Rockchip Developement Guide ISP30

文件标识：RK-KF-GX-612

发布版本：V1.2.3

日期：2022-2-19

文件密级：□绝密 □秘密 □内部资料 ■公开

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址： 福建省福州市铜盘路软件园A区18号

网址： www.rock-chips.com

客户服务电话： +86-4007-700-590

客户服务传真： +86-591-83951833

客户服务邮箱： fae@rock-chips.com

前言

概述

本文旨在描述RkAiq（Rk Auto Image Quality）模块的作用，整体工作流程，及相关的API接口。主要给
使用RkAiq模块进行ISP功能开发的工程师提供帮助。

产品版本``

芯片名称	内核版本
RK3588	Linux 5.10

读者对象

本文档（本指南）主要适用于以下工程师：

ISP模块软件开发工程师

系统集成软件开发工程师

各芯片系统支持状态

芯片名称	BuildRoot	Debian	Yocto	Android
RK3588	Y	N	N	Y

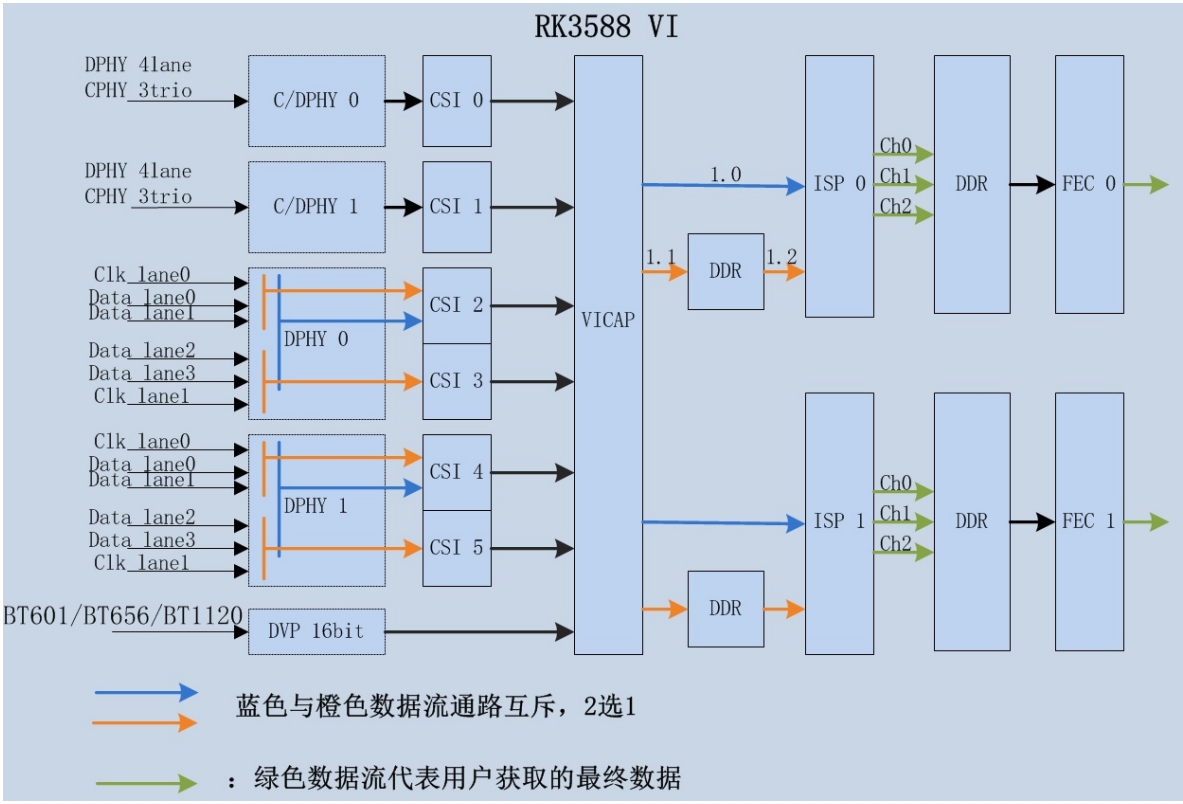
修订记录

版本号	作者	修改日期	修改说明
v0.0.1	All	2021-11-04	RK3588 ISP3.0 alpha版
v1.0.0	All	2022-1-1	1. AE / AWB / AF / CCM / 3DLut / Gamma / Dhz&Ehz / Merge / DRC / BLC / NR / Sharp 模块更新 2. 统计信息 章节更新 3.Demosaic / DPCC / FEC / LDCH / GIC 尚未提供
v1.1.0	ALL	2022-01-11	1. 修订 系统控制 API 部分 2. 增加 CSM/CPROC/IE 模块API说明 3. 增加 Camera组API说明 4. 增加LDCH / GIC / DPCC / Debayer 模块说明 5. 更新 ”统计信息“ 章节，AE统计值结构体相关说明更新
v1.2.2	朱林靖 欧阳亚凤	2022-1-25	1. “统计信息”章节，RKAiqExpI2cParam_s结构体更新说明 2. "Camera组"章节，RK_PS_SrcOverlapMap结构体更新说明 3. NR/Sharp 章节参数勘误
v1.2.3	徐鸿飞 武强	2022-2-19	1.Debayer / DPCC 章节API 以及数据类型更新说明

目录

总体概要

结构图



RK3588 VI作为视频输入模块总称，其中包含：

- 接口以及数据采集部分：C/DPHY、DVP、CSI、VICAP
- 图像处理部分：ISP-0，ISP-1、FEC-0、FEC-1

RK3588 ISP属于RK ISP v3.0版本，后续文档以ISP30标识。

工作模式

单ISP单CIS(cmos image sensor)：

- Raw直通/在线模式：结构图中蓝色箭头指示数据流1.0通路
- Raw回读/离线模式：结构图中橙色箭头指示数据流1.1，1.2通路

双ISP 2合1模式 for 单CIS：

- 2合一模式支持ISP-0 / ISP-1分别处理单个CIS图像的左右部分，以此来支持大分辨率CIS。

多CIS：

- 单ISP采用Raw回读/离线模式下，可以采用时分复用的方式支持多CIS的输入。

性能及约束概述

- 支持分辨率及吞吐率：

工作模式	吞吐率	支持最大分辨率	支持最小分辨率
单ISP单CIS	16Mega@30fps	4672x3504	208x128
双ISP 2合1单CIS	32Mega@30fps	8064x6048	\

- 支持Raw12数据输入。

ISP30支持处理Raw12数据。Raw14数据输入将低位丢弃成Raw12处理。位宽不足12bit图像数据输入将低位补0成Raw12处理。

- 支持Bayer color CIS、Mono CIS。
- 支持3合1、2合1 多帧HDR。支持HDR CIS 类型如下：

Multi exposure HDR	RK Platform support	CIS	应用
Sequential / Framebase HDR	ISP20: RV1109/RV1126 etc ISP21: RK356X etc ISP30: RK3588	OV4689	IPC/CVR
Staggered / DOL HDR	ISP20: RV1109/RV1126 etc ISP21: RK356X etc ISP30: RK3588	OS04A10 IMX464 AR0239	IPC/CVR
DCG HDR	ISP20: RV1109/RV1126 etc ISP21: RK356X etc ISP30: RK3588	OS04A10 IMX585	IPC/CVR
SME HDR	N	IMX378	消费类
BME HDR	N	IMX258	消费类
QBC HDR	N	OV50C40 IMX586	消费类
Lateral overflow HDR	N	AR0821	IPC
Splite-diode pixel HDR	N	OV10640	CVR

- Raw直通/在线模式下，ISP30数据输出至DDR通知后级的最小时延：约图像帧传输时间/15。
- ISP时分复用方式下，单ISP最多支持4路数据源输入。
- Raw直通/在线模式下，最小图像帧消隐136行，最小图像行消隐 16 像素

概述

ISP30 包含了一系列的图像处理算法模块，主要包括：暗电流矫正、坏点矫正、3A、HDR、镜头阴影矫正、镜头畸变矫正、3DLUT、去噪（包括RAW域去噪，多帧降噪，颜色去噪等）、锐化等。

ISP30包括硬件算法实现及软件逻辑控制部分，RkAiq即为软件逻辑控制部分的实现。

RkAiq软件模块主要实现的功能为：从ISP驱动获取图像统计，结合IQ Tuning参数，使用一系列算法计算出新的ISP、Sensor等硬件参数，不断迭代该过程，最终达到最优的图像效果。

功能描述

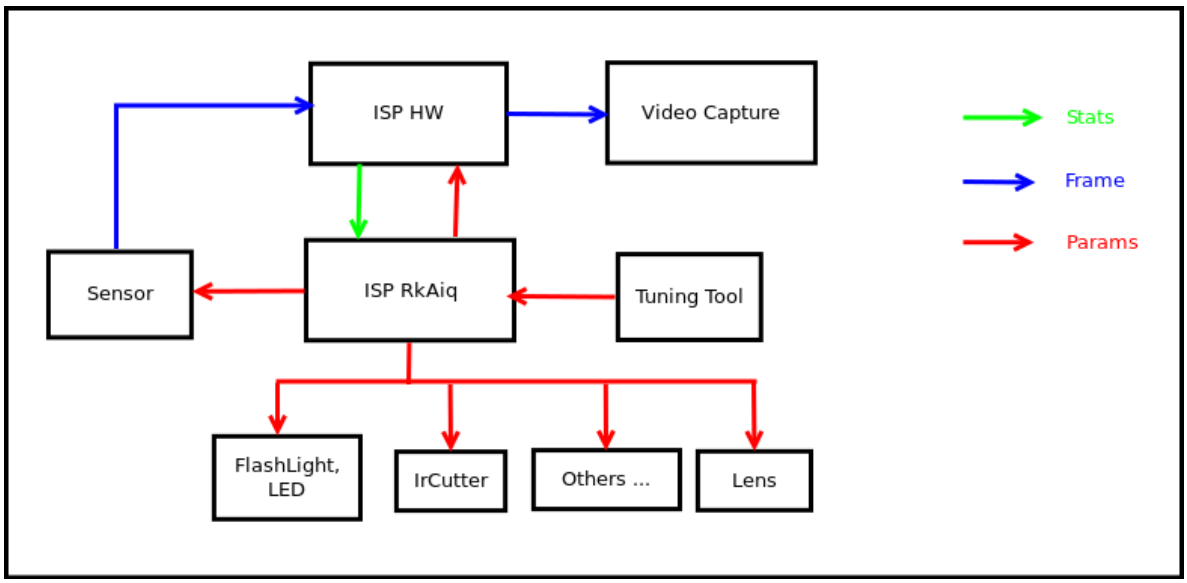


图1-1 ISP21 系统框图

ISP30总体软硬件框图如图1-1所示。Sensor输出数据流给ISP HW，ISP HW再输出经过一系列图像处理算法后的图像。RkAiq不断从ISP HW获取统计数据，并经过3A等算法生成新的参数反馈给各硬件模块。Tuning tool可在线实时调试参数，调试好后可保存生成新的iq参数文件。

RkAiq架构

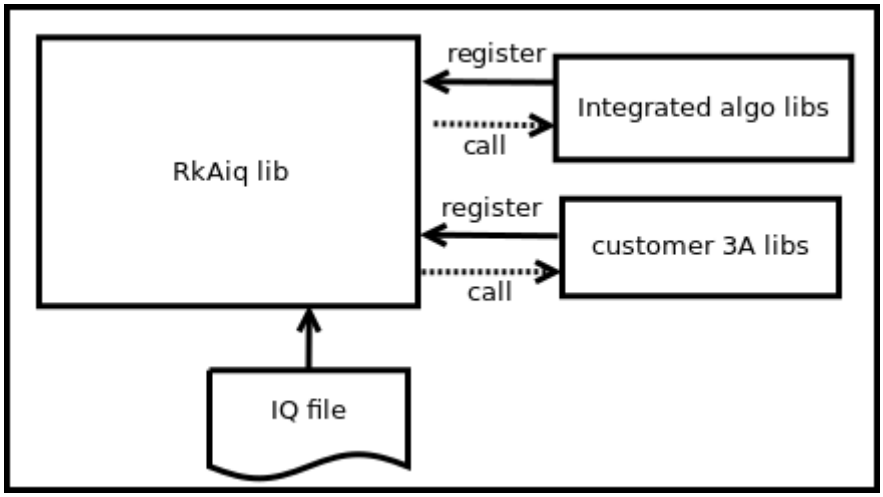


图1-2 RkAiq总体架构图

ISP30 RkAiq软件设计思路如图1-2所示。主要分成以下四个部分：

1. RkAiq lib 动态库。该库包含了主要的逻辑部分，负责从驱动获取统计，并传送给各个 算法库。
2. Integrated algo libs。Rk提供的静态算法库，已默认注册到RkAiq lib动态库。
3. customer 3A libs。客户可根据算法库接口定义实现自己的3A算法库，或者其他算法库。将自定义算法库注册给RkAiq lib动态库后，可根据提供的接口选择跑自定义库还是跑Rk库。
4. IQ file。iq tuning结果文件，保存的是算法相关参数以及CIS等一些系统静态参数。

软件架构

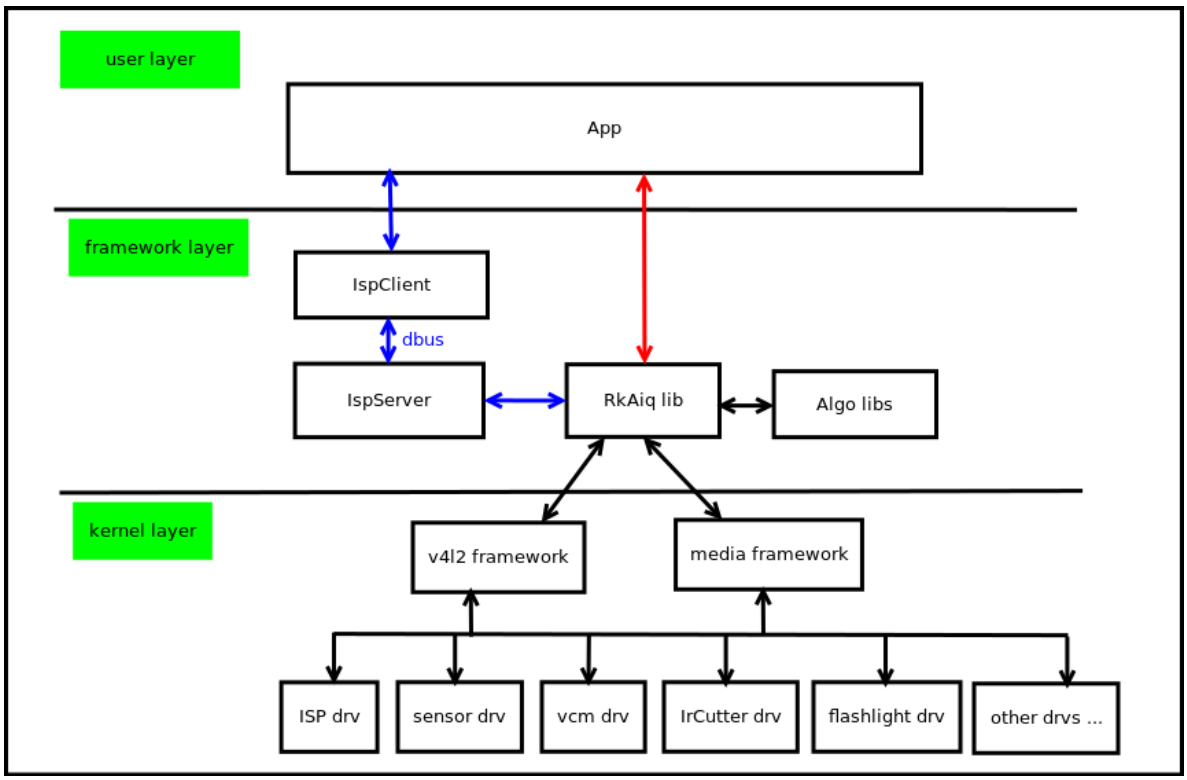


图1-3 软件架构框图

ISP30 软件框图如图1-3所示。主要分成以下三层：

1. kernel layer。该层包含所有Camera系统的硬件驱动，主要有ISP驱动、sensor驱动、vcm驱动、flashlight驱动、IrCutter驱动等等。驱动都基于V4L2及Media框架实现。
2. framework layer。该层为RkAiq lib的集成层，Rkaiq lib有两种集成方式：

- IspServer 方式
  该方式Rkaiq lib跑在 IspServer独立进程，客户端通过dbus与之通信。此外，该方式可为v4l-ctl等现有第三方应用，在不修改源码的情况下，提供具有ISP调试效果的图像。
- 直接集成方式
  RkAiq lib可直接集成进应用。

