             = "0"               ;;Pipe Handle, must equals to pipe_X;
MainPipeHdl             = "0"               ;;MainIsp Handle
PipeChnHdl              = "0"               ;;Pipe Chn
VpssHdl                 = "0"               ;;VpssGrpHdl
VportHdl                = "0"               ;;VpssChnHdl
VencHdl                 = "0"               ;; Not Used Currently
PipeParamIndex          = "0"               ;; Pipe Use Which Param
PipeType                = "1"               ;;0 - snap, 1-video

[pipe_0_1]
Enable                  = "0"               ;;0 - disable, 1 - able
VcapPipeHdl             = "1"               ;;Pipe Handle, must equals to pipe_X;
MainPipeHdl             = "0"               ;;MainIsp Handle
PipeChnHdl              = "0"               ;;Pipe Chn
VpssHdl                 = "0"               ;;VpssGrpHdl
VportHdl                = "0"               ;;VpssChnHdl
VencHdl                 = "0"               ;; Not Used Currently
PipeParamIndex          = "0"               ;; Pipe Use Which Param
PipeType                = "1"               ;;0 - snap, 1-video

[pipe_0_2]
Enable                  = "0"               ;;0 - disable, 1 - able
VcapPipeHdl             = "2"               ;;Pipe Handle, must equals to pipe_X;
MainPipeHdl             = "0"               ;;MainIsp Handle
PipeChnHdl              = "0"               ;;Pipe Chn
VpssHdl                 = "0"               ;;VpssGrpHdl
VportHdl                = "0"               ;;VpssChnHdl
VencHdl                 = "0"               ;; Not Used Currently
PipeParamIndex          = "0"               ;; Pipe Use Which Param
PipeType                = "1"               ;;0 - snap, 1-video
```

2.2 增加 sensor，如何跑 Sample

场景说明

自适应模块可以兼容多种 sensor，sample 提供了几种典型 sensor。当用户需要添加一款 sensor 的时候，需要做如下的适配。

工作流程

适配 Sensor 的过程如下：

步骤 1. 在 sample 主目录的 param 下新建一个文件夹，命名为 sensorXXXX。

步骤 2. 在 sensorXXXX 放置该 sensor 中要调节的参数。

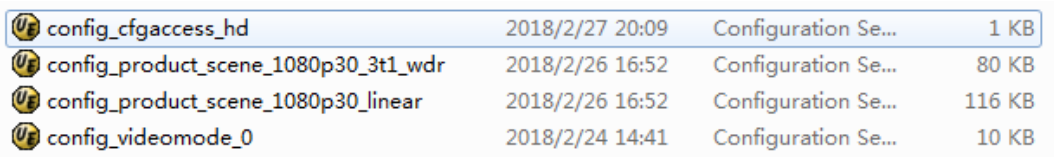
可以将其他 sensor 的参数文件夹下的文件拷贝至该文件下，增删其中的参数类型。

步骤 3. 修改参数文件夹中的 config_cfgaccess_hd.ini 中各个 path 的路径名。

步骤 4. 修改参数文件下各 ini 的参数。

首先增加该 sensor 自适应配置参数，如图 2-8 所示中的 config_product_scene_1080p30_3t1_wdr，建议按照分辨率帧率、线性 WDR 等来区分命名，其中的参数增删可参考 2.3 章节。

图2-6 Sensor 下 param 文件夹中的内容



再根据 sensor 会使用的媒体通路的配置来修改 config_videomode_0 里的内容。

最后修改 hd.ini 中的内容。

步骤 5. 在主目录下 make clean; make，生成自适应 sample 的可执行文件。

----结束

2.3 增删参数，如何跑 Sample

场景说明

在 PQ 调试过程当中，需要经常修改参数和增删参数，最后达到一个图像效果的最优。本小结，主要描述当增删参数时，自适应模块应该如何去做一个适配。

下面的工作流程是讲解 ini 配置项中需要增加参数，该如何去做。

工作流程

增加参数的过程如下：

步骤 1. 首先明确对应 sensor 该增加的参数是什么，对应的 isp 的接口数据类型是哪一个。

步骤 2. 在该 sensor 的 param 文件下的自适应配置参数 ini 文件中加上这个参数项，给出自己的命名。如 static_ae 中增加一个系统增益的最小值，增加前和增加后的对比如图 2-7 所示。

图2-7 自适应配置项中增加参数示意图

```
;;;;;;;;;;;;;;;static_AE_parameter;;;;;;;;;;;;;;;
[static_ae]
AERouteExValid          = "1"
AERunInterval           = "1"
AutoSpeed               = "64"
AutoTolerance           = "2"
AutoBlackDelayFrame     = "8"
AutoWhiteDelayFrame     = "0"
AutoSysGainMax          = "65536"
```

```
;;;;;;;;;;;;;;;static_AE_parameter;;;;;;;;;;;;;;;
[static_ae]
AERouteExValid          = "1"
AERunInterval           = "1"
AutoSpeed               = "64"
AutoTolerance           = "2"
AutoBlackDelayFrame     = "8"
AutoWhiteDelayFrame     = "0"
AutoSysGainMax          = "65536"
AutoSysGainMin          =
```

在 ini 配置文件中的 AutoSysGainMin 即对应于 SDK 接口中 ISP_EXPOSURE_ATTR_S 里的 stAuto.stSysGainRange.u32Min，使用者需要清晰了解这种映射关系。

同时该 sensor 下所有的自适应配置 ini 文件中都需要加个这个参数项。

步骤 3. 在 sample 主目录下 include 文件下的 hi_scene_setparam.h 中的数据结构体 HI_SCENE_STATIC_AE_S 中，增加一个参数，就是 ini 配置文件中 AutoSysGainMin 所对应的参数。该参数命名由用户指定如图 2-8 所示。

图2-8 自适应赋参头文件中增加参数示意图

```
typedef struct hiSCENE_STATIC_AE_S
{
    HI_BOOL bAERouteExValid;
    HI_U8 u8AERunInterval;
    HI_U32 u32AutoSysGainMax;
    HI_U8 u8AutoSpeed;
    HI_U8 u8AutoTolerance;
    HI_U16 u16AutoBlackDelayFrame;
    HI_U16 u16AutoWhiteDelayFrame;
} HI_SCENE_STATIC_AE_S;
```