3. user layer。用户应用层。

软件流程

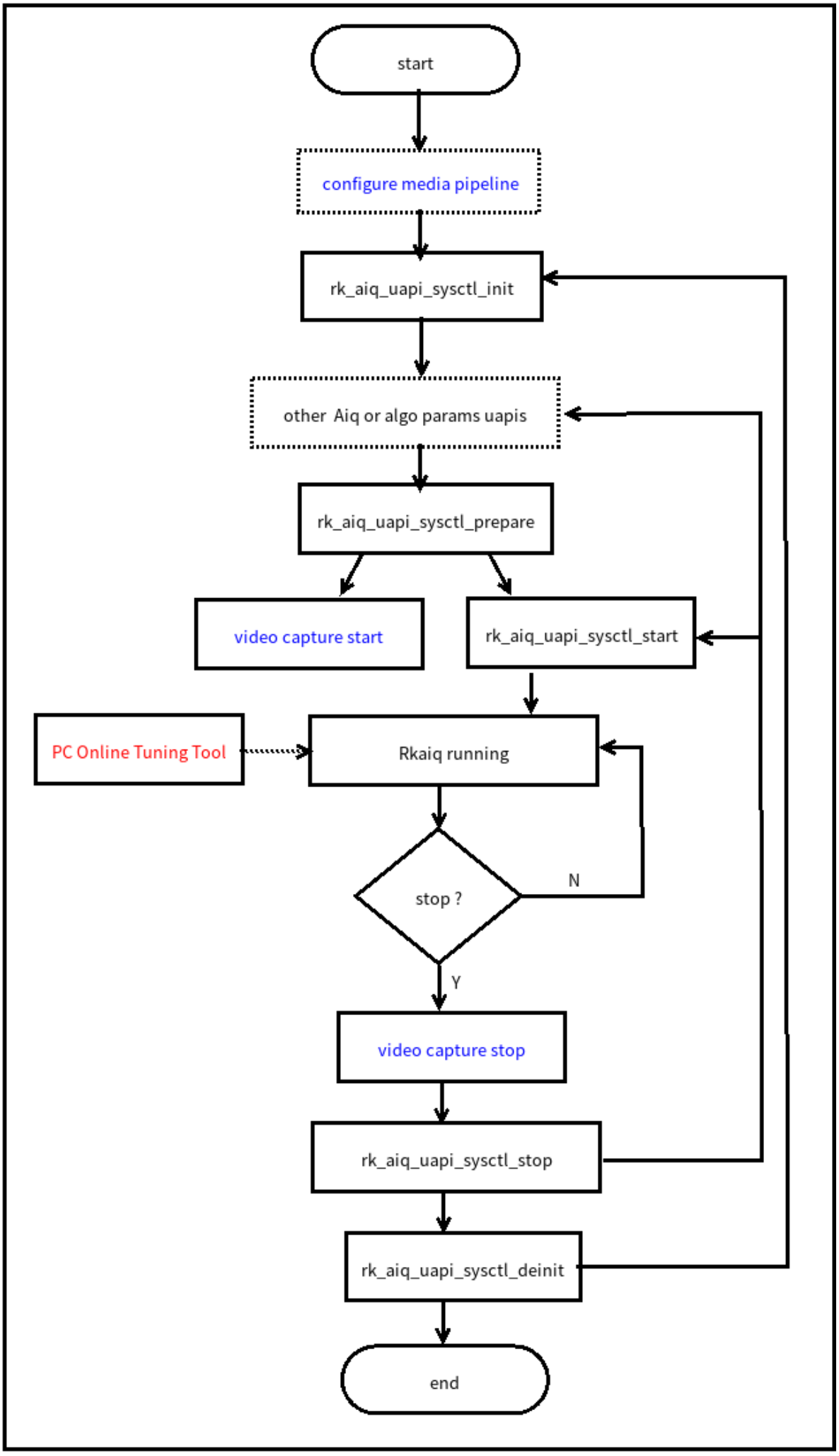

start
configure media pipeline
rk_aiq_uapi_sysctl_init
other Aiq or algo params uapis
rk_aiq_uapi_sysctl_prepare
video capture start
rk_aiq_uapi_sysctl_start
PC Online Tuning Tool
Rkaiq running
stop ?
N
Y
video capture stop
rk_aiq_uapi_sysctl_stop
rk_aiq_uapi_sysctl_deinit
end

图1-4 流程图

RkAiq接口调用流程如图1-4所示。图中虚线框部分为可选部分，蓝色字体部分为应用需要配合RkAiq流程所作的配置。

- configure media pipeline。可选项，配置ISP30 pipeline，如sensor输出分辨率等等，驱动已有默认配置。
- rk_aiq_uapi2_sysctl_init。初始化RkAiq，包括IQ tuning参数及各算法库初始化。
- other Aiq or algo params uapis。可选项，可通过各算法提供的API接口配置需要的参数，以及注册第三方算法库等等。
- rk_aiq_uapi2_sysctl_prepare。准备各算法库及各硬件模块的初始化参数，并设置到驱动。
- video capture start。该流程为应用端ISP数据流的开启，该流程需要在rk_aiq_uapi2_sysctl_prepare后调用。
- rk_aiq_uapi2_sysctl_start。启动RkAiq内部流程，该接口调用成功后，sensor开始输出数据，ISP开始处理数据，并输出处理后的图像。
- Rkaiq running。RkAiq不断从ISP驱动获取统计数据，调用3A等算法计算新参数，并应用新参数到驱动。
- PC Online Tuning Tool。PC端可通过Tuning Tool在线调整参数。
- video capture stop。停止RkAiq流程前需要先停止数据流部分。
- rk_aiq_uapi2_sysctl_stop。停止 RkAiq running 流程。可调整参数后再启动或者直接再启动。
- rk_aiq_uapi2_sysctl_deinit。反初始化RkAiq。

API说明

- RKAiq提供的API分为两个级别：功能级别API 与 模块级别API。
  - **功能级别API**：基于模块级别API封装而成，主要是面对产品应用基于该模块的一些简单功能设计。
  - **模块级别API**：该模块的详细参数设置以及查询，未对功能进行API区分。
- RKAiq提供的模块级API支持同步以及异步模式，功能级API默认采用模块级API的同步模式进行封装。
  - **同步模式(Sync)**：该模式为阻塞式API，AIQ基本以视频帧为颗粒度来进行参数的更新，同步模式下，调用API的线程有可能被阻塞的时间 <= 视频帧周期时间。
  - **异步模式(ASync)**：该模式为非阻塞式API，AIQ框架会将API设置参数存储在内部参数缓冲中，待最近一个ISP参数设置时刻在设置给硬件。
- 模块级API同步异步方式使用说明

  模块级API的第2个形参结构体一般为设置参数集合，称为属性结构体。该结构体内部rk_aiq_uapi_sync_t 成员可以配置当前API的调用模式，详细参考以下数据类型定义：

rk_aiq_uapi_sync_t

【说明】
模块sync模式

【定义】

```
typedef struct rk_aiq_uapi_sync_s {
    rk_aiq_uapi_mode_sync_e     sync_mode;
    bool                        done;
} rk_aiq_uapi_sync_t;
```

【成员】

成员名称	描述
sync_mode	@setAttrib:参数更新模式的标志； RK_AIQ_UAPI_MODE_DEFAULT: 默认是同步模式. RK_AIQ_UAPI_MODE_SYNC: 同步模式. RK_AIQ_UAPI_MODE_ASYNC: 异步模式. @getAttrib: RK_AIQ_UAPI_MODE_DEFAULT: 默认获取上次设置的属性(sync_mode == RK_AIQ_UAPI_MODE_ASYNC)，可能还没有生效. RK_AIQ_UAPI_MODE_SYNC: 获取当前使用的属性. RK_AIQ_UAPI_MODE_ASYNC: 与 RK_AIQ_UAPI_MODE_DEFAULT 相同.
done	@done (parsm out): 参数更新状态的标志, true 表示参数已更新，false 表示参数尚未更新。

- AIQ API参考代码按照模块进行划分，单独提供API示例，建议客户使用时可以直接参考 rkisp_demo/demo/sample

文件名	模块
sample_3dlut_module.cpp	3DLUT
sample_ccm_module.cpp	CCM
sample_csm_module.cpp	CSM
sample_ablc_module.cpp	BLC
sample_abayer2dnr_module.cpp	BayerNR 2D
sample_abayertnr_module.cpp	BayerNR 3D
sample_aynr_module.cpp	YNR
sample_acnr_module.cpp	CNR
sample_asharp_module.cpp	Sharp
sample_amerge_module.cpp	Merge(HDR)
sample_adrc_module.cpp	DRC
sample_adehaze_module.cpp	Dehaze & Enhance
sample_agamma_module.cpp	Gamma
sample_awb_module.cpp	AWB
sample_ae_module.cpp	AE
sample_af_module.cpp	AF

系统控制

功能概述

系统控制部分包含了AIQ 公共属性配置，初始化 AIQ、运行 AIQ、退出AIQ，设置 AIQ各模块等功能。

API参考

rk_aiq_uapi2_sysctl_preInit

【描述】

初始化AIQ前，预设值一些参数，如使用的IQ文件等。

【语法】

```
XCamReturn
rk_aiq_uapi2_sysctl_preInit (const char* sns_ent_name,
                             rk_aiq_working_mode_t mode,
                             const char* force_iq_file);
```

【参数】

参数名称	描述	输入/输出
sns_ent_name	sensor entity name	输入
mode	ISP工作模式	输入
force_iq_file	强制使用的 iq 文件	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

【注意】

- 应先于rk_aiq_uapi2_sysctl_init 调用
- 参数 mode 仅用来选择 IQ 文件，并不配置ISP工作模式，工作模式需要在 rk_aiq_uapi2_sysctl_prepare 中配置。如果参数 force_iq_file 非空，那么mode参数无效
- 参数 force_iq_file 为全路径
- 该函数不是必须，仅用于 rk_aiq_uapi2_sysctl_init 前更改一些默认的设置

rk_aiq_uapi2_sysctl_preInit_scene

【描述】

初始化AIQ前，选择IQ场景参数。

【语法】

```
XCamReturn
rk_aiq_uapi2_sysctl_preInit_scene (const char* sns_ent_name,
                           const char *main_scene,
                          const char *sub_scene);
```

【参数】

参数名称	描述	输入/输出
sns_ent_name	sensor entity name	输入
main_scene	主场景，值在IQ文件中定义	输入
sub_scene	子场景，值在IQ文件中定义	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

【注意】

- 应先于rk_aiq_uapi2_sysctl_init 调用
- main_scene 需要根据 IQ 文件中主场景数组来选择，默认一般是 “normal”
- sub_scene 需要根据 IQ 文件中主场景里的子场景数组来选择，默认一般是 “day”
- 由于目前默认选择的场景是 normal-day，如果要跑 hdr-day，或者其他自定义模式，则需要调用该函数进行指定，否则IQ参数可能不正确

rk_aiq_uapi2_sysctl_switch_scene

【描述】
初始化AIQ后，切换IQ场景参数。

【语法】

```
int
rk_aiq_uapi2_sysctl_switch_scene (const rk_aiq_sys_ctx_t* sys_ctx,
                           const char *main_scene,
                          const char *sub_scene);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ 上下文指针	输入
main_scene	主场景，值在IQ文件中定义	输入
sub_scene	子场景，值在IQ文件中定义	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

【注意】

- 在 rk_aiq_uapi2_sysctl_init 后调用
- main_scene 需要根据 IQ 文件中主场景数组来选择
- sub_scene 需要根据 IQ 文件中主场景里的子场景数组来选择
- 该接口涉及IQ动态切换，目前测试不充分，可能出现非预期结果，谨慎使用

rk_aiq_uapi2_sysctl_init

【描述】
初始化AIQ上下文。

【语法】

```
rk_aiq_sys_ctx_t*
rk_aiq_uapi2_sysctl_init (const char* sns_ent_name,
                          const char* iq_file_dir,
                          rk_aiq_error_cb err_cb,
                          rk_aiq_metas_cb metas_cb);
```

【参数】

参数名称	描述	输入/输出
sns_ent_name	sensor entity name	输入
iq_file_dir	标定参数文件路径	输入
err_cb	出错回调函数，可为NULL	输入
metas_cb	meta数据回调函数，可为NULL	输入

【返回值】

返回值	描述
rk_aiq_sys_ctx_t*	AIQ上下文指针

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

【注意】

- 应先于其他函数调用。

rk_aiq_uapi2_sysctl_deinit

【描述】
反初始化AIQ上下文环境。

【语法】

```
void
rk_aiq_uapi2_sysctl_deinit( rk_aiq_sys_ctx_t* ctx);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
无	无

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

【注意】

- 不应在AIQ处于start状态调用。

rk_aiq_uapi2_sysctl_prepare

【描述】
准备AIQ运行环境。

【语法】

```
XCamReturn
rk_aiq_uapi2_sysctl_prepare(const rk_aiq_sys_ctx_t* ctx,
                            uint32_t  width,
                            uint32_t  height,
                            rk_aiq_working_mode_t mode);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
width	sensor输出的分辨率宽度，仅用于校验	输入
height	sensor输出的分辨率高度，仅用于校验	输入
mode	ISP Pipeline工作模式(NORMAL/HDR)	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

【注意】

- 应在rk_aiq_uapi2_sysctl_start函数之前调用。
- 如果需要在rk_aiq_uapi2_sysctl_start之后调用本函数，那么先调用rk_aiq_uapi2_sysctl_stop函数，再调用rk_aiq_uapi2_sysctl_prepare重新准备运行环境。

rk_aiq_uapi2_sysctl_start

【描述】
启动AIQ控制系统。AIQ启动后，会不断的从ISP驱动获取3A统计信息，运行3A算法，并应用计算出的新参数。

【语法】

```
XCamReturn
rk_aiq_uapi2_sysctl_start(const rk_aiq_sys_ctx_t* ctx);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

【注意】

- 应在rk_aiq_uapi2_sysctl_prepare函数之后调用。

rk_aiq_uapi2_sysctl_stop

【描述】
停止AIQ控制系统。

【语法】

```
XCamReturn
rk_aiq_uapi2_sysctl_stop(const rk_aiq_sys_ctx_t* ctx, bool keep_ext_hw_st);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
keep_ext_hw_st	stop之后，外部设备设如(ircut/cpsl)的状态是否保持不变	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi2_sysctl_getStaticMetas

【描述】
查询sensor对应静态信息，如分辨率，数据格式等。

【语法】

```
XCamReturn
rk_aiq_uapi2_sysctl_getStaticMetas(const char* sns_ent_name,
rk_aiq_static_info_t* static_info);
```

【参数】

参数名称	描述	输入/输出
sns_ent_name	sensor entity name	输入
static_info	静态信息结构体指针	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi2_sysctl_enumStaticMetas

【描述】
枚举AIQ获取到的静态信息。

【语法】

```
XCamReturn
rk_aiq_uapi2_sysctl_enumStaticMetas(int index, rk_aiq_static_info_t*
static_info);
```

【参数】

参数名称	描述	输入/输出
index	索引号，从0开始	输入
static_info	静态信息结构体指针	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi2_sysctl_enableAxlib

【描述】
设置自定义算法库运行状态。

【语法】

```
XCamReturn
rk_aiq_uapi2_sysctl_enableAxlib(const rk_aiq_sys_ctx_t* ctx,
                                const int algo_type,
                                const int lib_id,
                                bool enable);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
algo_type	要操作的算法模块类型	输入
lib_id	算法库标识ID	输入
enable	状态设置	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

【注意】

- 如果lib_id等同于当前运行的算法库，本函数可以在除未初始化外的任何状态下调用。
- 其他情况，仅在prepared状态下调用，并且algo_type所标识的算法库将被lib_id标识的新算法库替代。

rk_aiq_uapi2_sysctl_getAxlibStatus

【描述】
获取算法库状态。

【语法】

```
bool
rk_aiq_uapi2_sysctl_getAxlibStatus(const rk_aiq_sys_ctx_t* ctx,
                                   const int algo_type,
                                   const int lib_id);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
algo_type	要操作的算法模块类型	输入
lib_id	算法库标识ID	输入

【返回值】

返回值	描述
false	关闭状态
true	使能状态

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi2_sysctl_getEnabledAxlibCtx

【描述】
获取使能算法库的上下文结构体。

【语法】

```
const RkAiqAlgoContext*
rk_aiq_uapi2_sysctl_getEnabledAxlibCtx(const rk_aiq_sys_ctx_t* ctx, const int
algo_type);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
algo_type	要操作的算法模块类型	输入

【返回值】

返回值	描述
NULL	获取失败
非NULL	获取成功

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

【注意】

- 返回的算法上下文结构体将被内部私有函数使用。对于用户自定义的算法库，该函数应在rk_aiq_uapi2_sysctl_enableAxlib之后调用，否则将返回NULL。

rk_aiq_uapi2_sysctl_setCpsLtCfg

【描述】
设置补光灯控制信息。

【语法】

```
XCamReturn
rk_aiq_uapi2_sysctl_setCpsLtCfg(const rk_aiq_sys_ctx_t* ctx,
                        rk_aiq_cpsl_cfg_t* cfg);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
cfg	补光灯配置结构体指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi2_sysctl_getCpsLtInfo

【描述】
获取补光灯控制信息。

【语法】

```
XCamReturn
rk_aiq_uapi2_sysctl_getCpsLtInfo(const rk_aiq_sys_ctx_t* ctx,
                        rk_aiq_cpsl_info_t* info);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
info	补光灯配置结构体指针	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi2_sysctl_queryCpsLtCap

【描述】
查询补光灯的支持能力。

【语法】

```
XCamReturn
rk_aiq_uapi2_sysctl_queryCpsLtCap(const rk_aiq_sys_ctx_t* ctx,
                        rk_aiq_cpsl_cap_t* cap);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
cap	补光灯支持能力查询结构体指针	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi2_sysctl_getBindedSnsEntNmByVd

【描述】
查询video结点所对应的sensor entity name。

【语法】

```
const char* rk_aiq_uapi2_sysctl_getBindedSnsEntNmByVd(const char* vd);
```

【参数】

参数名称	描述	输入/输出
vd	video路径，如/dev/video20	输入

【返回值】

返回值	描述
sensor entity name	字符串指针

【注意】

- 参数必须为ISPP scale结点路径。

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi2_sysctl_updateIq

【描述】
动态更新当前所使用的iq参数文件，不需要停止数据流。

【语法】

```
XCamReturn rk_aiq_uapi2_sysctl_updateIq(const rk_aiq_sys_ctx_t* sys_ctx, char*
iqfile);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
iqfile	新的iq文件	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【注意】

- iqfile 需要为全路径。
- 更新iq参数，并不意味着能切换运行模式，如需要切换hdr与normal，并不能通过更新iq文件实现；但某些功能的切换却可以通过iq参数的不同配置来实现，如日、夜切换可完全通过iq配置来实现切换。
- 切换iq时，iq中的配置参数将会覆盖掉用户API的设置。如AWB模块，在iq中可配置手动、自动模式，那么执行该函数后，不管当前AWB处于何种模式，最终都会被新iq中的默认配置覆盖掉。

【需求】

- 头文件：rk_aiq_user_api2_sysctl.h

- 库文件：librkaiq.so

rk_aiq_uapi2_sysctl_getCrop

【描述】
获取crop参数。

【语法】

```
XCamReturn
rk_aiq_uapi2_sysctl_getCrop(const rk_aiq_sys_ctx_t* sys_ctx, rk_aiq_rect_t
*rect);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
rect	crop参数结构体指针	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

数据类型

rk_aiq_working_mode_t

【说明】
AIQ pipeline工作模式

【定义】

```
typedef enum {
    RK_AIQ_WORKING_MODE_NORMAL,
    RK_AIQ_WORKING_MODE_ISP_HDR2     = 0x10,
    RK_AIQ_WORKING_MODE_ISP_HDR3     = 0x20,
} rk_aiq_working_mode_t;
```

【成员】

成员名称	描述
RK_AIQ_WORKING_MODE_NORMAL	普通模式
RK_AIQ_WORKING_MODE_ISP_HDR2	两帧HDR模式
RK_AIQ_WORKING_MODE_ISP_HDR3	三帧HDR模式

【注意事项】

- 需要先查询sensor及AIQ所支持的模式，若设置的模式不支持则设置无效。

rk_aiq_static_info_t

【说明】
AIQ 静态信息

【定义】

```
typedef struct {
    rk_aiq_sensor_info_t    sensor_info;
    rk_aiq_lens_info_t      lens_info;
    bool has_lens_vcm;
    bool has_fl;
    bool fl_strth_adj_sup;
    bool has_irc;
    bool fl_ir_strth_adj_sup;
} rk_aiq_static_info_t;
```

【成员】

成员名称	描述
sensor_info	sensor的名称、支持的分辨率等描述
lens_info	镜头信息
has_lens_vcm	是否带vcm
has_fl	是否带闪光灯
fl_strth_adj_sup	带闪光灯是否可调
bool has_irc	是否带IR-CUT
bool fl_ir_strth_adj_sup	

rk_aiq_sensor_info_t

【说明】
sensor信息

【定义】

```
typedef struct {
    char sensor_name[32];
    rk_frame_fmt_t  support_fmt[SUPPORT_FMT_MAX];
    int32_t num;
    /* binded pp stream media index */
    int8_t binded_strm_media_idx;
} rk_aiq_sensor_info_t;
```

【成员】

成员名称	描述
sensor_name	sensor的名称
support_fmt	支持的格式
num	支持的格式个数
has_fl	是否带闪光灯
binded_strm_media_idx	该sensor挂载的media节点号

rk_aiq_module_id_t

【说明】
AIQ 模块ID

【定义】

```
typedef enum {
    RK_MODULE_INVAL = 0,
    RK_MODULE_DPCC,
    RK_MODULE_BLS,
    RK_MODULE_LSC,
    RK_MODULE_AWB_GAIN,
    RK_MODULE_CTK,
    RK_MODULE_GOC,
    RK_MODULE_SHARP,
    RK_MODULE_AE,
    RK_MODULE_AWB,
    RK_MODULE_NR,
    RK_MODULE_GIC,
    RK_MODULE_3DLUT,
    RK_MODULE_LDCH,
    RK_MODULE_TNR,
    RK_MODULE_FEC,
    RK_MODULE_MAX
}rk_aiq_module_id_t;
```

【成员】

成员名称	描述
RK_MODULE_DPCC	坏点检测与纠正
RK_MODULE_BLS	黑电平
RK_MODULE_LSC	镜头阴影校正
RK_MODULE_AWB_GAIN	白平衡增益
RK_MODULE_CTK	颜色校正
RK_MODULE_GOC	伽玛
RK_MODULE_SHARP	锐化
RK_MODULE_AE	曝光
RK_MODULE_AWB	白平衡
RK_MODULE_NR	去噪
RK_MODULE_GIC	绿平衡
RK_MODULE_3DLUT	3DLUT
RK_MODULE_LDCH	LDCH
RK_MODULE_TNR	3D去噪
RK_MODULE_FEC	鱼眼校正

rk_aiq_cpsl_cfg_t

【说明】

补光灯设置信息结构体

【定义】

```
typedef struct rk_aiq_cpsl_cfg_s {
    RKAiqOPMode_t mode;
    rk_aiq_cpsls_t lght_src;
    bool gray_on; /*!< force to gray if light on */
    union {
        struct {
            float sensitivity; /*!< Range [0-100] */
            uint32_t sw_interval; /*!< switch interval time, unit  seconds */
        } a; /*< auto mode */
        struct {
            uint8_t on; /*!< disable 0, enable 1 */
            float strength_led; /*!< Range [0-100] */
            float strength_ir; /*!< Range [0-100] */
        } m; /*!< manual mode */
    } u;
} rk_aiq_cpsl_cfg_t;
```

【成员】

成员名称	描述
mode	工作模式
lght_src	光源类型
gray_on	切换为夜晚模式后是否将画面切为黑白
sensitivity	自动模式下的切换灵敏度，范围[0,100]
sw_interval	自动模式下的切换间隔，单位秒
on	手动模式下是否切换为夜晚模式
strength_led	手动模式下的LED灯强度，范围[0,100]
strength_ir	手动模式下的红外灯强度，范围[0,100]

rk_aiq_cpsl_info_t

【说明】
补光灯查询信息结构体

【定义】

```
typedef struct rk_aiq_cpsl_info_s {
    int32_t mode;
    uint8_t on;
    bool gray;
    float strength_led;
    float strength_ir;
    float sensitivity;
    uint32_t sw_interval;
    int32_t lght_src;
} rk_aiq_cpsl_info_t;
```

【成员】

成员名称	描述
mode	工作模式
lght_src	光源类型
gray	切换为夜晚模式后是否将画面切为黑白
sensitivity	自动模式下的切换灵敏度，范围[0,100]
sw_interval	自动模式下的切换间隔，单位秒
on	手动模式下是否切换为夜晚模式
strength_led	手动模式下的LED灯强度，范围[0,100]
strength_ir	手动模式下的红外灯强度，范围[0,100]

rk_aiq_cpsl_cap_t

【说明】
补光灯支持能力结构体

【定义】

```
typedef struct rk_aiq_cpsl_cap_s {
    int32_t supported_modes[RK_AIQ_OP_MODE_MAX];
    uint8_t modes_num;
    int32_t supported_lght_src[RK_AIQ_CPSLS_MAX];
    uint8_t lght_src_num;
    rk_aiq_range_t strength_led;
    rk_aiq_range_t sensitivity;
    rk_aiq_range_t strength_ir;
} rk_aiq_cpsl_cap_t;
```

【成员】

成员名称	描述
supported_modes	支持的工作模式
modes_num	支持的模式个数
gray	切换为夜晚模式后是否将画面切为黑白
supported_lght_src	支持的光源
lght_src_num	支持的光源个数
strength_led	LED的强度范围
sensitivity	灵敏度范围
strength_ir	红外灯的强度范围

rk_aiq_rect_t

【说明】
定义crop参数结构体

【定义】

```
typedef struct rk_aiq_rect_s {
    int left;
    int top;
    int width;
    int height;
} rk_aiq_rect_t;
```

【成员】

成员名称	描述
left	horizontal output offset
top	vertical output offset
width	horizontal output size
height	vertical output size

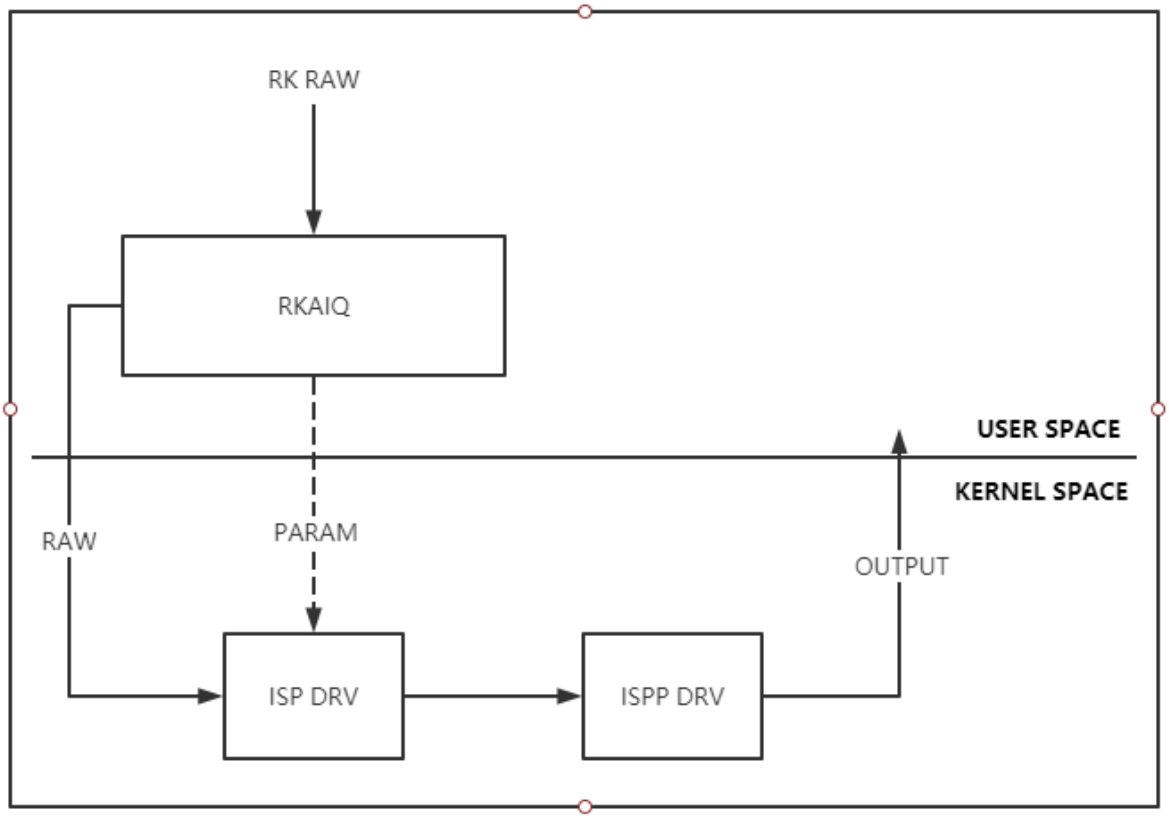
离线帧处理

概述

RKAIQ提供离线RAW帧处理功能，即RK自定义的RAW格式文件经RKAIQ解析后送ISP处理，输出为可显示正常效果的图像的功能。

注意：离线帧处理功能还未测试通过。

功能框图



离线帧处理框图

功能描述

- 支持RK-RAW文件输入。
  使用文件输入接口，调用进程将被阻塞，直到文件处理完成并成功输出。

- 支持RK-RAW buffer输入，异步处理模式。
  使用buffer输入接口，调用进程不会阻塞，buffer处理完成后将调用回调函数(如有注册回调函数)。
- 支持RK-RAW buffer输入，同步处理模式。
  使用buffer输入接口，调用进程将被阻塞，直到buffer处理完成并成功输出。

RK-RAW格式说明

参见《RK RAW文件格式》说明文档

支持的RAW格式

支持raw8/raw10/raw12，支持BGGR/GBRG/GRBG/RGGB四种bayer格式。

API参考

数据结构

注意事项

- 使用RK Raw数据处理功能，在创建AIQ Context时，rk_aiq_uapi2_sysctl_init接口的参数sns_ent_name须为“FakeCamera”。
- Raw数据的处理依赖IQ XML效果文件，XML文件生成时的分辨率应与传入的RK Raw帧数据的分辨率一致。效果文件须命名为FakeCamera.xml，放置于XML文件的加载路径下。

参考示例

离线帧处理API的使用方法请参考rkisp_demo，路径为
YOUR_SDK_DIR/external/camera_engine_rkaiq/rkisp_demo。

Camera组

概述

如环视等应用场景，需要将多个Camera归为一组做统一管理，曝光等参数需要统一设置等。Camera组相关API，可通过创建组，将多个Camera归类为同一组。
Camera组API基于单Camera的AIQ API实现，如无特殊说明，适用于单个AIQ的API也同样适用于Camera组，AIQ内部实现会根据传递的Context类型将请求分发给Camera组或者单Camera。

rk_aiq_uapi2_camgroup_create

【描述】
创建Camera组，会为每个Camera创建独立的 AIQ ctx，然后交给 Group Ctx 管理

【语法】

```
rk_aiq_camgroup_ctx_t*
rk_aiq_uapi2_camgroup_create (rk_aiq_camgroup_instance_cfg_t* cfg);
```

【参数】

参数名称	描述	输入/输出
cfg	配置参数	输入

【返回值】

返回值	描述
rk_aiq_camgroup_ctx_t*	组运行环境
NULL	失败

【需求】

- 头文件：rk_aiq_user_api2_camgroup.h
- 库文件：librkaiq.so

【注意】

- 与 rk_aiq_uapi2_sysctl_init 相区分，rk_aiq_uapi2_sysctl_init 创建单camera的 AIQ 运行环境。使用 rk_aiq_uapi2_sysctl_init 创建运行环境的sensor entity，不得再用 rk_aiq_uapi2_camgroup_create 创建组运行环境，反之亦然。
- 由于默认选择的IQ场景是 normal-day，如果需要为HDR或者其他模式，需要使用 rk_aiq_uapi2_sysctl_preInit_scene 进行预先设置

rk_aiq_uapi2_camgroup_prepare

【描述】
根据选择的pipeline模式，生成初始化参数以及建立pipeline。会调用每个Camera对应AIQ ctx的 rk_aiq_uapi2_sysctl_prepare。

【语法】

```
XCamReturn
rk_aiq_uapi2_camgroup_prepare (rk_aiq_camgroup_ctx_t* camgroup_ctx,

rk_aiq_working_mode_t mode);
```

【参数】

参数名称	描述	输入/输出
camgroup_ctx	组运行环境	输入
mode	pipeline工作模式	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：rk_aiq_user_api2_camgroup.h
- 库文件：librkaiq.so

rk_aiq_uapi2_camgroup_start

【描述】
运行Camera组。会调用每个Camera对应AIQ ctx的 rk_aiq_uapi2_sysctl_start。

【语法】

```
XCamReturn
rk_aiq_uapi2_camgroup_start (rk_aiq_camgroup_ctx_t* camgroup_ctx);
```

【参数】

参数名称	描述	输入/输出
camgroup_ctx	组运行环境	输入

【返回值】

返回值	描述
0	成功
非0	失败，参见错误码

【需求】

- 头文件：rk_aiq_user_api2_camgroup.h
- 库文件：librkaiq.so

rk_aiq_uapi2_camgroup_stop

【描述】
停止Camera组。会调用每个Camera对应AIQ ctx的 rk_aiq_uapi2_sysctl_stop。

【语法】

```
XCamReturn
rk_aiq_uapi2_camgroup_stop (rk_aiq_camgroup_ctx_t* camgroup_ctx);
```

【参数】

参数名称	描述	输入/输出
camgroup_ctx	组运行环境	输入

【返回值】

返回值	描述
0	成功
非0	失败，参见错误码

【需求】

- 头文件：rk_aiq_user_api2_camgroup.h
- 库文件：librkaiq.so

rk_aiq_uapi2_camgroup_destroy

【描述】
销毁Camera组。会调用每个Camera对应AIQ ctx的 rk_aiq_uapi2_sysctl_deinit。

【语法】

```
XCamReturn
rk_aiq_uapi2_camgroup_destroy (rk_aiq_camgroup_ctx_t* camgroup_ctx);
```

【参数】

参数名称	描述	输入/输出
camgroup_ctx	组运行环境	输入

【返回值】

返回值	描述
0	成功
非0	失败，参见错误码

【需求】

- 头文件：rk_aiq_user_api2_camgroup.h
- 库文件：librkaiq.so

rk_aiq_uapi2_camgroup_getOverlapMap

【描述】
环视应用场景下，从CameraGroup上下文获取各camera的重叠区域信息。

【语法】

```
struct RK_PS_SrcOverlapMap*
rk_aiq_uapi2_camgroup_getOverlapMap (rk_aiq_camgroup_ctx_t* camgroup_ctx);
```

【参数】

参数名称	描述	输入/输出
camgroup_ctx	组运行环境	输入

【返回值】

返回值	描述
struct RK_PS_SrcOverlapMap*	重叠区域信息结构体指针
NULL	失败

【需求】

- 头文件：rk_aiq_user_api2_camgroup.h
- 库文件：librkaiq.so

rk_aiq_uapi2_camgroup_getOverlapMap

【描述】
环视应用场景下，从文件获取各camera的重叠区域信息。

【语法】