```
typedef struct hiSCENE_STATIC_AE_S
{
    HI_BOOL bAERouteExValid;
    HI_U8 u8AERunInterval;
    HI_U32 u32AutoSysGainMax;
    HI_U32 u32AutoSysGainMin;
    HI_U8 u8AutoSpeed;
    HI_U8 u8AutoTolerance;
    HI_U16 u16AutoBlackDelayFrame;
    HI_U16 u16AutoWhiteDelayFrame;
} HI_SCENE_STATIC_AE_S;
```

步骤 4. 在该 src/core 文件下的 hi_scene_setparam.c 中，增加该参数赋参语句如下图 2-9 所示：

图2-9 自适应赋参源文件中增加参数示意图

```
HI_S32 HI_SCENE_SetStaticAE(VI_PIPE ViPipe, HI_U8 u8Index)
{
    if(HI_TRUE != g_astScenePipeParam[u8Index].stModuleState.bStaticAE)
    {
        return HI_SUCCESS;
    }

    HI_S32 i = 0;
    HI_S32 s32Ret = HI_SUCCESS;

    ISP_PUB_ATTR_S stPubAttr;
    ISP_EXPOSURE_ATTR_S stExposureAttr;
    ISP_AE_ROUTE_EX_S stAeRouteEx;

    s32Ret = HI_MPI_ISP_GetAERouteAttrEx(ViPipe, &stAeRouteEx);
    CHECK_SCENE_RET(s32Ret);

    stAeRouteEx.u32TotalNum = g_astScenePipeParam[u8Index].stStaticAeRouteEx.u32TotalNum;
    for (i = 0; i < stAeRouteEx.u32TotalNum; i++)
    {
        stAeRouteEx.astRouteExNode[i].u32IntTime = g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32IntTime[i];
        stAeRouteEx.astRouteExNode[i].u32Again= g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32Again[i];
        stAeRouteEx.astRouteExNode[i].u32Dgain= g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32Dgain[i];
        stAeRouteEx.astRouteExNode[i].u32IspDgain= g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32IspDgain[i];
    }

    s32Ret = HI_MPI_ISP_SetAERouteAttrEx(ViPipe, &stAeRouteEx);
    CHECK_SCENE_RET(s32Ret);

    s32Ret = HI_MPI_ISP_GetExposureAttr(ViPipe, &stExposureAttr);
    CHECK_SCENE_RET(s32Ret);

    s32Ret = HI_MPI_ISP_GetPubAttr(ViPipe, &stPubAttr);
    CHECK_SCENE_RET(s32Ret);

    i = g_astScenePipeParam[u8Index].stStaticAeRouteEx.u32TotalNum - 1;
    stExposureAttr.stAuto.stExpTimeRange.u32Max = g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32IntTime[i];

    if (WDR_MODE_NONE != stPubAttr.enWDRMode)
    {
        stExposureAttr.stAuto.stDGainRange.u32Max = g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32Dgain[i];
        stExposureAttr.stAuto.stISPDGainRange.u32Max = g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32IspDgain[i];
    }

    stExposureAttr.bAERouteExValid = g_astScenePipeParam[u8Index].stStaticAe.bAERouteExValid;
    stExposureAttr.u8AERunInterval = g_astScenePipeParam[u8Index].stStaticAe.u8AERunInterval;
    stExposureAttr.stAuto.stSysGainRange.u32Max = g_astScenePipeParam[u8Index].stStaticAe.u32AutoSysGainMax;

HI_S32 HI_SCENE_SetStaticAE(VI_PIPE ViPipe, HI_U8 u8Index)
{
    if(HI_TRUE != g_astScenePipeParam[u8Index].stModuleState.bStaticAE)
    {
        return HI_SUCCESS;
    }

    HI_S32 i = 0;
    HI_S32 s32Ret = HI_SUCCESS;

    ISP_PUB_ATTR_S stPubAttr;
    ISP_EXPOSURE_ATTR_S stExposureAttr;
    ISP_AE_ROUTE_EX_S stAeRouteEx;

    s32Ret = HI_MPI_ISP_GetAERouteAttrEx(ViPipe, &stAeRouteEx);
    CHECK_SCENE_RET(s32Ret);

    stAeRouteEx.u32TotalNum = g_astScenePipeParam[u8Index].stStaticAeRouteEx.u32TotalNum;
    for (i = 0; i < stAeRouteEx.u32TotalNum; i++)
    {
        stAeRouteEx.astRouteExNode[i].u32IntTime = g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32IntTime[i];
        stAeRouteEx.astRouteExNode[i].u32Again= g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32Again[i];
        stAeRouteEx.astRouteExNode[i].u32Dgain= g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32Dgain[i];
        stAeRouteEx.astRouteExNode[i].u32IspDgain= g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32IspDgain[i];
    }

    s32Ret = HI_MPI_ISP_SetAERouteAttrEx(ViPipe, &stAeRouteEx);
    CHECK_SCENE_RET(s32Ret);

    s32Ret = HI_MPI_ISP_GetExposureAttr(ViPipe, &stExposureAttr);
    CHECK_SCENE_RET(s32Ret);

    s32Ret = HI_MPI_ISP_GetPubAttr(ViPipe, &stPubAttr);
    CHECK_SCENE_RET(s32Ret);

    i = g_astScenePipeParam[u8Index].stStaticAeRouteEx.u32TotalNum - 1;
    stExposureAttr.stAuto.stExpTimeRange.u32Max = g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32IntTime[i];

    if (WDR_MODE_NONE != stPubAttr.enWDRMode)
    {
        stExposureAttr.stAuto.stDGainRange.u32Max = g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32Dgain[i];
        stExposureAttr.stAuto.stISPDGainRange.u32Max = g_astScenePipeParam[u8Index].stStaticAeRouteEx.au32IspDgain[i];
    }

    stExposureAttr.bAERouteExValid = g_astScenePipeParam[u8Index].stStaticAe.bAERouteExValid;
    stExposureAttr.u8AERunInterval = g_astScenePipeParam[u8Index].stStaticAe.u8AERunInterval;
    stExposureAttr.stAuto.stSysGainRange.u32Max = g_astScenePipeParam[u8Index].stStaticAe.u32AutoSysGainMax;
    stExposureAttr.stAuto.stSysGainRange.u32Min = g_astScenePipeParam[u8Index].stStaticAe.u32AutoSysGainMin;
```

步骤 5. 在自适应主目录下的 scene_loadparam.c 中的 SCENE_LoadStaticAE 函数体内，增加如图 2-10 所示语句。

图2-10 自适应获取参数源文件中增加参数示意图