```
XCamReturn
rk_aiq_uapi2_camgroup_getOverlapMap_from_file (const char* map_file,
                                               struct
RK_PS_SrcOverlapMap** overlapMap);
```

【参数】

参数名称	描述	输入/输出
map_file	存储重叠信息的文件	输入
overlapMap	存储获取到的重叠信息	输入，输出

【返回值】

返回值	描述
0	成功
非0	失败，参见错误码

【需求】

- 头文件：rk_aiq_user_api2_camgroup.h
- 库文件：librkaiq.so

rk_aiq_uapi2_camgroup_getAiqCtxBySnsNm

【描述】
通过组内 Sensor 实体名获取对应的单个Camera的AIQ上下文。

【语法】

```
rk_aiq_sys_ctx_t*
rk_aiq_uapi2_camgroup_getAiqCtxBySnsNm (rk_aiq_camgroup_ctx_t* camgroup_ctx,

const char* sns_entity_name);
```

【参数】

参数名称	描述	输入/输出
camgroup_ctx	组运行环境	输入
sns_entity_name	组内sensor实体名称	输入

【返回值】

返回值	描述
rk_aiq_sys_ctx_t*	获取到的AIQ运行环境
NULL	失败

【需求】

- 头文件：rk_aiq_user_api2_camgroup.h
- 库文件：librkaiq.so

rk_aiq_uapi2_camgroup_getCamInfos

【描述】
获取组内camera的信息，如Sensor实体名称等。

【语法】

```
XCamReturn
rk_aiq_uapi2_camgroup_getCamInfos (rk_aiq_camgroup_ctx_t* camgroup_ctx,
                   rk_aiq_camgroup_camInfos_t* camInfos);
```

【参数】

参数名称	描述	输入/输出
camgroup_ctx	组运行环境	输入
camInfos	获取到的组内camera信息	输出

【返回值】

返回值	描述
0	成功
非0	失败，参见错误码

【需求】

- 头文件：rk_aiq_user_api2_camgroup.h
- 库文件：librkaiq.so

rk_aiq_uapi2_camgroup_bind

【描述】
预留接口，未实现。

rk_aiq_uapi2_camgroup_unbind

【描述】
预留接口，未实现。

数据结构

rk_aiq_camgroup_instance_cfg_t

【说明】
Camera Group 配置参数

【定义】

```
typedef struct rk_aiq_camgroup_instance_cfg_s {
    const char* sns_ent_nm_array[RK_AIQ_CAM_GROUP_MAX_CAMS];
    int sns_num;
    const char* config_file_dir;

     /* followings are relative path to config_file_dir */
     const char* single_iq_file;
     const char* group_iq_file;
     const char* overlap_map_file;
 } rk_aiq_camgroup_instance_cfg_t;
```

【成员】

成员名称	描述
sns_ent_nm_array	需要加入组管理的Sensor实体 数组
sns_num	需要加入组管理的Sensor实体个数
config_file_dir	指定IQ文件路径
single_iq_file	指定IQ文件名称，可为NULL，NULL时由AIQ自动选择
group_iq_file	指定Camera组IQ配置文件，目前暂不使用
overlap_map_file	指定Camera重叠信息文件 ，可为NULL

rk_aiq_camgroup_camInfos_t

【说明】
Camera 组内各Camera信息

【定义】

```
 typedef struct rk_aiq_camgroup_camInfos_s {
     int valid_sns_num;
     const char* sns_ent_nm[RK_AIQ_CAM_GROUP_MAX_CAMS];
     int sns_camPhyId[RK_AIQ_CAM_GROUP_MAX_CAMS];
} rk_aiq_camgroup_camInfos_t;
```

【成员】

成员名称	描述
valid_sns_num	组内包含的Sensor实体数量
sns_ent_nm	组内包含的Sensor实体名称数组
sns_camPhyId	组内包含的Sensor实体对应的物理ID数组

struct RK_PS_SrcOverlapMap

【说明】
定义Camera重叠区域信息

【定义】

```
    struct RK_PS_SrcOverlapMap
{
        char versionInfo[64];
        RK_PS_SrcOverlapPosition srcOverlapPositon[8];
        unsigned char overlapMap[15 * 15 * 8];
    };
```

【成员】

成员名称	描述
versionInfo	版本信息
srcOverlapPositon	各Camera的安装时的旋转方向，可为0,90,180,270度
overlapMap	8个camera每个统计块的重叠信息，具体如下： 每个相机的统计数据是15*15列，总共8个相机(编号0~7)，实际不足8个相机的，相应位置用0补齐。按行存储，存完8个相机的第一行后，紧接着存第二行。以此类推。 取值范围为0~255，完全不重叠则值为0，完全重叠则值为255，部分重叠根据占比进行插值 camera0 row0; camera1 row0; camera2 row0; camera3 row0; camera4 row0; camera5 row0; camera6 row0; camera7 row0; camera0 row1; camera1 row1; camera2 row1; camera3 row1; camera4 row1; camera5 row1; camera6 row1; camera7 row1; camera0 row14; camera1 row14; camera2 row14; camera3 row14; camera4 row14; camera5 row14; camera6 row14; camera7 row14;

AE

概述

AE 模块实现的功能是：根据自动测光系统获得当前图像的曝光量，再自动配置镜头光圈、 sensor 快门及增益来获得最佳的图像质量。

重要概念

- 曝光时间： sensor 积累电荷的时间，是 sensor pixel 从开始曝光到电量被读出的这
- 段时间。
- 曝光增益：对 sensor 的输出电荷的总的放大系数，一般有数字增益和模拟增益，
- 模拟增益引入的噪声会稍小，所以一般优先用模拟增益。
- 光圈：光圈是镜头中可以改变通光孔径大小的机械装置。
- 抗闪烁：由于电灯的电源工频与 sensor 的帧率不匹配而导致的画面闪烁，一般通过限定曝光时间和修改 sensor 的帧率来达到抗闪烁的效果。

功能描述

AE 模块由AE 统计信息及 AE 控制策略的算法两部分组成。

功能级API参考

rk_aiq_uapi2_setAeLock

【描述】

设置ae曝光锁定功能

【语法】

```
XCamReturn rk_aiq_uapi2_setAeLock（const rk_aiq_sys_ctx_t* ctx, bool on);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
on	锁定使能	输入

【返回值】

返回值	描述
0	成功
非0	

rk_aiq_uapi2_setExpMode

【描述】
设置曝光模式，支持设置自动曝光和手动曝光。

【语法】

```
XCamReturn rk_aiq_uapi2_setExpMode(const rk_aiq_sys_ctx_t* ctx, opMode_t mode);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
mode	曝光模式	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【注意】

- 曝光模式切为手动模式时的增益和曝光时间采用图像效果文件中定义的初始值。如果切换手动模式同时需要设置曝光值，可以使用rk_aiq_uapi2_setManualExp接口。

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getExpMode

【描述】
获取曝光模式。

【语法】

```
XCamReturn rk_aiq_uapi2_getExpMode(const rk_aiq_sys_ctx_t* ctx, opMode_t *mode);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
mode	曝光模式	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setManualExp

【描述】
使用手动曝光模式，并且设置增益和曝光时间。

【语法】

```
XCamReturn rk_aiq_uapi2_setManualExp(const rk_aiq_sys_ctx_t* ctx, float gain,
float time);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
gain	曝光增益	输入
time	曝光时间	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setExpGainRange

【描述】
设置增益范围。

【语法】

```
XCamReturn rk_aiq_uapi2_setExpGainRange(const rk_aiq_sys_ctx_t* ctx, paRange_t
*gain);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
gain	曝光增益范围	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getExpGainRange

【描述】
获取增益范围。

【语法】

```
XCamReturn rk_aiq_uapi2_getExpGainRange(const rk_aiq_sys_ctx_t* ctx, paRange_t *gain);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
gain	曝光增益范围	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setExpTimeRange

【描述】
设置曝光时间范围。

【语法】

```
XCamReturn rk_aiq_uapi2_setExpTimeRange(const rk_aiq_sys_ctx_t* ctx, paRange_t *time);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
time	曝光时间范围	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getExpTimeRange

【描述】
获取曝光时间范围。

【语法】

```
XCamReturn rk_aiq_uapi2_getExpTimeRange(const rk_aiq_sys_ctx_t* ctx, paRange_t *time);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
time	曝光时间范围	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setBLCMode

【描述】
背光补偿开关、区域设置。

【语法】

```
XCamReturn rk_aiq_uapi2_setBLCMode(const rk_aiq_sys_ctx_t* ctx, bool on, aeMeasAreaType_t areaType);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
on	开关	输入
areaType	补偿区域选择	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【注意】

- 该接口仅在线性模式下可用。

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setBLCStrength

【描述】
设置暗区提升强度。

【语法】

```
XCamReturn rk_aiq_uapi2_setBLCStrength(const rk_aiq_sys_ctx_t* ctx, int
strength);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
strength	提升强度，范围[1,100]	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【注意】

- 该接口仅在线性模式下可用。

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h

- 库文件：librkaiq.so

rk_aiq_uapi2_setHLCMode

【描述】
强光抑制开关。

【语法】

```
XCamReturn rk_aiq_uapi2_setHLCMode(const rk_aiq_sys_ctx_t* ctx, bool on);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
on	开关	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【注意】

- 该接口仅在线性模式下可用。

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setHLCStrength

【描述】
设置强光抑制强度。

【语法】

```
XCamReturn rk_aiq_uapi2_setHLCStrength(const rk_aiq_sys_ctx_t* ctx, int
strength);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
strength	抑制强度，范围[1,100]	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【注意】

- 该接口仅在线性模式下可用。

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setAntiFlickerEn

【描述】
设置抗工频闪烁开关

【语法】

```
XCamReturn rk_aiq_uapi2_setAntiFlickerEn(const rk_aiq_sys_ctx_t* ctx, bool on);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
on	功能开关参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getAntiFlickerEn

【描述】
设置抗工频闪烁开关

【语法】

```
XCamReturn rk_aiq_uapi2_getAntiFlickerEn(const rk_aiq_sys_ctx_t* ctx, bool*
on);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
on	功能开关参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setAntiFlickerMode

【描述】
设置抗闪模式。

【语法】

```
XCamReturn rk_aiq_uapi2_setAntiFlickerMode(const rk_aiq_sys_ctx_t* ctx,
antiFlickerMode_t mode);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
mode	抗闪模式	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getAntiFlickerMode

【描述】
获取抗闪模式。

【语法】

```
XCamReturn rk_aiq_uapi2_getAntiFlickerMode(const rk_aiq_sys_ctx_t* ctx,
antiFlickerMode_t *mode);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
mode	抗闪模式	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setExpPwrLineFreqMode

【描述】
设置抗闪频率。

【语法】

```
XCamReturn rk_aiq_uapi2_setExpPwrLineFreqMode(const rk_aiq_sys_ctx_t* ctx,
expPwrLineFreq_t freq);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
freq	抗闪频率	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getExpPwrLineFreqMode

【描述】
获取抗闪频率。

【语法】

```
XCamReturn rk_aiq_uapi2_getExpPwrLineFreqMode(const rk_aiq_sys_ctx_t* ctx,
expPwrLineFreq_t *freq);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
freq	抗闪频率	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

功能级API数据类型

opMode_t

【说明】
定义自动手动模式

【定义】

```
typedef enum opMode_e {
    OP_AUTO = 0,
    OP_MANUALl = 1,
    OP_INVAL
} opMode_t;
```

【成员】

成员名称	描述
OP_AUTO	自动模式
OP_MANUALl	手动模式
OP_INVAL	无效值

paRange_t

【说明】
定义参数范围

【定义】

```
typedef struct paRange_s {
    float max;
    float min;
} paRange_t;
```

【成员】

成员名称	描述
max	上限值
min	下限值

aeMeasAreaType_t

【说明】
定义AE测量区域类型

【定义】

```
typedef enum aeMeasAreaType_e {
    AE_MEAS_AREA_AUTO = 0,
    AE_MEAS_AREA_UP,
    AE_MEAS_AREA_BOTTOM,
    AE_MEAS_AREA_LEFT,
    AE_MEAS_AREA_RIGHT,
    AE_MEAS_AREA_CENTER,
} aeMeasAreaType_t;
```

【成员】

成员名称	描述
AE_MEAS_AREA_AUTO	自动
AE_MEAS_AREA_UP	上方区域
AE_MEAS_AREA_BOTTOM	下方区域
AE_MEAS_AREA_LEFT	左边区域
AE_MEAS_AREA_RIGHT	右边区域
AE_MEAS_AREA_CENTER	中心区域

expPwrLineFreq_t

【说明】
定义抗闪频率

【定义】

```
typedef enum expPwrLineFreq_e {
        EXP_PWR_LINE_FREQ_DIS    = 0,
        EXP_PWR_LINE_FREQ_50HZ   = 1,
        EXP_PWR_LINE_FREQ_60HZ   = 2,
} expPwrLineFreq_t;
```

【成员】

成员名称	描述
EXP_PWR_LINE_FREQ_DIS	
EXP_PWR_LINE_FREQ_50HZ	50赫兹
EXP_PWR_LINE_FREQ_60HZ	60赫兹

antiFlickerMode_t

【说明】
定义抗闪模式

【定义】

```
typedef enum antiFlickerMode_e {
    ANTIFLICKER_NORMAL_MODE = 0,
    ANTIFLICKER_AUTO_MODE = 1,
} antiFlickerMode_t;
```

【成员】

成员名称	描述
ANTIFLICKER_NORMAL_MODE	普通模式
ANTIFLICKER_AUTO_MODE	自动选择模式

模块级API参考

rk_aiq_user_api2_ae_setExpSwAttr

【描述】
设定 AE曝光软件属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_setExpSwAttr(const rk_aiq_sys_ctx_t* ctx,
                                 const Uapi_ExpSwAttrV2_t expSwAttr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
expSwAttr	AE公共功能控制参数结构体	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

【举例】

- 设置手动曝光属性

曝光分量包括sensor曝光时间、sensor曝光增益、isp数字增益。设置手动曝光模式之后，还需要分别设置各曝光分量的手动状态（ManualGainEn、ManualTimeEn、ManualIspDgainEn）及其对应手动值。手动曝光模式下，要求所有曝光分量为手动状态，设置的各曝光分量的值，会受到sensor及镜头的限制。如设置的曝光分量值超过sensor的限制，算法内部会自动进行校正。

```
Uapi_ExpSwAttrV2_t expSwAttr;
ret = rk_aiq_user_api2_ae_getExpSwAttr(ctx, &expSwAttr);
expSwAttr.AecOpType = RK_AIQ_OP_MODE_MANUAL;
//LinearAE
expSwAttr.stManual.LinearAE.ManualGainEn = true;
expSwAttr.stManual.LinearAE.ManualTimeEn = true;
expSwAttr.stManual.LinearAE.GainValue = 1.0f; /*gain = 1x*/
expSwAttr.stManual.LinearAE.TimeValue = 0.02f; /*time = 1/50s*/

//HdrAE (should set all frames)
expSwAttr.stManual.HdrAE.ManualGainEn = true;
expSwAttr.stManual.HdrAE.ManualTimeEn = true;
expSwAttr.stManual.HdrAE.GainValue[0] = 1.0f; /*sframe gain = 1x*/
expSwAttr.stManual.HdrAE.TimeValue[0] = 0.002f; /*sframe time = 1/500s*/
expSwAttr.stManual.HdrAE.GainValue[1] = 2.0f; /*mframe gain = 2x*/
expSwAttr.stManual.HdrAE.TimeValue[1] = 0.01f; /*mframe time = 1/100s*/
expSwAttr.stManual.HdrAE.GainValue[2] = 4.0f; /*lframe gain = 4x*/
expSwAttr.stManual.HdrAE.TimeValue[2] = 0.02f; /*lframe time = 1/50s*/

ret = rk_aiq_user_api2_ae_setExpSwAttr(ctx, expSwAttr);
```

- 设置自动曝光属性

自动曝光可以设置曝光分量的作用范围，若设置的曝光分量范围超过sensor的限制，算法内部会自动进行校正。

【注】设置曝光分量range的参数与获取曝光分量range的参数不同

```
Uapi_ExpSwAttrV2_t expSwAttr;
```

```
ret = rk_aiq_user_api2_ae_getExpSwAttr(ctx, &expSwAttr);
expSwAttr.AecOpType = RK_AIQ_OP_MODE_AUTO;

//set time range in struct "stAdvanced"
expSwAttr.stAdvanced.SetAeRangeEn = true;/*must enable*/
//LinAE
expSwAttr.stAdvanced.SetLinAeRange.stExpTimeRange.Max = 0.04f; /*time_max =
0.04*/
expSwAttr.stAdvanced.SetLinAeRange.stExpTimeRange.Min = 0.001f; /*time_min =
0.001*/
//HdrAE
expSwAttr.stAdvanced.SetHdrAeRange.stExpTimeRange[0].Max = 0.002f; /*sframe
time_max = 0.02*/
expSwAttr.stAdvanced.SetHdrAeRange.stExpTimeRange[0].Min = 0.001f; /*sframe
time_min = 0.01*/
expSwAttr.stAdvanced.SetHdrAeRange.stExpTimeRange[1].Max = 0.003f; /*mframe
time_max = 0.03*/
expSwAttr.stAdvanced.SetHdrAeRange.stExpTimeRange[1].Min = 0.002f; /*mframe
time_min = 0.02*/
expSwAttr.stAdvanced.SetHdrAeRange.stExpTimeRange[2].Max = 0.04f; /*lframe
time_max = 0.03*/
expSwAttr.stAdvanced.SetHdrAeRange.stExpTimeRange[2].Min = 0.03f; /*lframe
time_min = 0.02*/

//get time range in struct "stAuto"
printf("linear time range=[%f,%f]\n",
    expSwAttr.stAuto.LinAeRange.stExpTimeRange.Min,
expSwAttr.stAuto.LinAeRange.stExpTimeRange.Max);
printf("hdr stime range=[%f,%f], mtime range=[%f,%f], ltime range=[%f,%f]\n",
    expSwAttr.stAuto.HdrAeRange.stExpTimeRange[0].Min,
expSwAttr.stAuto.HdrAeRange.stExpTimeRange[0].Max,
    expSwAttr.stAuto.HdrAeRange.stExpTimeRange[1].Min,
expSwAttr.stAuto.HdrAeRange.stExpTimeRange[1].Max,
    expSwAttr.stAuto.HdrAeRange.stExpTimeRange[2].Min,
expSwAttr.stAuto.HdrAeRange.stExpTimeRange[2].Max);

//set gain range in struct "stAdvanced"
expSwAttr.stAdvanced.SetAeRangeEn = true;/*must enable*/
//LinAE
expSwAttr.stAdvanced.SetLinAeRange.stGainRange.Max = 32.0f; /*gain_max = 32x*/
expSwAttr.stAdvanced.SetLinAeRange.stGainRange.Min = 1.0f; /*gain_min = 1x*/
//HdrAE
expSwAttr.stAdvanced.SetHdrAeRange.stGainRange[0].Max = 32.0f; /*sframe gain_max
= 2x*/
expSwAttr.stAdvanced.SetHdrAeRange.stGainRange[0].Min = 1.0f; /*sframe gain_min
= 1x*/
expSwAttr.stAdvanced.SetHdrAeRange.stGainRange[1].Max = 64.0f; /*mframe gain_max
= 64x*/
expSwAttr.stAdvanced.SetHdrAeRange.stGainRange[1].Min = 1.0f; /*mframe gian_min
= 1x*/
expSwAttr.stAdvanced.SetHdrAeRange.stGainRange[2].Max = 64.0f; /*lframe gain_max
= 64x*/
expSwAttr.stAdvanced.SetHdrAeRange.stGainRange[2].Min = 1.0f; /*lframe gain_min
= 1x*/

//get gain range in struct "stAuto"
printf("linear gain range=[%f,%f]\n",
```

```
    expSwAttr.stAuto.LinAeRange.stGainRange.Min,
expSwAttr.stAuto.LinAeRange.stGainRange.Max);
printf("hdr sgain range=[%f,%f], mgain range=[%f,%f], lgain range=[%f,%f]\n",
    expSwAttr.stAuto.HdrAeRange.stGainRange[0].Min,
expSwAttr.stAuto.HdrAeRange.stGainRange[0].Max,
    expSwAttr.stAuto.HdrAeRange.stGainRange[1].Min,
expSwAttr.stAuto.HdrAeRange.stGainRange[1].Max,
    expSwAttr.stAuto.HdrAeRange.stGainRange[2].Min,
expSwAttr.stAuto.HdrAeRange.stGainRange[2].Max);

ret = rk_aiq_user_api2_ae_setExpSwAttr(ctx, expSwAttr);
```

- 设置半手动曝光属性

半手动曝光是曝光分量（sensor曝光时间、sensor曝光增益、isp数字增益）中至少有一个曝光分量为手动状态，其他曝光分量为自动状态，否则将报错退出。例如，为了实现曝光增益手动，曝光时间、isp数字增益自动的半自动曝光功能，有以下两种实现方式：方式一，在手动曝光模式中，将曝光时间、isp数字增益的手动使能设为false，设置曝光增益的手动使能为true，并设置对应手动值；方式二，在自动曝光模式下，将曝光增益的最大最小值均设为需要固定的值。

【注】方式一：HDR曝光模式下，所有帧的手动行为是同步的（即各帧的对应曝光分量手动行为一致），同时需保证长帧最大曝光大于短帧最大曝光，长帧最小曝光大于短帧最小曝光；方式二：HDR曝光模式下，各帧的手动行为允许不一致（如短帧增益手动，长帧增益可为自动），同时需保证长帧最大曝光大于短帧最大曝光，长帧最小曝光大于短帧最小曝光。

```
//Method One
Uapi_ExpSwAttrV2_t expSwAttr;
ret = rk_aiq_user_api2_ae_getExpSwAttr(ctx, &expSwAttr);
expSwAttr.AecOpType = RK_AIQ_OP_MODE_MANUAL;
//LinearAE
expSwAttr.stManual.LinearAE.ManualGainEn = true;
expSwAttr.stManual.LinearAE.ManualTimeEn = false;
expSwAttr.stManual.LinearAE.ManualIspDgainEn = false;
expSwAttr.stManual.LinearAE.GainValue = 2.0f; /*gain = 2x*/
//HdrAE (need to set all frames)
expSwAttr.stManual.HdrAE.ManualGainEn = true;
expSwAttr.stManual.HdrAE.ManualTimeEn = false;
expSwAttr.stManual.HdrAE.ManualIspDgainEn = false;
expSwAttr.stManual.HdrAE.GainValue[0] = 1.0f; /*sframe gain = 1x*/
expSwAttr.stManual.HdrAE.GainValue[1] = 2.0f; /*mframe gain = 2x*/
expSwAttr.stManual.HdrAE.GainValue[2] = 4.0f; /*lframe gain = 4x*/
ret = rk_aiq_user_api2_ae_setExpSwAttr(ctx, expSwAttr);

//Method Two
Uapi_ExpSwAttrV2_t expSwAttr;
ret = rk_aiq_user_api2_ae_getExpSwAttr(ctx, &expSwAttr);
expSwAttr.AecOpType = RK_AIQ_OP_MODE_AUTO;
//set gain range
expSwAttr.stAdvanced.SetAeRangeEn = true;/*必须使能*/
//LinAE
expSwAttr.stAdvanced.SetLinAeRange.stGainRange.Max = 2.0f; /*gain_max = 2x*/
expSwAttr.stAdvanced.SetLinAeRange.stGainRange.Min = 2.0f;  /*gain_min = 2x*/
//HdrAE (allow to set only one frame)
expSwAttr.stAdvanced.SetHdrAeRange.stGainRange[0].Max = 2.0f; /*sframe gain_max
= 2x*/
expSwAttr.stAdvanced.SetHdrAeRange.stGainRange[0].Min = 2.0f;  /*sframe gain_min
= 2x*/
```

```
ret = rk_aiq_user_api2_ae_setExpSwAttr(ctx, expSwAttr);
```

- 设置固定帧率或自动降帧

固定帧率模式下，允许设置的帧率不得超过驱动定义的最大帧率，如超过会有log提醒，且帧率设置失效。

```
//set fixed framemode
Uapi_ExpSwAttrV2_t expSwAttr;
ret = rk_aiq_user_api2_ae_getExpSwAttr(ctx, &expSwAttr);
expSwAttr.stAuto.stFrmRate.isFpsFix = true;
expSwAttr.stAuto.stFrmRate.FpsValue = 25; /*fps = 25*/
ret = rk_aiq_user_api2_ae_setExpSwAttr(ctx, expSwAttr);

//set auto framemode
Uapi_ExpSwAttrV2_t expSwAttr;
ret = rk_aiq_user_api2_ae_getExpSwAttr(ctx, &expSwAttr);
expSwAttr.stAuto.stFrmRate.isFpsFix = false;
/*一般自动降帧模式由tuning人员事先配置好最低帧率和切换帧率对应的gain值*/
ret = rk_aiq_user_api2_ae_setExpSwAttr(ctx, expSwAttr);
```

- 设置曝光调节速度及延迟帧数

```
Uapi_ExpSwAttrV2_t expSwAttr;
ret = rk_aiq_user_api2_ae_getExpSwAttr(ctx, &expSwAttr);
//set ae speed
expSwAttr.stAuto.stAeSpeed.DampOver = 0.8f;
expSwAttr.stAuto.stAeSpeed.DampDark2Bright = 0.8f;
expSwAttr.stAuto.stAeSpeed.DampUnder = 0.8f;
expSwAttr.stAuto.stAeSpeed.DampBright2Dark = 0.8f;
//set ae delay
expSwAttr.stAuto.BlackDelayFrame = 2;
expSwAttr.stAuto.WhiteDelayFrame = 4;
ret = rk_aiq_user_api2_ae_setExpSwAttr(ctx, expSwAttr);
```

- 设置抗闪功能

```
Uapi_ExpSwAttr_t expSwAttr;
ret = rk_aiq_user_api2_ae_getExpSwAttr(ctx, &expSwAttr);
//set antifilicker mode
expSwAttr.stAuto.stAntiFlicker.enable = true;
expSwAttr.stAuto.stAntiFlicker.Frequency = AECV2_FLICKER_FREQUENCY_50HZ;
expSwAttr.stAuto.stAntiFlicker.Mode = AECV2_ANTIFLICKER_AUTO_MODE;
ret = rk_aiq_user_api2_ae_setExpSwAttr(ctx, expSwAttr);
```

- 设置AE权重

设置15X15权重，算法内部根据硬件实际分块规格，进行权重的压缩。目前有两种设置权重的方式：方式一直接修改json中的权重；方式二：通过stadvanced修改权重，若消除使能，即可还原回json中的权重（推荐使用这种方式）。

针对人脸应用建议使用方式二，出现人脸时expSwAttr.stAdvanced.enable = true，使用expSwAttr.stAdvanced结构体中的权重，人脸消失时expSwAttr.stAdvanced.enable = false，使用原权重。

```

```

```
Uapi_ExpSwAttr_t expSwAttr;
ret = rk_aiq_user_api2_ae_getExpSwAttr(ctx, &expSwAttr);

uint8_t GridWeights[225]={
0,  0, 1,  2,  2,  3,  4,  5,  4, 3,  2,  2,  1,  0,  0,
0,  1, 2,  3,  3,  4,  5,  6,  5, 4,  3,  3,  2,  1,  0,
1,  2, 3,  5,  5,  6,  7,  8,  7, 6,  5,  5,  3,  2,  1,
2,  3, 5,  7,  7,  8,  9, 10,  9, 8,  7,  7,  5,  3,  2,
2,  3, 5,  7,  8,  9, 10, 11, 10, 9,  8,  7,  5,  3,  2,
2,  4, 6,  8,  9, 10, 11, 12, 11,10,  9,  8,  6,  4,  2,
2,  4, 6,  9, 10, 11, 12, 13, 12,11, 10,  9,  6,  4,  2,
3,  5 ,7, 10, 11, 12, 13, 14, 13,12, 11, 10,  7,  5,  3,
2,  4, 6,  9, 10, 11, 12, 13, 12,11, 10,  9,  6,  4,  2,
2,  4, 6,  8,  9, 10, 11, 12, 11,10,  9,  8,  6,  4,  2,
2,  3, 5,  7,  8,  9, 10, 11, 10, 9,  8,  7,  5,  3,  2,
2,  3, 5,  7,  7,  8,  9, 10,  9, 8,  7,  7,  5,  3,  2,
1,  2, 4,  6,  6,  7,  8,  9,  8, 7,  6,  6,  4,  2,  1,
0,  1, 3,  5,  5,  6,  7,  8,  7, 6,  5,  5,  3,  1,  0,
0,  1, 3,  5,  5,  6,  7,  8,  7, 6,  5,  5,  3,  1,  0
};

//method one:
memcpy(expSwAttr.GridWeights.uCoeff, GridWeights,
sizeof(expSwAttr.GridWeights.uCoeff));

//method two:
expSwAttr.stAdvanced.enable = true; //important! true means preferring to use
these parameters
memcpy(expSwAttr.stAdvanced.GridWeights,GridWeights,sizeof(expSwAttr.stAdvanced.
GridWeights));

ret = rk_aiq_user_api2_ae_setExpSwAttr(ctx, expSwAttr);
```

rk_aiq_user_api2_ae_getExpSwAttr

【描述】

获取 AE 曝光软件属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_getExpSwAttr(const rk_aiq_sys_ctx_t* ctx,
                          Uapi_ExpSwAttrV2_t* pExpSwAttr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
pExpSwAttr	AE曝光软件属性结构体指针	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api2_ae_setLinAeRouteAttr

【描述】
设置线性模式下AE的场景曝光分配策略。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_setLinAeRouteAttr(const rk_aiq_sys_ctx_t* ctx, const
Uapi_LinAeRouteAttr_t linAeRouteAttr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
linAeRouteAttr	AE曝光分配策略结构体	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

【举例】

```
Uapi_LinAeRouteAttr_t LinAeRouteAttr;
memset(&LinAeRouteAttr,0x00,sizeof(Uapi_LinAeRouteAttr_t));
rk_aiq_user_api2_ae_getLinAeRouteAttr(sys_ctx,&LinAeRouteAttr);

int len = 8;
float TimeDot[8]={0,0.01,0.01,0.02,0.02,0.03,0.03,0.04};
float GainDot[8]={1,1,4,4,8,8,16,32};
float IspGainDot[8]={1,1,1,1,1,1,1,1};
int PirisDot[8]={512,512,512,512,512,512,512,512};

LinAeRouteAttr.TimeDot_len = len;
```

```
LinAeRouteAttr.GainDot_len = len;
LinAeRouteAttr.IspDGainDot_len = len;
LinAeRouteAttr.PIrisDot_len = len;

LinAeRouteAttr.GainDot = GainDot;
LinAeRouteAttr.IspDGainDot = IspGainDot;
LinAeRouteAttr.TimeDot = TimeDot;
LinAeRouteAttr.PIrisDot = PirisDot;

rk_aiq_user_api2_ae_setLinAeRouteAttr(sys_ctx,LinAeRouteAttr);
```

rk_aiq_user_api2_ae_getLinAeRouteAttr

【描述】

获取线性模式下AE的场景曝光分配策略。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_getLinAeRouteAttr(const rk_aiq_sys_ctx_t* ctx,
Uapi_LinAeRouteAttr_t* pLinAeRouteAttr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
pLinAeRouteAttr	AE曝光分配策略结构体指针	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api2_ae_setHdrAeRouteAttr

【描述】

设置HDR模式下AE的场景曝光分配策略。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_setHdrAeRouteAttr(const rk_aiq_sys_ctx_t* ctx, const
Uapi_HdrAeRouteAttr_t hdrAeRouteAttr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
hdrAeRouteAttr	AE曝光分配策略结构体	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

【举例】
Uapi_HdrAeRouteAttr_t stHdrRoute;
memset(&stHdrRoute,0x00,sizeof(Uapi_HdrAeRouteAttr_t));
ret = rk_aiq_user_api2_ae_getHdrAeRouteAttr(ctx,&stHdrRoute);

```
int len = 6;
float HdrTimeDot[3][6] = {0.0, 0.01, 0.01, 0.01, 0.01, 0.01,
                          0.0, 0.02, 0.02, 0.02, 0.02, 0.02,
                          0.0, 0.03, 0.03, 0.03, 0.03, 0.03
                         };
float HdrGainDot[3][6] = {1, 1, 4, 6, 8, 12,
                          1, 1, 4, 6, 8, 12,
                          1, 1, 4, 6, 8, 12
                         };
float HdrIspDGainDot[3][6] = {1, 1, 1, 1, 1, 1,
                              1, 1, 1, 1, 1, 1,
                              1, 1, 1, 1, 1, 1
                             };
int   HdrPIrisGainDot[6] = {1, 1, 1, 1, 1, 1};

stHdrRoute.Frm0TimeDot_len = len;
stHdrRoute.Frm0GainDot_len = len;
stHdrRoute.Frm0IspDGainDot_len = len;
stHdrRoute.Frm1TimeDot_len = len;
stHdrRoute.Frm1GainDot_len = len;
stHdrRoute.Frm1IspDGainDot_len = len;
stHdrRoute.Frm2TimeDot_len = len;
stHdrRoute.Frm2GainDot_len = len;
stHdrRoute.Frm2IspDGainDot_len = len;
stHdrRoute.PIrisDot_len = len;

stHdrRoute.Frm0TimeDot = HdrTimeDot[0];
stHdrRoute.Frm0GainDot = HdrGainDot[0];
stHdrRoute.Frm0IspDGainDot = HdrIspDGainDot[0];
stHdrRoute.Frm1TimeDot = HdrTimeDot[1];
stHdrRoute.Frm1GainDot = HdrGainDot[1];
```

```
stHdrRoute.Frm1IspDGainDot = HdrIspDGainDot[1];
stHdrRoute.Frm2TimeDot = HdrTimeDot[2];
stHdrRoute.Frm2GainDot = HdrGainDot[2];
stHdrRoute.Frm2IspDGainDot = HdrIspDGainDot[2];
stHdrRoute.PIrisDot = HdrPIrisGainDot;

ret = rk_aiq_user_api2_ae_setHdrAeRouteAttr(ctx,stHdrRoute);
```

rk_aiq_user_api2_ae_getHdrAeRouteAttr

【描述】

获取HDR模式下AE的场景曝光分配策略。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_getHdrAeRouteAttr(const rk_aiq_sys_ctx_t* ctx,
Uapi_HdrAeRouteAttr_t* pHdrAeRouteAttr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
pHdrAeRouteAttr	AE曝光分配策略结构体指针	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api2_ae_setLinExpAttr

【描述】

设置AE线性模式曝光参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_setLinExpAttr(const rk_aiq_sys_ctx_t* ctx, const
Uapi_LinExpAttrV2_t linExpAttr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
linExpAttr	AE曝光参数结构体	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

【举例】

设置AE线性模式曝光参数，包括但不局限于：设置调整亮度力度Evbias，设置背光补偿功能BackLightCtrl，设置强光抑制功能OverExpCtrl等。值得注意的是，背光补偿和强光抑制不能同时使能。