```
snprintf(aszIniNodeName, SCENE_INIPARAM_NODE_NAME_LEN, "static_ae:AutoSysGainMin");/*AutoSysGainMin*/
s32Ret = HI_CONFACCESS_GetString(SCENE_INIPARAM, pszIniModule, aszIniNodeName, NULL, &pszString);
SCENE_INIPARAM_CHECK_LOAD_RESULT(s32Ret, aszIniNodeName);
if (HI_NULL != pszString)
{
    free(pszString);
    pszString = HI_NULL;
    s32Ret = HI_CONFACCESS_GetInt(SCENE_INIPARAM, pszIniModule, "static_ae:AutoSysGainMin", 0, &s32Value);
    if (HI_SUCCESS != s32Ret)
    {
        MLOGE("load static_ae:AutoSysGainMin failed\n");
        return HI_FAILURE;
    }
    pstStaticAe->u32AutoSysGainMax = (HI_U32)s32Value;
}
```

步骤 6. 当需要减少调试参数，则只需要做一下增加 sensor 的反方向操作，但是 scene_loadparam.c 中的代码无需修改，因为这是一份所有 sensor 共享的公共代码，可以做增量，不影响其他 sensor 的代码，但是如果做减量，就会影响其余 sensor 的代码实现。

----结束

2.4 典型通路配置方案

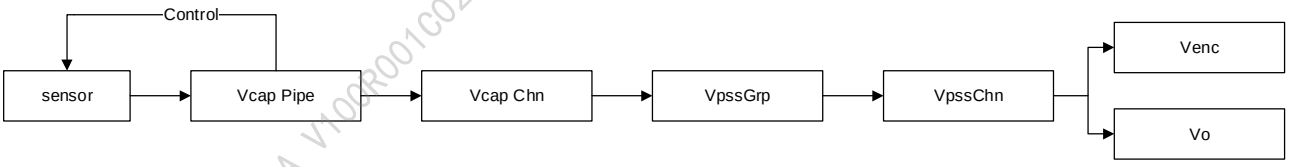
图像质量自适应调节的时候需要调用 HI_SCENE_SetSceneMode。不同的媒体通路类型，该接口的入参不同。没有特别说明，则 ISP 模式为线性。下面以媒体业务的典型的四种媒体通路来依次介绍上述接口入参 HI_SCENE_MODE_S 如何配置。

2.4.1 单 sensor 预览录像

现假定该 pipe 为 pipe0，且此时工作模式为 4K30 录像。媒体通路中使能了一个 pipe。同时该 pipe 中的 ISP 也是控制 sensor 的 ISP。因此 MainPipeHdl 数量为 1，且和使能的 VcapPipeHdl 句柄一致。

单 sensor 预览录像通路，如图 2-11 所示。

图2-11 单 sensor 预览录像通路示意图



当媒体通路如图 2-11 所示，HI_SCENE_MODE_S 的参数应配置，如表 2-1 所示。

表2-1 通路一 HI_SCENE_MODE_S 参数配置表

参数类型	astPipeAttr[0]	astPipeAttr[1]
bEnable	1	0
VcapPipeHdl	0	default
MainPipeHdl	0	default
u8PipeParamIndex	0	default
enPipeType	VIDEO	default

其余值按通路实际状态配置。

2.4.2 单 sensor 预览抓拍

现假定预览通路 pipe 为 pipe0，抓拍通路 pipe 为 pipe1。且此时工作模式为 8M 抓拍。媒体通路中使能了两个 pipe。同时两个 pipe 的数据都来源于一个 sensor，sensor 数量为 1，因此 MainPipeHdl 的个数为 1。且控制 sensor 的 isp 所在的 pipe 在预览通路。

当媒体通路如图 2-12 所示，HI_SCENE_MODE_S 的参数应配置，如表 2-2 所示。

图2-12 单 sensor 预览抓拍通路示意图

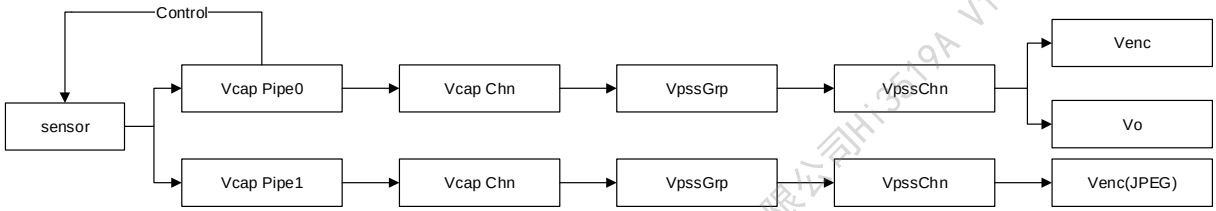


表2-2 通路二 HI_SCENE_MODE_S 参数配置表

参数类型	astPipeAttr[0]	astPipeAttr[1]
bEnable	1	1
VcapPipeHdl	0	1
MainPipeHdl	0	0
u8PipeParamIndex	1	2
enPipeType	VIDEO	SNAP

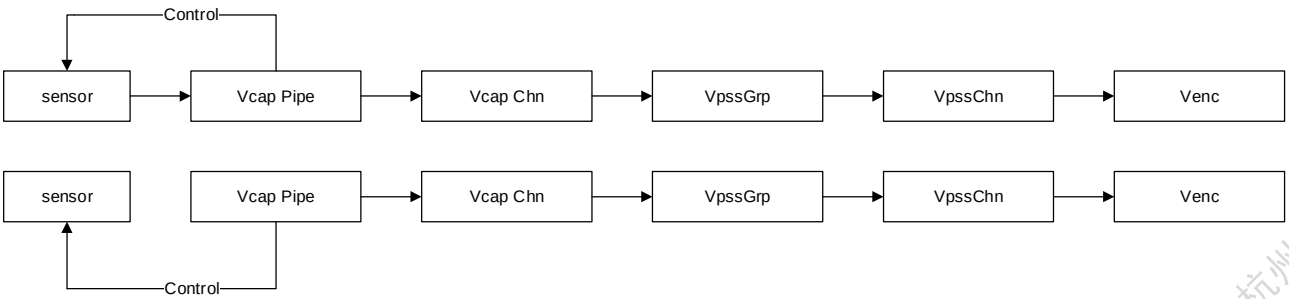
其余值按通路实际状态配置。

2.4.3 多 sensor 预览录像

现假定预览通路 pipe 为 pipe0 和 pipe1。且此时工作模式为 4K30 前后路预览录像。媒体通路中使能了两个 pipe。同时两个 pipe 的数据都来源于两个 sensor，sensor 数量为 2，因此 MainPipeHdl 的个数为 2。且控制 sensor 的 isp 所在的 pipe 在预览通路。

多 sensor 预览录像通路，如图 2-13 所示。

图2-13 多 sensor 预览录像通路示意图



当媒体通路如图 2-13 所示，HI_SCENE_MODE_S 的参数应配置，如表 2-3 所示。

表2-3 通路三 HI_SCENE_MODE_S 参数配置表

参数类型	astPipeAttr[0]	astPipeAttr[1]
bEnable	1	1
VcapPipeHdl	0	1
MainPipeHdl	0	1
u8PipeParamIndex	0	0
enPipeType	VIDEO	VIDEO

其余值按通路实际状态配置。

2.4.4 单 sensor WDR 预览录像

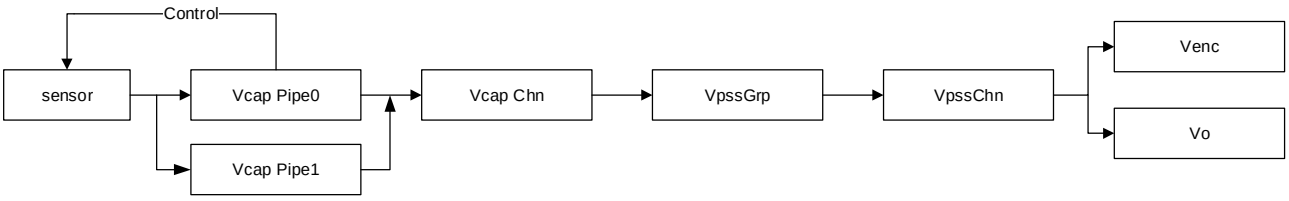
WDR 2to1 模式需要使能两个 pipe，但是只使能一个 ISP。因此对应场景自适应模式来说，当前的 pipe 通路只有一个。

现假定该 pipe 为 pipe0，且此时工作模式为 4K30 录像。

媒体通路中使能了两个 pipe。同时两个 pipe 的数据都来源于一个 sensor，因此 MainPipeHdl 的个数为 1。且控制 sensor 的 isp 所在的 pipe 为 Vcap Pipe0。

单 sensor WDR2to1 预览录像通路，如图 2-14 所示。

图2-14 单 sensor WDR2to1 预览录像通路示意图



当媒体通路如图 2-14 所示，HI_SCENE_MODE_S 的参数应配置，如表 2-4 所示。

表2-4 通路四 HI_SCENE_MODE_S 参数配置表

参数类型	astPipeAttr[0]	astPipeAttr[1]
bEnable	1	0
VcapPipeHdl	0	default
MainPipeHdl	0	default
u8PipeParamIndex	1	default
enPipeType	VIDEO	default

3 API 参考

SCENE 提供以下 API：

- HI_SCENE_Init：初始化 SCENE 模块。
- HI_SCENE_SetSceneMode：配置自适应媒体参数，开启自适应。
- HI_SCENE_Pause：暂停或恢复自适应。
- HI_SCENE_GetSceneMode：获取当前的自适应媒体参数。
- HI_SCENE_Deinit：去初始化 SCENE 模块。

HI_SCENE_Init

【描述】

初始化 SCENE 模块，导入自适应配置参数。

【语法】

```
HI_S32 HI_SCENE_Init(const HI_SCENE_PARAM_S* pstSceneParam)
```

【参数】

参数名称	描述	输入/输出
pstSceneParam	自适应配置参数总集。	输入

【返回值】

返回值	描述
0	成功。
非 0	参见错误码。

【需求】

- 头文件：hi_scene.h, hi_scene_autogenerate.h

- 库文件：Null

【注意】

- pstSceneParam 不能为空。
- 最大支持 10 种自适应参数配置。

【举例】

参考 scene_sample.c。

HI_SCENE_SetSceneMode

【描述】

配置自适应媒体参数，开启自适应。

【语法】

```
HI_S32 HI_SCENE_SetSceneMode(const HI_SCENE_MODE_S* pstSceneMode)
```

【参数】

参数名称	描述	输入/输出
pstSceneMode	符合当前媒体通路的自适应媒体配置	输入

【返回值】

返回值	描述
0	成功。
非 0	参见错误码。

【需求】

- 头文件：hi_scene.h
- 库文件：Null

【注意】

- pstSceneMode 不能为空。
- 不支持多线程调用。
- 必须在自适应模块初始化之后调用。
- 自适应媒体参数中的使能关系必须和当前媒体通路一致。

【举例】

参考 scene_sample.c。

HI_SCENE_Pause

【描述】

暂停或者恢复自适应。

【语法】

```
HI_S32 HI_SCENE_Pause(HI_BOOL bEnable)
```

【参数】

参数名称	描述	输入/输出
bEnable	当 True 的时候，暂停自适应。当 False 的时候，恢复自适应	输入

【返回值】

返回值	描述
0	成功。
非 0	参见错误码。

【需求】

- 头文件：hi_scene.h
- 库文件：Null

【注意】

- 在外部媒体通路销毁之前，必须先暂停自适应。因为自适应会不停配置 ISP 属性，如果 ISP 因为通路销毁而 stop，那么自适应配置 ISP 就会报错。
- 在外部媒体通路使能之后，启动自适应的时候，需要恢复自适应。
- 必须在自适应模块初始化之后调用。

【举例】

参考 scene_sample.c。

HI_SCENE_GetSceneMode

【描述】

获取自适应媒体参数

【语法】