```
Uapi_LinExpAttrV2_t linExpAttr;
memset(&linExpAttr,0,sizeof(Uapi_LinExpAttrV2_t));
ret = rk_aiq_user_api2_ae_getLinExpAttr(ctx, &linExpAttr);
//set Evbias
printf("Evbias=%f\n", linExpAttr.Evbias);
linExpAttr.Evbias = 100.0f;

//set BackLightCtrl
linExpAttr.BackLightCtrl.Enable = true;
linExpAttr.BackLightCtrl.StrBias = 200.0f;

//set OverExpCtrl
linExpAttr.OverExpCtrl.Enable = true;
linExpAttr.OverExpCtrl.StrBias = 100.0f;

ret = rk_aiq_user_api2_ae_setLinExpAttr(ctx, linExpAttr);
```

rk_aiq_user_api2_ae_getLinExpAttr

【描述】
获取AE线性模式曝光参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_getLinExpAttr(const rk_aiq_sys_ctx_t* ctx,
Uapi_LinExpAttrV2_t* pLinExpAttr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
pLinExpAttr	AE曝光参数结构体指针	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api2_ae_setHdrExpAttr

【描述】
设置AE HDR模式曝光参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_setHdrExpAttr(const rk_aiq_sys_ctx_t* ctx, const
Uapi_HdrExpAttrV2_t hdrExpAttr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
hdrExpAttr	AE曝光参数结构体	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

【举例】

设置AE HDR模式曝光参数，包括但不局限于：设置曝光比为手动或自动，调整亮度力度Evbias。

```
Uapi_HdrExpAttrV2_t HdrExpAttr;
```

```
memset(&HdrExpAttr,0,sizeof(Uapi_HdrExpAttrV2_t));
ret = rk_aiq_user_api2_ae_getHdrExpAttr(ctx, &HdrExpAttr);

//set hdr mframe params
int len = 8;
float explevel[8] = {0, 0.096, 0.192, 0.384, 0.96, 1.344, 1.92, 3};
float setpoint[8] = {50, 45, 40, 35, 30, 25, 20, 15};
HdrExpAttr.HdrAEAttr.MframeCtrl.MExpLevel_len = len;
HdrExpAttr.HdrAEAttr.MframeCtrl.MSetPoint_len = len;
HdrExpAttr.HdrAEAttr.MframeCtrl.MExpLevel = explevel;
HdrExpAttr.HdrAEAttr.MframeCtrl.MSetPoint = setpoint;

//set ExpRatioType (AUTO/FIX)
HdrExpAttr.ExpRatioCtrl.ExpRatioType =
AECV2_HDR_RATIOTYPE_MODE_AUTO（AECV2_HDR_RATIOTYPE_MODE_FIX）;
//set Evbias
HdrExpAttr.Evbias = -100.0f;

ret = rk_aiq_user_api2_ae_setHdrExpAttr(ctx, HdrExpAttr);
```

rk_aiq_user_api2_ae_getHdrExpAttr

【描述】
获取AE HDR模式曝光参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_getHdrExpAttr(const rk_aiq_sys_ctx_t* ctx,
Uapi_HdrExpAttrV2_t* pHdrExpAttr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
pHdrExpAttr	AE曝光参数结构体指针	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api2_ae_setIrisAttr

【描述】
设置AE 光圈控制参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_setIrisAttr(const rk_aiq_sys_ctx_t * sys_ctx, const
Uapi_IrisAttrV2_t irisAttr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
irisAttr	光圈控制参数结构体	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

rk_aiq_user_api2_ae_getIrisAttr

【描述】
获取AE 光圈控制参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_getIrisAttr(const rk_aiq_sys_ctx_t * sys_ctx, const
Uapi_IrisAttrV2_t* irisAttr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
irisAttr	光圈控制参数结构体	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【举例】

- 设置光圈控制参数

```
Uapi_IrisAttrV2_t irisAttr;
ret = rk_aiq_user_api2_ae_getIrisAttr(ctx, &irisAttr);
```

```
irisAttr.enable = true; /*run AIris*/

//set P-iris attributes
irisAttr.IrisType = IRIS_P_TYPE;
irisAttr.PIrisAttr.TotalStep = 81;
irisAttr.PIrisAttr.EffcStep = 44;
irisAttr.PIrisAttr.ZeroIsMax = true;
uint16_t StepTable[1024] = {
512, 511, 506,499 491 483, 474, 465, 456,
446, 437, 427, 417, 408, 398, 388, 378, 368,
359, 349, 339, 329, 319, 309, 300, 290, 280,
271, 261, 252, 242, 233, 224, 214, 205, 196,
187, 178, 170, 161, 153, 144, 136, 128, 120,
112, 105, 98, 90, 83, 77, 70, 64, 58,
52, 46, 41, 36, 31, 27, 23, 19, 16,
13, 10, 8, 6, 4, 3, 1, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0};
memcpy(irisAttr.PIrisAttr.StepTable,StepTable,sizeof(irisAttr.PIrisAttr.StepTabl
e));
ret = rk_aiq_user_api2_ae_setIrisAttr(ctx, irisAttr);

//set DC-iris attributes
irisAttr.IrisType = IRIS_DC_TYPE;
irisAttr.DCIrisAttr.Kp= 0.5f;
irisAttr.DCIrisAttr.Ki= 0.2f;
irisAttr.DCIrisAttr.Kd = 0.3f;
irisAttr.DCIrisAttr.OpenPwmDuty = 40;
irisAttr.DCIrisAttr.ClosePwmDuty = 22;
irisAttr.DCIrisAttr.MinPwmDuty = 0;
irisAttr.DCIrisAttr.MaxPwmDuty = 100;
ret = rk_aiq_user_api2_ae_setIrisAttr(ctx, irisAttr);

//set manual iris with auto ae
irisAttr.IrisOpType = RK_AIQ_OP_MODE_MANUAL;
if(irisAttr.IrisType == IRIS_P_TYPE);
    irisAttr.ManualAttr.PIrisGainValue = 512; /*p-iris F#=1.4*/
if(irisAttr.IrisType == IRIS_DC_TYPE);
    irisAttr.ManualAttr.DCIrisHoldValue = 20; /*dc-iris PwmDuty=20*/
ret = rk_aiq_user_api2_ae_setIrisAttr(ctx, irisAttr);
```

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api2_ae_setExpWinAttr

【描述】
设置AE统计窗口属性

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_setExpWinAttr(const rk_aiq_sys_ctx_t* sys_ctx, const
Uapi_ExpWin_t ExpWin);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
ExpWin	窗口属性参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

【举例】

根据人脸位置修改AE统计坐标，实现针对人脸区域进行亮度统计

【注】设置统计坐标时，需保证
ExpWin.h_offs+ExpWin.h_size<=pic_width,ExpWin.v_offs+ExpWin.v_size<=pic_height

```
Uapi_ExpWin_t ExpWin;
//假设sensor分辨率为2688X1520 人脸相对于画面左上角的横向偏移为100个pixel，纵向偏移为100个pixel，人脸尺寸为边长100个pixel的正方形
ExpWin.h_offs=100;
ExpWin.v_offs=100;
ExpWin.h_size=100;
ExpWin.v_size=100;
rk_aiq_user_api2_ae_setExpWinAttr(ctx, ExpWin);
```

rk_aiq_user_api2_ae_getExpWinAttr

【描述】
获取AE统计窗口属性

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_getExpWinAttr(const rk_aiq_sys_ctx_t* sys_ctx,const
Uapi_ExpWin_t* ExpWin);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
ExpWin	窗口属性参数	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api2_ae_queryExpResInfo

【描述】

获取 AE 内部状态信息。

【语法】

```
XCamReturn
rk_aiq_user_api2_ae_queryExpResInfo(const rk_aiq_sys_ctx_t* ctx,
Uapi_ExpQueryInfo_t* pExpResInfo);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
pExpResInfo	AE内部状态信息结构体指针	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ae.h、rk_aiq_uapi2_ae_int.h
- 库文件：librkaiq.so

模块级API数据类型

Uapi_ExpSwAttr_t

【说明】

AE公共功能控制参数结构体

【定义】

```
typedef struct Uapi_ExpSwAttrV2_s {
    rk_aiq_uapi_sync_t                  sync;
    uint8_t                             Enable;
    CalibDb_CamRawStatsModeV2_t         RawStatsMode;
    CalibDb_CamHistStatsModeV2_t        HistStatsMode;
    CalibDb_CamYRangeModeV2_t           YRangeMode;
    uint8_t                             AecRunInterval;
    RKAiqOPMode_t                       AecOpType;
    Cam15x15UCharMatrix_t               GridWeights;
    Uapi_AeAttrV2_t                     stAuto;
    Uapi_MeAttrV2_t                     stManual;
    Uapi_ExpSwAttr_AdvancedV2_t         stAdvanced;
} Uapi_ExpSwAttrV2_t;
```

【成员】

成员名称	描述
sync	参考rk_aiq_uapi_sync_t定义，参见 **“概述/API说明”** 章节
Enable	Aec使能开关。1，开启Aec算法；0，关闭Aec算法，曝光保持在关闭前的值。
RawStatsMode	Aec模块亮度统计模式。共四种模式分别为：CAM_RAWSTATSV2_MODE_Y/R/G/B，默认为Y模式。
HistStatsMode	Aec模块直方图统计模式。共五种模式分别为：CAM_HISTV2_MODE_Y/R/G/B，默认为Y模式。
YRangeMode	Aec模块Y通道Range模式。共两种模式分别为CAM_YRANGEV2_MODE_FULL/LIMITED，默认为FULL模式。
AecRunInterval	Ae算法运行间隔，取值范围[0,255]，默认值为0。取值为0时，每帧运行AE；取值为1时，每隔1帧运行AE；以此类推。
AecOpType	曝光模式，分为自动曝光(RK_AIQ_OP_MODE_AUTO)模式/手动(RK_AIQ_OP_MODE_MANUAL)曝光模式。默认为AUTO模式。手动曝光模式需要与stManual一起配合，进行手动曝光值的设置。
GridWeights	窗口（直方图）权重，包含15*15个数组元素
stAuto	自动曝光参数结构体
stManual	手动曝光参数结构体
stAdvanced	优先使用参数结构体

【相关数据类型】

- rk_aiq_uapi_sync_t
- Uapi_ExpSwAttr_AdvancedV2_t
- Uapi_AeAttrV2_t
- Uapi_MeAttrV2_t

Uapi_ExpSwAttr_AdvancedV2_t

【说明】

优先使用参数结构体

【定义】

```
typedef struct Aec_uapi_advanced_attr_s {
    bool                            enable;
    uint8_t                         GridWeights[15 * 15];
    bool                            SetAeRangeEn;
    Aec_LinAeRange_t                SetLinAeRange;
    Aec_HdrAeRange_t                SetHdrAeRange;
} Aec_uapi_advanced_attr_t;

typedef Aec_uapi_advanced_attr_t Uapi_ExpSwAttr_AdvancedV2_t;
```

【成员】

成员名称	描述
enable	置1，优先使用该结构体中的参数
GridWeights	AE权重，大小为15X15。
SetAeRangeEn	置1，设置自动曝光分量range；置0，不设置自动曝光分量range，以调试文件为准
SetLinAeRange	线性曝光的自动曝光分量range参数值（并非最终生效range）
SetHdrAeRange	HDR曝光的自动曝光分量range参数值（并非最终生效range）

【注意事项】

- Uapi_ExpSwAttrV2_t 结构体中定义了一组AE权重，权重个数为15X15。算法内部根据硬件实际配置权重个数，进行权重的扩展。如有类似人脸曝光的需求，可使用Uapi_ExpSwAttr_AdvancedV2_t中定义的一组AE权重，其权重个数为15X15。算法内部根据硬件实际配置权重个数，进行权重的压缩。
- 如需使用Uapi_ExpSwAttr_Advanced_t中定义的一组AE权重，需要打开使能enable，将其置1。
- **通过api设置AE参数范围时，需要将SetAeRangeEn置1**，否则默认使用调试文件中的自动曝光参数范围。曝光参数范围的设置，按照曝光模式的不同，分为SetLinAeRange和SetHdrAeRange两套。其中SetHdrAeRange内支持各帧自动曝光参数范围的设置。另，最终生效的曝光时间及增益range需要在stAuto结构体中查询，此处仅为设置的range，而非最终生效range。

Aec_AeRange_t

【说明】
定义AE参数范围

【定义】

```
typedef struct Aec_AeRange_s {
    float                   Min;
    float                   Max;
} Aec_AeRange_t;
```

【成员】

成员名称	描述
Min	下限值
Max	上限值

Aec_LinAeRange_t

【说明】
定义AE线性模式的参数范围

【定义】

```
typedef struct Aec_LinAeRange_s {
    Aec_AeRange_t      stExpTimeRange;
    Aec_AeRange_t      stGainRange;
    Aec_AeRange_t      stIspDGainRange;
    Aec_AeRange_t      stPIrisRange;
} Aec_LinAeRange_t;
```

【成员】

成员名称	描述
stExpTimeRange	曝光时间范围，设置最大值和最小值，以毫秒为单位
stGainRange	Sensor 模拟增益范围，设置最大值和最小值
stIspDGainRange	ISP数字增益范围，设置最大值和最小值
stPIrisRange	光圈等效增益范围，仅支持P-Iris光圈大小控制

【注意事项】

- 各曝光分量最大值/最小值为默认值0时，设置的曝光分量范围不生效，各曝光分量实际最大值/最小值以AecRoute曝光分解路线节点最小值和最大值决定。
- 各曝光分量最大值/最小值不为0时，设置的曝光分量范围生效，当设置的曝光分量范围不超过sensor或ISP的限制，则各曝光分量实际最大值/最小值以设置的曝光分量范围为准，并对AecRoute曝光分解路线做校正，节点最大值/最小值改为设置的曝光分量最大值/最小值；若超过sensor或ISP的限制，则各曝光分量实际最大值/最小值以AecRoute曝光分解路线节点最大值和最小值为准。

Aec_HdrAeRange_t

【说明】
定义AE HDR模式的参数范围

【定义】

```
typedef struct Aec_HdrAeRange_s {
    Aec_AeRange_t      stExpTimeRange[3];
    Aec_AeRange_t      stGainRange[3];
    Aec_AeRange_t      stIspDGainRange[3];
    Aec_AeRange_t      stPIrisRange;
} Aec_HdrAeRange_t;
```

【成员】

成员名称	描述
stExpTimeRange	曝光时间范围，设置最大值和最小值，以秒为单位，数组0/1/2分别为短帧、中帧、长帧。
stGainRange	Sensor 模拟增益范围，设置最大值和最小值，数组0/1/2分别为短帧、中帧、长帧。
stIspDGainRange	ISP数字增益范围，设置最大值和最小值，数组0/1/2分别为短帧、中帧、长帧。
stPIrisRange	光圈等效增益值值范围，设置最大值和最小值

【注意事项】

- stExpTimeRange、stGainRange、stIspDGainRange预定义为包含3个成员的数组。2帧HDR模式下，仅成员0、1有效、分别表示短帧和长帧对应的曝光分量范围；3帧HDR模式下，0-2皆有效，分别表示短、中、长帧对应的曝光分量范围。
- 各曝光分量最大值/最小值为默认值0时，设置的曝光分量范围不生效，此时各曝光分量实际最大值/最小值以算法校正过的曝光分解路线节点最小值和最大值决定。
- 各曝光分量最大值/最小值不为0时，设置的曝光分量范围生效，当设置的曝光分量范围不超过sensor或ISP的限制，则各曝光分量实际最大值/最小值以设置的曝光分量范围为准，并对曝光分解路线做校正，节点最大值/最小值改为设置的曝光分量最大值/最小值；若超过sensor或ISP的限制，则各曝光分量实际最大值/最小值以曝光分解路线节点最大值和最小值为准。

Uapi_AeAttrV2_t

【说明】
定义AE自动曝光 属性

【定义】

```
typedef struct Uapi_AeAttrV2_s {
    Uapi_AeSpeedV2_t            stAeSpeed;
    //DelayFrmNum
    uint8_t                     BlackDelayFrame;
    uint8_t                     WhiteDelayFrame;
    //Auto/Fixed fps
    Uapi_AeFpsAttrV2_t          stFrmRate;
    Uapi_AntiFlickerV2_t        stAntiFlicker;
    //auto range
    Aec_LinAeRange_t            LinAeRange;//result LinAerange
    Aec_HdrAeRange_t            HdrAeRange;//result HdrAerange
} Uapi_AeAttrV2_t;
```

【成员】

成员名称	描述
stAeSpeed	自动曝光调节速度
BlackDelayFrame	自动曝光延时属性，图像亮度低于目标值超过BlackDelayFrame帧时，Ae开始调节
WhiteDelayFrame	自动曝光延时属性，图像亮度高于目标值超过WhiteDelayFrame帧时，Ae开始调节
stFrmRate	自动曝光帧率模式结构体，固定帧率模式或自动降帧模式
stAntiFlicker	抗工频闪烁参数
LinAeRange	线性模式自动曝光量范围结构体(最终生效的线性曝光range)
HdrAeRange	Hdr模式自动曝光量范围结构体（最终生效的hdr曝光range）