```
HI_S32 HI_SCENE_GetSceneMode(HI_SCENE_MODE_S* pstSceneMode)
```

【参数】

参数名称	描述	输入/输出
pstSceneMode	当前使用的自适应媒体配置。	输出

【返回值】

返回值	描述
0	成功。
非 0	参见错误码。

【需求】

- 头文件：hi_scene.h
- 库文件：Null

【注意】

必须在自适应模块初始化之后调用。

【举例】

参考 scene_sample.c。

HI_SCENE_Deinit

【描述】

去初始化 SCENE 模块。

【语法】

```
HI_S32 HI_SCENE_Deinit(HI_VOID)
```

【参数】

无。

【返回值】

返回值	描述
0	成功。
非 0	参见错误码。

【需求】

- 头文件：hi_scene.h
- 库文件：Null

【注意】

必须在自适应模块初始化之后调用。

【举例】

参考 scene_sample.c。

4 数据类型

相关数据类型、数据结构定义如下：

- HI_SCENE_PARAM_S：自适应调试参数总集。
- HI_SCENE_MODE_S：自适应媒体参数。
- HI_SCENE_PIPE_MODE_E：媒体通路模式。
- HI_SCENE_PIPE_ATTR_S：一个 PIPE 上的自适应媒体参数。
- HI_SCENE_PIPE_TYPE_E：Pipe 的类型。

HI_SCENE_PARAM_S

【说明】

自适应调试参数总集，即所有的自适应调试参数被包括在这个数据结构中。

【定义】

```
typedef struct hiSCENE_PARAM_S
{
    HI_SCENE_PIPE_PARAM_S astPipeParam[HI_SCENE_PIPETYPE_NUM];
} HI_SCENE_PARAM_S;
```

【成员】

成员名称	描述
astPipeParam[HI_SCENE_PIPETYPE_NUM]	自适应调试参数数组，一组调试参数对应着数组中的一项。

【芯片差异】

无。

【注意事项】

不同 Sensor 当调试参数不同时，HI_SCENE_PIPE_PARAM_S 数据结构的内部成员也不同。

【相关数据类型及接口】

HI_SCENE_SetSceneMode

HI_SCENE_MODE_S

【说明】

自适应媒体参数。当外部媒体通路建立之后，可以通过该参数告知自适应模块哪些 pipe 生效，给生效的 pipe 配置相应的自适应调试参数。

【定义】

```
typedef struct hiSCENE_MODE_S
{
    HI_SCENE_PIPE_MODE_E enPipeMode;
    HI_SCENE_PIPE_ATTR_S astPipeAttr[HI_SCENE_PIPE_MAX_NUM];
} HI_SCENE_MODE_S;
```

说明

宏定义如下：

#define HI_SCENE_PIPE_MAX_NUM (8)

【成员】

成员名称	描述
enPipeMode	当前媒体通路模式是线性、WDR 还是 HDR
astPipeAttr[HI_SCENE_PIPE_MAX_NUM]	当前媒体通路上各 Pipe 的自适应媒体参数

【芯片差异】

无。

【注意事项】

- 当媒体通路由线性切换到 WDR，或者切换分辨率之后，一般要重新配置一下自适应的参数，可以通过配置 HI_SCENE_PARAM_S 之后调用 HI_SCENE_SetSceneMode 来完成功能。
- 具体的使用方法请参考 sample。

【相关数据类型及接口】

- HI_SCENE_SetSceneMode
- HI_SCENE_GetSceneMode
- HI_SCENE_PIPE_MODE_E
- HI_SCENE_PIPE_ATTR_S
- HI_SCENE_PIPE_MAX_NUM

HI_SCENE_PIPE_MODE_E

【说明】

媒体通路模式，线性、WDR 还是 HDR。

【定义】

```
typedef enum hiSCENE_PIPE_MODE_E
{
    HI_SCENE_PIPE_MODE_LINEAR = 0,
    HI_SCENE_PIPE_MODE_WDR,
    HI_SCENE_PIPE_MODE_HDR,
    HI_SCENE_PIPE_MODE_BUTT
} HI_SCENE_PIPE_MODE_E;
```

【成员】

成员名称	描述
HI_SCENE_PIPE_MODE_LINEAR	线性模式
HI_SCENE_PIPE_MODE_WDR	WDR 模式
HI_SCENE_PIPE_MODE_HDR	HDR 模式
HI_SCENE_PIPE_MODE_BUTT	无效模式

【芯片差异】

无。

【注意事项】

- 不同模式曝光量的使用方式不同，同时不同模式所要配置的参数模块也不一样。
- 具体的使用方法请参考 sample。

【相关数据类型及接口】

HI_SCENE_MODE_S

HI_SCENE_PIPE_ATTR_S

【说明】

一个 Pipe 上的自适应媒体参数。

【定义】

```
typedef struct hiSCENE_PIPE_ATTR_S
{
    HI_BOOL bEnable;
    HI_HANDLE MainPipeHdl;
```

```
    HI_HANDLE VcapPipeHdl;
    HI_HANDLE PipeChnHdl;
    HI_HANDLE VencHdl;
    HI_HANDLE VpssHdl;
    HI_HANDLE VPortHdl;
    HI_U8 u8PipeParamIndex;
    HI_SCENE_PIPE_TYPE_E enPipeType;
} HI_SCENE_PIPE_ATTR_S;
```

【成员】

成员名称	描述
bEnable	当前 Pipe 的使能关系，true 代表着打开，false 代表关闭。
MainPipeHdl	和当前 Pipe 一组的主 ISP 的 Pipe。当 MainPipeHdl 与 VcapPipeHdl 不一致，表示 MainPipeHdl 中的 ISP 控制 sensor，接收该 sensor 数据的除了 MainPipeHdl 还有 VcapPipeHdl，但是 VcapPipeHdl 上的 ISP 只做图像处理并不控制 Sensor。
VcapPipeHdl	当前 Pipe 的句柄
PipeChnHdl	与当前 Pipe 连接的 PipeChn
VpssHdl	与当前 Pipe 连接的 VpssGrp 的 Hdl
VPortHdl	与当前 Pipe 连接的 VpssGrp 的 Chn 的 Hdl
VencHdl	与当前 Pipe 连接的 VencHdl 的 Hdl
u8PipeParamIndex	当前 Pipe 上应该配置什么样的自适应参数，值为 HI_SCENE_PARAM_S 中 astPipeParam[HI_SCENE_PIPETYPE_NUM]的数组下标值。
enPipeType	当前 Pipe 的类型

【芯片差异】

无。

【注意事项】

目前仅支持动态调整与 Pipe 相连的一个 PipeChn、一个 Vpss 和一个 Venc，已经足够覆盖业务场景，后续可以根据业务调整与 Pipe 绑定的句柄数量。

【相关数据类型及接口】

HI_SCENE_MODE_S

HI_SCENE_PIPE_TYPE_E

【说明】

Pipe 的类型，当前 Pipe 上做拍照业务还是录像业务。

【定义】

```
typedef enum hiSCENE_PIPE_TYPE_E
{
   HI_SCENE_PIPE_TYPE_SNAP = 0,
   HI_SCENE_PIPE_TYPE_VIDEO,
   HI_SCENE_PIPE_TYPE_BUTT
} HI_SCENE_PIPE_TYPE_E;
```

【成员】

成员名称	描述
HI_SCENE_PIPE_TYPE_SNAP	当前 Pipe 上做拍照业务
HI_SCENE_PIPE_TYPE_VIDEO	当前 Pipe 上做录像业务
HI_SCENE_PIPE_TYPE_BUTT	无效类型

【芯片差异】

无。

【注意事项】

无。

【相关数据类型及接口】

HI_SCENE_PIPE_ATTR_S

5 错误码

SCENE 提供的错误码如表 5-1 所示。

表5-1 SCENE 模块的错误码

错误代码	宏定义	描述
0x80000001	HI_SCENE_EINVAL	非法参数
0x80000002	HI_SCENE_ENOTINIT	模块未初始化
0x80000003	HI_SCENE_ENONPTR	参数中有空指针
0x80000004	HI_SCENE_EOUTOFRANGE	参数超出范围
0x80000005	HI_SCENE_EINTER	内部错误
0x80000006	HI_SCENE_EINITIALIZED	已初始化
-1	HI_FAILURE	操作失败

6 配置文件和参数调节说明

6.1 ISP 相关参数

由于 Hi3519AV100 的 ISP 支持多 pipe，该部分参数由 vi_pipe 参数进行区分，例如四路拼接的应用，vi_pipe 为 0/1/2/3 分别配置相关参数。当前 ISP 相关的参数分为静态参数和动态参数，静态参数即参数不需要根据 ISO、曝光量、曝光比进行联动，配置一次即可；动态参数需要根据 ISO、曝光量或者曝光比进行联动。

当前 Sensor 驱动中，Sharpen、Demosaic、LDCI、BayerNR、AntiFalseColor、Saturation 等模块本身使用的是 Auto 接口，且接口内部根据 ISO 进行联动，故自适应中不再涉及这些模块。

- 针对线性模式：自适应中涉及到动态参数需要根据曝光量联动的如下：
  - AE 目标值和 AE offset；
  - Gamma 曲线；
  - LDCI 的使能；
  - Dehaze 的手动强度；
  - Mesh-Shading 的强度；

  自身中涉及到动态参数需要根据 ISO 进行联动：

  当前 3DNR 参数由于采用的是 Manual 接口，自适应内部手动依据 ISO 联动插值处理；
- 针对 WDR 模式：当前涉及到动态参数需要根据曝光量联动的如下：
  - AE 目标值和 AE offset；
  - Gamma 曲线；
  - LDCI 的使能；
  - Dehaze 的手动强度；
  - Mesh-Shading 的强度；
  - DRC 自定义曲线

  动态参数需要根据 ISO 进行联动：
  - 当前 3DNR 参数由于采用的是 Manual 接口，自适应内部手动根据 ISO 联动插值处理；

- DRC 的细节层参数如 LocalMixingBrightMax、LocalMixingBrightMin 等，具体可以参考表 6-11。

6.1.1 AE 相关参数

当前 AE 模块配置的参数包括静态参数和动态参数。静态参数主要包括 AE 的容忍值、AE 统计信息的权重表、AE 的扩展分配路线表、AE 收敛速度、系统最大增益的限制；动态参数主要包括 AE 的目标值和 offset 根据曝光量的联动。

AE 模块配置的静态参数如表 6-1 所示。

表6-1 AE 配置的静态参数

变量	含义	值域
AERouteExValid	AE 扩展分配路线是否生效开关，该值为 HI_TRUE 时使用扩展分配路线，否则使用普通 AE 分配路线。	[0-1]
AERunInterval	AE 算法运行的间隔，取值范围为[1,255]，取值为 1 时表示每帧都运行 AE 算法，取值为 2 时表示每 2 帧运行 1 次 AE 算法，依此类推。建议该值设置不要大于 2，否则 AE 调节速度会受到影响。WDR 模式时，该值建议设置为 1，这样 AE 收敛会更加平滑。	[1-255]
AutoSpeed	默认初始 AE 调节速度，海思 AE 算法采用该值作为初始 100 帧的调节速度，可用于运动 DV 加速启动。 建议关注快速启动的产品形态可将该值配置为 128。若不设置，该值默认为 0。AE 算法内部会将该值钳位至[64, 255]。	[0, 255]
AutoTolerance	AE 调整容忍值。	[0, 255]
AutoBlackDelayFrame	图像亮度小于目标亮度时间超过 u16BlackDelayFrame 帧时，AE 开始调节	[0, 65535]
AutoWhiteDelayFrame	图像亮度大于目标亮度时间超过 u16WhiteDelayFrame 帧时，AE 开始调节。	[0, 65535]
AutoSysGainMax	系统增益范围，设置最大值和最小值，10bit 小数精度。	[0x400, 0xFFFFFFFF]，具体范围与 Sensor 相关。
TotalNum	AE 扩展分配路线节点的个数	[0-16]
RouteEXIntTime	AE 扩展分配路线节点曝光时间，单位为微秒(us)	(0x0, 0xFFFFFFFF]，具体和 Sensor 有关
RouteEXAGain	AE 扩展分配路线节点的模拟增益，10bit 精度。	[0x400, 0x3FFFFF]，具体和 Sensor 有关。

变量	含义	值域
RouteEXDGain	AE 扩展分配路线节点的数字增益，10bit 精度。	[0x400, 0x3FFFFF]，具体和 Sensor 有关。
RouteEXISPDGain	AE 扩展分配路线节点的 ISP 数字增益，10bit 精度。	[0x400, 0x40000] ，具体和 Sensor 有关。
AEWeight[15][17]	15x17 个区间的 AE 权重表	[0, 15]

表6-2 AE 配置的动态参数

变量	含义	值域
aeExpCount	根据不同照度调整画面的亮度，此值为分组的个数。	[0-8]