【注意事项】

- LinAeRange/HdrAeRange为算法内部经过校验校正后**实际使用的参数范围**。当api设置的AE参数范围超过sensor限制的参数范围时，会使用sensor限制的参数范围。sensor限制的参数范围，详见Rockchip_Tuning_Guide_ISP30_CN.pdf 文档中的AE模块sensorinfo参数。
- 设置自动曝光分量range需api调用上文Uapi_ExpSwAttr_AdvancedV2_t中的参数，将SetAeRangeEn置1，否则默认使用调试文件中的自动曝光参数范围。

Uapi_AeSpeedV2_t

【说明】
定义AE条件速度属性

【定义】

```
typedef struct CalibDb_AeSpeedV2_s {
    float                   DampOver;
    float                   DampUnder;
    float                   DampDark2Bright;
    float                   DampBright2Dark;
} CalibDb_AeSpeedV2_t;
typedef CalibDb_AeSpeedV2_t Uapi_AeSpeedV2_t;
```

【成员】

成员名称	描述
DampOver	环境亮度稳定，图像亮度高于目标值时对应的曝光调节速度，取值范围[0,1]
DampUnder	环境亮度稳定，图像亮度低于目标值时对应的曝光调节速度，取值范围[0,1]
DampDark2Bright	环境亮度突变，从暗到亮时对应的曝光调节速度，取值范围[0,1]
DampBright2Dark	环境亮度突变，从亮到暗时对应的曝光调节速度，取值范围[0,1]

Uapi_AeFpsAttrV2_t

【说明】
定义AE 帧率属性

【定义】

```
typedef struct CalibDb_AeFrmRateAttrV2_s {
    bool             isFpsFix;
    uint8_t          FpsValue;
} CalibDb_AeFrmRateAttrV2_t;
typedef CalibDb_AeFrmRateAttrV2_t Uapi_AeFpsAttrV2_t;
```

【成员】

成员名称	描述
isFpsFix	自动曝光固定帧率模式的使能，默认值为FALSE, 即采用自动降帧模式；值为TRUE时，表示固定帧率模式。
FpsValue	仅在固定帧率时有效，默认值为0时，固定使用驱动默认的帧率；值不会0时，使用设定的帧率值。

Uapi_AntiFlickerV2_t

【说明】
定义抗闪属性

【定义】

```
typedef struct CalibDb_AntiFlickerAttrV2_s {
    bool                            enable;
    CalibDb_FlickerFreq_t           Frequency;
    CalibDb_AntiFlickerMode_t       Mode;
} CalibDb_AntiFlickerAttrV2_t;
typedef CalibDb_AntiFlickerAttrV2_t Uapi_AntiFlickerV2_t;
```

【成员】

成员名称	描述
enable	工频抗闪功能使能状态，0：关闭抗闪功能；1：开启抗闪功能
Frequency	抗闪频率属性，共包含3种帧率，分别为：AECV2_FLICKER_FREQUENCY_OFF（抗闪使能位enable置0时有效）、AECV2_FLICKER_FREQUENCY_50HZ、AECV2_FLICKER_FREQUENCY_60HZ，默认为AECV2_FLICKER_FREQUENCY_50HZ（工频50赫兹）。
Mode	抗闪模式，共包含两种抗闪模式：AECV2_ANTIFLICKER_NORMAL_MODE（普通抗闪模式）、AECV2_ANTIFLICKER_AUTO_MODE（自动抗闪模式）

【注意事项】

- 关闭抗闪功能，需将抗闪使能位enable置0，算法内部自动配置Frequency=AECV2_FLICKER_FREQUENCY_OFF；如抗闪使能位enable置1时，抗闪频率配置为AECV2_FLICKER_FREQUENCY_OFF，该频率值将配置无效，使用默认值AECV2_FLICKER_FREQUENCY_50HZ。

- AECV2_ANTIFLICKER_NORMAL_MODE为普通抗闪模式，曝光时间可以根据亮度进行调节，最小曝光时间固定为 1/120 sec（ 60Hz）或 1/100 sec(50Hz)，不受曝光时间最小值的限制。

  有灯光的环境：曝光时间可与光源频率相匹配，能够防止图像闪烁。

  高亮度的环境：亮度越高，要求曝光时间就最短。而普通抗闪模式的最小曝光时间不能匹配光源频率，产生过曝。

- AECV2_ANTIFLICKER_AUTO_MODE为自动抗闪模式，曝光时间可以根据亮度进行调节，最小曝光时间可达到 sensor 的最小曝光时间。与普通抗闪模式的差别主要在高亮度的环境体现。高亮度的环境：最小曝光时间可以达到 sensor 的最小曝光时间，能够有效抑制过曝，但此时抗闪失效。

Uapi_MeAttrV2_t

【说明】

手动曝光参数设置，根据曝光模式分为LinearAE和HdrAE两套参数。

【定义】

```
typedef struct Uapi_MeAttrV2_s {
    Uapi_LinMeAttrV2_t     stLinMe;
    Uapi_HdrMeAttrV2_t     stHdrMe;
} Uapi_MeAttrV2_t;
```

【相关数据类型】

- Uapi_LinMeAttrV2_t
- Uapi_HdrMeAttrV2_t

Uapi_LinMeAttrV2_t

【说明】

LinearAE的手动曝光控制参数

【定义】

```
typedef struct Uapi_LinMeAttrV2_s {
    bool                ManualTimeEn;
    bool                ManualGainEn;
    bool                ManualIspDgainEn;
    float               TimeValue;
    float               GainValue;
    float               IspDGainValue;
} Uapi_LinMeAttrV2_t;
```

【成员】

成员名称	描述
ManualTimeEn	手动曝光时间使能，默认值为1
ManualGainEn	手动sensor增益使能，默认值为1
ManualIspDgainEn	手动ISP数字增益使能，默认值为1
TimeValue	手动曝光时间值，以s为单位，参数值受sensor限制
GainValue	手动sensor增益值，参数值受sensor限制
IspDGainValue	手动ISP数字增益值，参数值受ISP限制

【注意事项】

- 该模块仅在AeOptype = MANUAL时有效。ManualTimeEn,ManualGainEn, ManualIspDgainEn皆为1时，为手动模式；以上四者中只要任意一项不使能，则为半自动模式；以上四者皆为0，则等同自动模式，系统会报错提醒。
- 手动/半手动模式下，手动曝光时间和增益会受自动模式下的最大/最小曝光时间和增益限制。超出自动曝光限制的范围之后，将使用自动模式下最大/最小值替代。
- RK356X目前暂不支持ISP数字增益，故ManualIspDgainEn、IspDGainValue皆无效。

Uapi_HdrMeAttrV2_t

【说明】

HdrAE的手动曝光控制参数

【定义】

```
typedef struct Uapi_HdrMeAttrV2_s {
    bool                    ManualTimeEn;
    bool                    ManualGainEn;
    bool                    ManualIspDgainEn;

    Cam3x1FloatMatrix_t     TimeValue;
    Cam3x1FloatMatrix_t     GainValue;
    Cam3x1FloatMatrix_t     IspDGainValue;

} Uapi_HdrMeAttrV2_t;
```

【成员】

变量含义与Uapi_LinMeAttrV2_t相同

【注意事项】

- TimeValue/GainValue/IspDGainValue皆为成员个数为3的数组。Hdr 2帧模式下，数组[0-1]成员有效，分别表示短、长帧；Hdr 3帧模式下，数组[0-2]成员皆有效，分别对应短、中、长帧。
- 手动/半手动模式下，手动曝光时间和增益会受自动模式下的最大/最小曝光时间和增益限制。超出自动曝光限制的范围之后，将使用自动模式下最大/最小值替代。
- RK356X目前不支持ISP数字增益，故ManualIspDgainEn、IspDGainValue皆无效。

Uapi_LinAeRouteAttr_t

【说明】

定义AE线性曝光分解路径属性

【定义】

```
typedef struct CalibDb_LinAeRoute_AttrV2_s {
    float* TimeDot;
    int TimeDot_len;
    float* GainDot;
    int GainDot_len;
    float* IspDGainDot;
    int IspDGainDot_len;
    int* PIrisDot;
    int PIrisDot_len;
} CalibDb_LinAeRoute_AttrV2_t;

typedef struct Uapi_LinAeRouteAttr_s {
  rk_aiq_uapi_sync_t            sync;
  CalibDb_LinAeRoute_AttrV2_t   Params;
} Uapi_LinAeRouteAttr_t;
```

【成员】

成员名称	描述
TimeDot	sensor曝光时间节点，单位为s
GainDot	sensor曝光增益节点
IspgainDot	Isp数字增益节点
PIrisGainDot	光圈等效增益节点

【注意事项】

- 曝光分解曲线节点个数没有限制，但建议不要少于6个
- 节点的曝光量是曝光时间、sensor增益、ISP数字增益、光圈等效增益等各分量的乘积。节点曝光量必须单调递增，即后一个节点的曝光量必须大于前一个节点的曝光量。第一个节点的曝光量最小，第二个节点的曝光量最大。
- 节点中曝光时间分量的单位为秒，最小值允许为0，实际最小曝光时间代码内部会根据sensor限制进行校正。
- 光圈分量仅支持P-Iris, 不支持DC-Iris。P-iris等效增益分量仅在Airis自动光圈功能使能时有效，否则默认光圈固定为初始值大小。P-iris等效增益的计算详见AecIrisCtrl模块。
- 设置的曝光分解路线节点不是最终生效的曝光分解路线。系统最终各曝光分量的实际最大/小值由曝光分解节点和手动配置的曝光分量最大/小值共同决定。先对曝光分解路线节点最大/小值做第一次校正，当节点最大/小值不超过sensor或isp的限制时，节点最大/小值不变；当节点最大/小值超过sensor或isp的限制时，节点最大/小值以sensor或isp的限制为准。当手动配置的曝光分量最大/小值为0时，最终生效的曝光分解路线以第一次校正的分解路线为准；当手动配置的曝光分量最大/小值不为0时，且设置的最大/小值不超过sensor或isp的限制时，对曝光分解路线做第二次校正，节点最大/小值以手动设置的范围为准；若设置曝光分量的最大/小值超过sensor或isp的限制时，曝光分解路线曝光分量的节点最大/小值以第一次校正结果为准。
- 如果相邻节点的曝光量增加，则应该只有一个曝光分量增加，其他曝光分量固定。增加的分量决定该段路线的分配策略。例如增益分量增加，其他分量固定，那么该段路线的分配策略是增益优先。
- RK356X目前暂不支持ISP数字增益，故IspgainDot无效。

Uapi_HdrAeRouteAttr_t

【说明】

定义AE HDR曝光分解路径属性

【定义】

```
typedef struct CalibDb_HdrAeRoute_AttrV2_s {
    float* Frm0TimeDot;
    int Frm0TimeDot_len;
    float* Frm0GainDot;
    int Frm0GainDot_len;
    float* Frm0IspDGainDot;
    int Frm0IspDGainDot_len;
    float* Frm1TimeDot;
    int Frm1TimeDot_len;
    float* Frm1GainDot;
    int Frm1GainDot_len;
    float* Frm1IspDGainDot;
    int Frm1IspDGainDot_len;
    float* Frm2TimeDot;
    int Frm2TimeDot_len;
    float* Frm2GainDot;
    int Frm2GainDot_len;
    float* Frm2IspDGainDot;
    int Frm2IspDGainDot_len;
    int* PIrisDot;
    int PIrisDot_len;
} CalibDb_HdrAeRoute_AttV2r_t;

typedef struct Uapi_HdrAeRouteAttr_s {
  rk_aiq_uapi_sync_t            sync;
  CalibDb_HdrAeRoute_AttrV2_t   Params;
} Uapi_HdrAeRouteAttr_t;
```

【成员】

成员名称	描述
Frm0/1/2TimeDot	曝光时间节点，单位为秒。Hdr 2帧模式时，仅Frm0/1TimeDot有效；Hdr 3帧模式时，Frm0/1/2TimeDot皆有效。Frm0~3依次为曝光量从短至长的帧序号。
Frm0/1/2GainDot	sensor增益节点。Hdr 2帧模式时，仅Frm0/1GainDot有效；Hdr 3帧模式时，Frm0/1/2GainDot皆有效。此处增益值为实际值，单位为1x。Frm0~3依次为曝光量从短至长的帧序号。
Frm0/1/2IspDGainDot	Isp数字增益节点。Hdr 2帧模式时，仅Frm0/1IspDGainDot有效；Hdr 3帧模式时，Frm0/1/2IspDGainDot皆有效。此处增益值为实际值，单位为1x。Frm0~3依次为曝光量从短至长的帧序号。
PIrisDot	光圈等效增益节点，此处增益值为实际值，单位为1x

【注意事项】

- 曝光分解曲线节点个数不限，**建议至少设置6个节点**，才可实现曝光分解的平滑。
- 需要注意的是：HDR 2帧模式下，仅需设置Frm0/1TimeDot、Frm0/1GainDot、Frm0/1IspDGainDot，分别对应实际的短、长帧；HDR 3帧模式下，需设置Frm0/1/2TimeDot、

Frm0/1/2GainDot、Frm0/1/2IspDGainDot，分别对应短、中、长帧。设置HDR模式下各帧的sensor曝光时间时，需要合理分配曝光时间，各帧**曝光时间的总和不能超过帧率所允许的最大曝光时间**！

- 节点的曝光量是曝光时间、sensor增益、ISP数字增益、光圈等效增益等各分量的乘积。节点曝光量必须单调递增，即后一个节点的曝光量必须大于前一个节点的曝光量。第一个节点的曝光量最小，第二个节点的曝光量最大。
- 节点中曝光时间分量的单位为秒，最小值允许为0，实际最小曝光时间代码内部会根据sensor限制进行校正。
- 光圈分量仅支持P-Iris, 不支持DC-Iris。P-iris等效增益分量仅在Airis自动光圈功能使能时有效，否则默认光圈固定为初始值大小。P-iris等效增益的计算详见AecIrisCtrl模块。
- 设置的曝光分解路线节点不是最终生效的曝光分解路线。系统最终各曝光分量的实际最大/小值由曝光分解节点和手动配置的曝光分量最大/小值共同决定。先对曝光分解路线节点最大/小值做第一次校正，当节点最大/小值不超过sensor或isp的限制时，节点最大/小值不变；当节点最大/小值超过sensor或isp的限制时，节点最大/小值以sensor或isp的限制为准。当手动配置的曝光分量最大/小值为0时，最终生效的曝光分解路线以第一次校正的分解路线为准；当手动配置的曝光分量最大/小值不为0时，且设置的最大/小值不超过sensor或isp的限制时，对曝光分解路线做第二次校正，节点最大/小值以手动设置的范围为准；若设置曝光分量的最大/小值超过sensor或isp的限制时，曝光分解路线曝光分量的节点最大/小值以第一次校正结果为准。
- 如果相邻节点的曝光量增加，则应该只有一个曝光分量增加，其他曝光分量固定。增加的分量决定该段路线的分配策略。例如增益分量增加，其他分量固定，那么该段路线的分配策略是增益优先。
- 356X目前暂不支持ISP数字增益，故Frm0/1/2ispDGainDot皆无效。

Uapi_LinExpAttrV2_t

【说明】

定义AE线性曝光调试参数

【定义】

```
typedef struct CalibDb_LinearAE_AttrV2_s {
    uint8_t RawStatsEn;
    float ToleranceIn;
    float ToleranceOut;
    float Evbias;
    CalibDb_AeStrategyModeV2_t      StrategyMode;
    CalibDb_LinExpInitExpV2_t       InitExp;
    CalibDb_LinAeRoute_AttrV2_t     Route;
    CalibDb_AecDynamicSetpointV2_t  DySetpoint;
    CalibDb_AecBacklightV2_t        BackLightCtrl;
    CalibDb_AecOverExpCtrlV2_t OverExpCtrl;
} CalibDb_LinearAE_AttrV2_t;

typedef struct Uapi_LinExpAttrV2_s {
  rk_aiq_uapi_sync_t            sync;
  CalibDb_LinearAE_AttrV2_t     Params;
} Uapi_LinExpAttrV2_t;
```

【成员】

成员名称	描述
RawStatsEn	线性曝光使用Raw域统计亮度功能使能
EvBias	自动曝光调节时，曝光量的偏差百分比，单位为%，参考取值范围为[-200,+200]
ToleranceIn/Out	自动曝光调节时，画面亮度的容忍度。单位为%，取值范围为[0,100]
StrategyMode	自动曝光策略模式，高光优先或低光优先【暂时无效】
DySetpoint	自动曝光调节的动态目标亮度值属性，随曝光量动态变化
BackLightCtrl	背光补偿功能参数
OverExpCtrl	强光抑制功能参数

【注意事项】

- 曝光量偏差EvBias，用于特殊场景下对（固定/动态）目标亮度值（SetPoint/IRSetPoint）进行微调。真实生效目标亮度为（SetPoint）*[1+abs(EvBias)/100]^[EvBias/abs(EvBias)]。如设置EvBias=100时，目标亮度为默认参数的2倍；EvBias=-100时，目标亮度为默认参数的1/2。
- 自动曝光画面亮度的容忍度为Tolerance，当自动曝光收敛时画面亮度值B应在 [真实生效目标亮度X（1-Tolerance/100），真实生效目标亮度 X（1+Tolerance/100）] 范围内。ToleranceIn/Out设置较大，一方面会影响AE的响应速度，一方面会影响EvBias值。当EvBias调整的间隔值低于ToleranceIn/Out，有可能导致亮度调整不生效。
- StrategyMode暂无效。