aeExpLtoHThresh	从亮到暗变化时，曝光等级亮到暗阈值。	[最小曝光时间(微秒)*最小系统增益的倍数~最大曝光时间(微秒)*最大系统增益的倍数]
aeExpHtoLThresh	从暗到亮变化时，曝光等级暗到亮阈值。	[最小曝光时间(微秒)*最小系统增益的倍数~最大曝光时间(微秒)*最大系统增益的倍数]
AutoCompesation	AE 亮度目标值，建议用 0x38~0x40。	[0, 255]
AutoHistOffset	感兴趣区域对统计平均值影响的最大程度。 默认值为 0x10。	[0, 255]

6.1.2 Global CAC 相关参数

当前 CAC 模块只配置了 Global CAC 的静态参数。

Global CAC 模块配置的静态参数如表 6-3 所示。

表6-3 Global CAC 配置的静态参数

变量	含义	值域
GlobalCacEnable	横向色差校正使能。	[0,1]
VerCoordinate	光学中心垂直坐标。	[0,Height]，Height 为 ISP 处理图像的高度
HorCoordinate	光学中心水平坐标。	[0,Width]，Width 为 ISP 处理图像的宽度
ParamRedA	R 通道对应的半径多项式的系数 a	[-256,255]

变量	含义	值域
ParamRedB	R 通道对应的半径多项式的系数 b	[-256,255]
ParamRedC	R 通道对应的半径多项式的系数 c	[-256,255]
ParamBlueA	B 通道对应的半径多项式的系数 a	[-256,255]
ParamBlueB	B 通道对应的半径多项式的系数 b	[-256,255]
ParamBlueC	B 通道对应的半径多项式的系数 c	[-256,255]
VerNormShift	垂直方向的归一化移位参数	[0,7]
VerNormFactor	垂直方向的归一化系数	[0,31]
HorNormShift	水平方向的归一化移位参数	[0,7]
HorNormFactor	水平方向的归一化参数	[0,31]
CorVarThr	横向色差校正的方差阈值	[0,4095]

6.1.3 LDCI 相关参数

LDCI 模块的参数当前主要是通过 Sensor 的 cmos 驱动进行配置，当前自适应主要影响的是根据曝光量来开启和关闭 LDCI 参数。

LDCI 模块配置的动态参数如表 6-4 所示。

表6-4 LDCI 模块配置的静态参数

变量	含义	值域
EnableCount	根据不同曝光量调整 LDCI 使能，此值为分组的个数。	[0,6]
EnableExpThreshLtoH	从亮到暗变化时，曝光等级亮到暗阈值	[最小曝光时间(微秒)*最小系统增益的倍数~最大曝光时间(微秒)*最大系统增益的倍数]
Enable	LDCI 模块的使能	[0,1]

6.1.4 Dehaze 相关参数

Dehaze 模块当前配置参数包括静态参数和动态参数，静态参数主要是配置 Dehaze 模块的基本属性，如使能配置、自定义曲线是否使能、去雾手动和自动类型的选择以及自定义曲线表。动态参数主要将去雾的强度和曝光量进行联动。

Dehaze 模块配置的静态参数如表 6-5 所示。

表6-5 Dehaze 模块配置的静态参数

变量	含义	值域
bEnable	使能 Dehaze 功能。	[0,1]
bDehazeUserLutEnable	用户自定义去雾曲线使能	[0,1]
DehazeOpType	操作模式	[0,1]
DehazeLut[256]	用户自定义去雾曲线表	[0,255]

Dehaze 模块配置的动态参数如表 6-6 所示。

表6-6 Dehaze 模块配置的动态参数

变量	含义	值域
ExpThreshCnt	根据不同照度调整去雾的手动强度，此值为分组的个数。	[0,5]
ExpThreshLtoH	从亮到暗变化时，曝光等级亮到暗阈值	[最小曝光时间(微秒)*最小系统增益的倍数~最大曝光时间(微秒)*最大系统增益的倍数]
ManualDehazeStr	手动模式下去雾强度	[0,255]

6.1.5 Mesh-Shading 相关参数

Mesh-Shading 模块当前配置参数包括静态参数和动态参数，静态参数主要是配置 Mesh-Shading 模块的使能配置。动态参数主要将 Shading 校正的强度和曝光量进行联动。

Mesh-Shading 模块配置的静态参数如表 6-7 所示。

表6-7 Mesh-Shading 模块配置的静态参数

变量	含义	值域
bEnable	使能 Mesh-shading 功能。	[0,1]

Dehaze 模块配置的动态参数如表 6-8 所示。

表6-8 Mesh-Shading 模块配置的动态参数

变量	含义	值域
ExpThreshCnt	根据不同照度调整去雾的手动强度，此值为分组	[0,5]

变量	含义	值域
	的个数。	
ExpThreshLtoH	从亮到暗变化时，曝光等级亮到暗阈值	[最小曝光时间(微秒)*最小系统增益的倍数~最大曝光时间(微秒)*最大系统增益的倍数]
ManualStrength	用来全局纠正 Mesh Shading 标定后的强度。当镜头 shading 很严重的时候，画面四个角补偿的增益很大，容易导致噪声很大，而且画面有格子现象。这个调整 u16MeshStrength 小于 4096，可以减少 Mesh Shading 的补偿值，让四个角的补偿增益较小一些，达到优化噪声和画面格子问题。	[0, 65535]

6.1.6 WDRExposureAttr 相关参数

WDRExposureAttr 模块当前配置参数包括静态参数，静态参数主要配置曝光比的类型、手动曝光比的大小自动曝光比的上限和下限。

WDRExposureAttr 模块配置的参数如表 6-9 所示。

表6-9 WDRExposureAttr 模块配置的静态参数

变量	含义	值域
ExpRatioType	仅在多帧合成 WDR 模式下有效。 OP_TYPE_AUTO：根据场景自动计算长短帧曝光比；OP_TYPE_MANUAL：手动配置长短帧曝光比。	[0,1]
ExpRatioMax	仅在多帧合成 WDR 模式下有效。 当 enExpRatioType 为 OP_TYPE_AUTO 时，u32ExpRatioMax 表示最长帧与最短帧曝光时间比值的最大值。即 2 帧合成时表示 S/VS 曝光比的最大值，3 帧合成时表示 M/VS 曝光比的最大值，4 帧合成时表示 L/VS 曝光比的最大值。 当 enExpRatioType 为 OP_TYPE_MANUAL 时，u32ExpRatioMax 无效。 6bit 小数精度，0x40 表示曝光比为 1 倍。	[0x40, 0x4000]。
ExpRatioMin	仅在多帧合成 WDR 模式下有效。 当 enExpRatioType 为 OP_TYPE_AUTO 时，u32ExpRatioMin 表示长帧曝光时间与短帧曝光时间比值的最小值。 当 enExpRatioType 为 OP_TYPE_MANUAL 时，u32ExpRatioMin 无效。	[0x40, u32ExpRatioMax]

变量	含义	值域
	格式为无符号 6.6bit 定点，0x40 表示长帧曝光时间与短帧曝光时间的比值为 1 倍。默认值为 0x40。	
ExpRatio	仅在多帧合成 WDR 模式下有效。 当 enExpRatioType 为 OP_TYPE_AUTO 时，au32ExpRatio 无效。 当 enExpRatioType 为 OP_TYPE_MANUAL 时，au32ExpRatio 为可读写，表示多帧合成 WDR 相邻 2 帧曝光比期望值。其中 au32ExpRatio[0]作为 S/VS 曝光比，au32ExpRatio[1]作为 M/S 曝光比，au32ExpRatio[2]作为 L/M 曝光比。上述 VS，S，M，L 分别表示极短帧，短帧，中帧和长帧的曝光时间，2 帧合成时只有 VS 和 S；3 帧合成时采用 VS，S 和 M；4 帧合成时采用 VS，S，M 和 L。6bit 小数精度，0x40 表示曝光比为 1 倍。	[0x40, 0xFFF]。

6.1.7 FSWDR 相关参数

FSWDR 模块当前配置参数主要为动态参数，动态参数主要包括运动补偿的使能、合成模块去噪的操作模式、合成模块去噪的强度。

6.1.8 Gamma 相关参数

在不同场景中对比度不一样，要求也不一样，因此需要根据不同曝光量设置不同的 Gamma。Gamma 相关参数如表 6-10 所示。

表6-10 Gamma 相关参数

变量	含义	值域
gammaInterval	Gamma 切换变换速度。	[1, 4294967295] 表示从前一条 Gamma 曲线变换到下一条 Gamma 曲线所需要的次数
gammaExpCount	曝光等级个数。	[1, 10]
gammaExpThreshLtoH	曝光等级变化阈值（画面从亮到暗）。	[最小曝光时间(微秒)*最小系统增益的倍数~最大曝光时间(微秒)*最大系统增益的倍数]
gammaExpThreshHtoL	曝光等级变化阈值（画面从暗到亮）。	[最小曝光时间(微秒)*最小系统增益的倍数~最大曝光时间(微秒)*最大系统增益的倍数]
Table_K	第 K 组 gamma。	[1, 4095]

6.1.9 DRC 相关参数

DRC 当前配置的参数主要包括静态参数和动态参数。静态参数主要是 DRC 的曲线选择参数、DRC 强度的类型。动态参数包括 DRC 细节增强参数如 LocalMixingBrightMax、LocalMixingBrightMin、u8LocalMixingDarkMax、u8LocalMixingDarkMin 等以及 DRC 自定义曲线。当前 Hi3519AV100， DRC 自定义曲线当前设置两组曲线 DRCToneMappingValue1 和 DRCToneMappingValue2，两组曲线的融合主要根据曝光比的大小进行插值处理，针对强宽动态场景和弱宽动态场景分别使用不同的 DRC 自定义曲线。

DRC 相关参数如表 6-11 所示。

表6-11 DRC 相关参数

变量	含义	值域
RationCnt	不同曝光比的个数。	[0,6]
RationLevel	不同曝光比的分级阈值。	WDR 模式使用。 [1,256]
IsoCnt	不同 ISO 的个数。	[0,7]
IsoLevel	不同 ISO 的分级阈值。	[最小系统 ISO~最大系统 ISO]，具体和 Sensor 相关
Interval	DRC 参数变换的速度	[0, 4294967295]
LocalMixingBrightMax	用来控制正向细节的增益最大值。	[0x0, 0x80]
LocalMixingBrightMin	用来控制正向细节的增益最小值。	[0x0, 0x40]
u8LocalMixingDarkMax	用来控制负向细节的增益最大值。	[0x0, 0x80]
u8LocalMixingDarkMin	用来控制负向细节的增益最小值。	[0x0, 0x40]
DetailAdjustFactor	细节微调系数。值越大，整体细节越强；值越小，整体细节越弱。	[-0xF, 0xF] 详见《HiISP 开发参考》
SpatialFltCoef	控制空域滤波系数。值越大，运动 halo 越不明显，细节越少，值越小，细节表现越好，运动 halo 越明显。	[0x0, 0xA] 详见《HiISP 开发参考》
u8RangeFltCoef	控制值域滤波系数。值越大，halo 越明显；值越小，halo 表现越好，但是在强边缘处可能会出现边线。	[0x0, 0xA] 详见《HiISP 开发参考》
GradRevMax	去边线强度。值越大，边线减弱越明显，但是可能会带来细节损失	[0, 0xFF] 详见《HiISP 开发参考》
GradRevThr	边线检测阈值。值越大，边线减弱越明显，但是可能会带来细节损失。	[0, 0xFF] 详见《HiISP 开发参考》
Compress	用来生成 Asymmetry curve，compress 越	[0x64,0xC8]

变量	含义	值域
	大,在垂直方向压缩曲线程度越大，整体亮度变低。	详见《HiISP 开发参考》
Stretch	用来生成 Asymmetry curve，值越小，整体图像越亮	[0x1E,0x3C]

6.1.10 3DNR 相关参数

当前 3DNR 接口主要配置的是静态参数，使用的是 3DNR 的 Maunal 接口，这边需要配置的各个 ISO 下的 3DNR 参数，自适应根据 ISO 进行内部插值处理。

3DNR 参数具体说明详见《Hi3519AV100/Hi3556AV100 3DNR 参数配置说明》文档。