【相关数据类型】

- CalibDb_AecDynamicSetpointV2_t
- CalibDb_AecBacklightV2_t
- CalibDb_AecOverExpCtrlV2_t

CalibDb_AecDynamicSetpointV2_t

【说明】
定义AE动态目标值

【定义】

```
typedef struct CalibDb_AecDynamicSetpointV2_s {
    float* ExpLevel;
    int ExpLevel_len;
    float* DySetpoint;
    int DySetpoint_len;
} CalibDb_AecDynamicSetpointV2_t;
```

【成员】

成员名称	描述
ExpLevel	动态曝光量节点属性，节点值为当前曝光量值，节点个数不限，建议至少设置6个节点，以防曝光过渡不平滑。
DySetpoint	动态目标亮度值节点属性，节点值随曝光量动态变化，曝光量节点值越大，目标亮度节点值越小，并与曝光量节点一一对应。节点个数不限，需要与ExpLevel节点个数一致，建议至少设置6个节点，以防曝光过渡不平滑。

Uapi_HdrExpAttrV2_t

【说明】

定义AE HDR曝光调试参数

【定义】

```
typedef struct CalibDb_HdrAE_AttrV2_s {
    float ToleranceIn;
    float ToleranceOut;
    float Evbias;
    CalibDb_AeStrategyModeV2_t StrategyMode;
    float LumaDistTh; //used for area-growing
    CalibDb_HdrExpInitExpV2_t InitExp;
    CalibDb_HdrAeRoute_AttrV2_t Route;
    CalibDb_ExpRatioCtrlV2_t ExpRatioCtrl;
    CalibDb_LongFrmCtrlV2_t LongFrmMode;
    CalibDb_LfrmCtrlV2_t LframeCtrl;
    CalibDb_MfrmCtrlV2_t MframeCtrl;
    CalibDb_SfrmCtrlV2_t SframeCtrl;
} CalibDb_HdrAE_AttrV2_t;

typedef struct Uapi_HdrExpAttrV2_s {
  rk_aiq_uapi_sync_t            sync;
  CalibDb_HdrAE_AttrV2_t        Params;
} Uapi_HdrExpAttrV2_t;
```

【成员】

成员名称	描述
ToleranceIn/Out	自动曝光调节时，画面亮度的容忍度。单位为%，取值范围为[0,100]
LongfrmMode	长帧模式参数
StrategyMode	自动曝光策略模式，高光优先或低光优先（AECV2_STRATEGY_MODE_LOWLIGHT_PRIOR/AECV2_STRATEGY_MODE_HIGHLIGHT_PRIOR）
Evbias	自动曝光调节时，曝光量的偏差百分比，单位为%，参考取值范围为[-200,+200]
ExpRatioCtrl	HdrAE曝光比控制模块，仅在Hdr模式多帧合成下有效
LframeCtrl	长帧控制参数
MframeCtrl	中帧控制参数，仅在HDR 3帧模式下有效
SframeCtrl	短帧控制参数

【相关数据类型】

- CalibDb_LFrameCtrlV2_t
- CalibDb_MFrameCtrlV2_t
- CalibDb_SFrameCtrlV2_t

Uapi_IrisAttrV2_t

【说明】

光圈控制参数

【定义】

```
typedef struct CalibDb_AecIrisCtrlV2_s {
    uint8_t Enable;
    CalibDb_IrisTypeV2_t IrisType;
    RKAiqOPMode_t IrisOpType;
    CalibDb_MIris_AttrV2_t ManualAttr;
    CalibDb_PIris_AttrV2_t PIrisAttr;
    CalibDb_DCIris_AttrV2_t DCIrisAttr;
} CalibDb_AecIrisCtrlV2_t;

typedef struct Uapi_IrisAttrV2_s {
  rk_aiq_uapi_sync_t            sync;
  CalibDb_AecIrisCtrlV2_t       Params;
} Uapi_IrisAttrV2_t;
```

【成员】

成员名称	描述
Enable	自动光圈控制功能的使能
IrisType	光圈类型，P（即P-iris光圈）或DC（即DC-iris光圈）
IrisOpType	光圈控制模式：分为自动(RK_AIQ_OP_MODE_AUTO)模式/手动(RK_AIQ_OP_MODE_MANUAL)模式
ManualAttr	手动光圈参数
PIrisAttr	P光圈属性参数
DCIrisAttr	DC光圈属性参数

CalibDb_MIris_AttrV2_t

【说明】

手动光圈参数

【定义】

```
typedef struct CalibDb_MIris_AttrV2_s {
    int PIrisGainValue;
    int DCIrisHoldValue;
} CalibDb_MIris_AttrV2_t;
```

【成员】

成员名称	描述
PIrisGainValue	手动P光圈等效增益值，参数值受P光圈设备限制，取值范围为[1，1024]
DCIrisHoldValue	DC光圈HoldValue值，参数值与DC光圈设备有关，取值范围为[0，100]

【注意事项】

- IrisOpType = RK_AIQ_OP_MODE_MANUAL，手动光圈使能。当光圈类型为P光圈时，仅PirisGainValue有效；当光圈类型为DC光圈时，仅DCIrisHoldValue有效。
- DCIrisHoldValue，手动模式下直接设置电机的PWM占空比值，取值范围[0，100]。手动模式下若设置为HoldValue值（即AecIrisCtrl中ClosePwmDuty到OpenPwmDuty区间内的值），则DC光圈孔径维持在当前大小；若设置的值大于OpenPwmDuty，则光圈处于打开状态，该值越大打开的速度越大；若设置的值小于ClosePwmDuty，则光圈处于关闭状态，该值越小关闭的速度越大。

CalibDb_PIris_AttrV2_t

【说明】

P光圈属性参数

【定义】

```
#define AEC_PIRIS_STAP_TABLE_MAX (1024)
typedef struct CalibDb_PIris_AttrV2_s {
    uint16_t              TotalStep;
    uint16_t              EffcStep;
    bool                  ZeroIsMax;
    uint16_t              StepTable[AEC_PIRIS_STAP_TABLE_MAX];
} CalibDb_PIris_AttrV2_t;
```

【成员】

成员名称	描述
TotalStep	P-iris步进电机总步数，具体大小与P-iris镜头有关。
EffcStep	P-iris步进电机的可用步数，具体大小与P-iris镜头有关。
ZeroIsMax	P-iris步进电机step0是否对应最大光圈位置，具体取值与P-iris镜头有关。该值为0，代表步进电机位置为step0时，光圈开到最小；该值为1，代表步进电机位置为step0时，光圈开到最大。
StepTable	P-iris步进电机位置与光圈等效增益的映射表，具体数值与P-iris镜头有关。

CalibDb_DCIris_AttrV2_t

【说明】

DC光圈属性

【定义】

```
typedef struct CalibDb_DCIris_AttrV2_s {
    float       Kp;
    float       Ki;
    float       Kd;
    int         MinPwmDuty;
    int         MaxPwmDuty;
    int         OpenPwmDuty;
    int         ClosePwmDuty;
} CalibDb_DCIris_AttrV2_t;
```

【成员】

成员名称	描述
Kp	比例系数, 用于限制光圈剧烈变化时光圈的开关速度，该值越大，光线剧烈变化时光圈打开和关闭的速度越慢。该值过大，调节过程制动就会超前，致使调节时间过长；该值过小，调节过程制动就会落后，从而导致超调增加。该值的合理设置与DC-iris镜头及电路特性有关。建议值为0.5。 取值范围[0，1]。
Ki	积分系数，用于调节光圈的开关速度，该值越大光圈打开和关闭的速度越大。该值过大，容易出现超调导致振荡；该值过小，光圈调节速度较慢、环境亮度变化较剧烈时容易发生振荡。建议值为0.2。 取值范围[0，1]。
Kd	微分系数，用于调节光圈的开关速度，该值越大光圈打开和关闭的速度越大。建议值为0.3。 取值范围[0，1]。
MinPwmDuty	最小PWM占空比，具体大小与DC-iris镜头、电路特性有关，单位为%。该值越小，所支持的光圈关闭速度越快，但容易导致光圈振荡。取值范围[0,100]，默认值为0。
MaxPwmDuty	最大PWM占空比，具体大小与DC-iris镜头、电路特性有关，单位为%。该值越大，所支持的光圈打开速度越快，该值过小，可能导致光圈尚未达到最大时就退出光圈控制。取值范围[0,100]，默认值为100。
OpenPwmDuty	光圈打开时的PWM占空比阈值，当光圈PWM占空比高于（不含）OpenPwmDuty时，光圈处于打开状态。具体大小与DC-iris镜头有关，单位为%。
ClosePwmDuty	光圈关闭时的PWM占空比阈值，当光圈PWM占空比小于（不含）ClosePwmDuty时，光圈处于关闭状态。具体大小与DC-iris镜头有关，单位为%。

【注意事项】

- 自动光圈功能关闭时，对于DC-iris光圈，默认会打开到最大；对于P-iris光圈，默认会打开到最大光圈所对应的步进电机位置。如想改变上述光圈位置，可至AecInitValue模块中修改InitPIrisGainValue、InitDCIrisDutyValue。
- 自动光圈Airis算法的基本控制流程如下：

  针对DC-iris镜头，Airis根据当前亮度与目标亮度的偏差值，控制DC-iris镜头的光圈大小。当曝光达到最小值时，且当前亮度超出目标亮度容忍度范围，将退出AE控制，曝光时间及曝光增益固定不变，进入AIris控制范围。若当前画面亮度稳定且DC-iris的PWM占空值大于OpenPwmDuty时，认为当前光圈达到最大，退出AIris光圈控制，控制权交由AE。

  针对P-iris镜头，光圈控制通过AecRoute模块进行。P-iris镜头的光圈大小换算为等效增益，参与曝光分解计算。

- P-iris的步进电机位置与光圈等效增益映射表StepTable一般根据镜头厂家提供的步进电机位置与光圈孔径对应关系制作。P-iris的控制是通过AE的AecRoute模块来控制的，该模块将光圈孔径大小换算成等效增益，因此要求P-iris的光圈控制需要具有较好的线性度。等效增益的取值范围为[1,1024]，用等效增益1024表示F1.0,等效增益512表示F1.4，以此类推，等效增益1表示F32.0。制作表时，需要将步进电机位置对应的光圈孔径换算为等效增益，填入StepTable中，并固定按照步进电机位置递增（即step0、step1……stepN）的顺序填入。
- TotalStep表示P-iris步进电机总步数，具体大小与P-iris镜头有关。EffcStep表示 P-iris步进电机的可用步数，一般要求小于TotalStep。因为为靠近光圈关闭端的位置，其对应等效增益的值误差较大，光圈调节过程中容易出现振荡，所以通常不会使用光圈关闭端附近的步进位置。
- 表4-1为P-iris步进电机位置与光圈孔径和等效增益的对应表，以此表为例来说明StepTable该如何设置。表4-1中第1-2、4-5列的步进电机位置step和光圈孔径面积的对应关系为某镜头原厂提供。该款P-iris镜头的步进电机调节总步数为81，step0时对应的光圈孔径最大，标称最大光圈数为1.4。光圈数为1.4时对应的等效增益为512，故step0处对应的等效增益为512。其他孔径面积对应的等效增益，此处以step3为例，计算方式如下：step3的孔径面积为195.869，对应等效增益=512*（195.869/201.062）= 499（四舍五入）。以此类推，其他步进电机位置对应的等效增益值也可据此算出。从表1-1中可知，步进电机位置靠近关闭端时，对应的孔径面积很小，与最大的孔径面积相差可达几千倍，对应的等效增益值误差较大，因此建议靠近光圈关闭端的步进电机位置不要使用，以免因为误差导致曝光振荡。将表中各步进电机位置对应的等效增益按照步进电机位置递增（即step0、step1……stepN）的顺序填入StepTable。
- DC-iris的OpenPwmDuty与ClosePwmDuty取值需要进行实测，其具体值与DC-iris镜头相关。对于部分镜头，存在当PWM占空比大于OpenPwmDuty时，光圈执行打开操作；当PWM占空比小于OpenPwmDuty时，光圈执行关闭操作；当PWM占空比大于等于ClosePwmDuty且小于等于OpenPwmDuty时，光圈稳定在当前位置，该区间内的值皆为HoldValue。另存在某些镜头，只存在一个光圈开关的阈值，即当PWM占空比大于该阈值时，光圈执行打开操作；当PWM占空比小于该阈值时，光圈执行关闭操作；当PWM占空比等于该阈值时，光圈稳定在当前位置，该阈值即为HoldValue。此时可令ClosePwmDuty = OpenPwmDuty = HoldValue。
- 光圈的手动模式参数设置与曝光的手动模式一致。需要使用手动光圈功能时，需将AecOpType设置为手动模式，并使能AecManualCtrl模块中的ManualIrisEn参数。当IrisType为P-iris时，仅PirisGainValue有效；当IrisType为P-iris时时，仅DCIrisValue有效。

表4-1 P-iris步进电机位置与光圈孔径和等效增益的对应表

Step	孔径面积(mm2)	等效增益	Step	孔径面积(mm2)	等效增益
0	201.062	512	41	56.653	144
1	200.759	511	42	53.438	136
2	198.583	506	43	50.282	128
3	195.869	499	44	47.188	120
4	192.879	491	45	44.159	112
5	189.677	483	46	41.197	105
6	186.293	474	47	38.307	98
7	182.744	465	48	35.49	90
8	179.035	456	49	32.751	83
9	175.271	446	50	30.093	77
10	171.484	437	51	27.519	70
11	167.681	427	52	25.034	64
12	163.865	417	53	22.642	58
13	160.036	408	54	20.347	52
14	156.198	398	55	18.154	46
15	152.351	388	56	16.068	41
16	148.499	378	57	14.096	36
17	144.642	368	58	12.245	31
18	140.783	359	59	10.522	27
19	136.925	349	60	8.935	23
20	133.069	339	61	7.484	19
21	129.217	329	62	6.169	16
22	125.371	319	63	4.987	13
23	121.535	309	64	3.936	10
24	117.709	300	65	3.014	8
25	113.897	290	66	2.22	6
26	110.1	280	67	1.55	4
27	106.321	271	68	1.003	3
28	102.562	261	69	0.577	1
29	98.826	252	70	0.268	1

Step	孔径面积(mm2)	等效增益	Step	孔径面积(mm2)	等效增益
30	95.115	242	71	0.075	0
31	91.431	233	72	close	0
32	87.777	224	73	close	0
33	84.156	214	74	close	0
34	80.569	205	75	close	0
35	77.02	196	76	close	0
36	73.51	187	77	close	0
37	70.043	178	78	close	0
38	66.621	170	79	close	0
39	63.247	161	80	close	0
40	59.923	153			

Uapi_ExpWin_t

【说明】
定义AE统计窗口属性参数

【定义】

```
typedef struct window {
    uint16_t h_offs;
    uint16_t v_offs;
    uint16_t h_size;
    uint16_t v_size;
} window_t;

typedef struct Uapi_ExpWin_s {
  rk_aiq_uapi_sync_t            sync;
  window                        Params;
} Uapi_ExpWin_t;
```

【成员】

成员名称	描述
h_offs	窗口左上角相对坐标原点的水平偏移值，这里的坐标原点指sensor感光区域左上角
v_offs	窗口左上角相对坐标原点的垂直偏移值，这里的坐标原点指sensor感光区域左上角
h_size	窗口水平方向尺寸
v_size	窗口竖直方向尺寸

Uapi_ExpQueryInfo_t

【说明】

定义AE曝光参数查询

【定义】

```
typedef struct Uapi_ExpQueryInfo_s {

    bool                IsConverged;
    bool                IsExpMax;
    float               LumaDeviation;
    float               HdrLumaDeviation[3];

    float               MeanLuma;
    float               HdrMeanLuma[3];

    float               GlobalEnvLux;
    float               BlockEnvLux[ISP2_RAWAE_WINNUM_MAX];

    RKAiqAecExpInfo_t CurExpInfo;
    unsigned short    Piris;
    float               LinePeriodsPerField;
    float               PixelPeriodsPerLine;
    float               PixelClockFreqMHZ;

} Uapi_ExpQueryInfo_t;
```

【成员】

成员名称	描述
IsConverged	自动曝光是否收敛
IsExpMax	ISP曝光是否达到最大值
LumaDeviation	线性模式下，AEC的目标值与实际画面亮度的差值，该值为正，表示实际亮度大于目标亮度；该值为负，表示实际亮度小于目标亮度；该值为0，表示实际画面亮度在目标值的容忍度范围内。
HdrLumaDeviation	Hdr模式下，各帧的亮度目标值与实际画面亮度的差值，该值为正，表示实际亮度大于目标亮度；该值为负，表示实际亮度小于目标亮度；该值为0，表示实际画面亮度在目标值的容忍度范围内。2帧模式下，仅0、1有效，分别表示短、长帧；3帧模式下，0-2皆有效，分别表示短、中、长帧。
MeanLuma	线性模式下，当前画面的平均亮度。
HdrMeanLuma	HDR模式下，各帧当前画面的平均亮度。2帧模式下，数组仅0、1成员有效，分别表示短、长帧；3帧模式下，数组0-2成员皆有效，分别表示短、中、长帧。
CurExpInfo	当前曝光参数，包括sensor曝光时间、sensor曝光增益值。按照曝光模式，分为LinearExp和HdrExp[3] 2套曝光参数。
Piris	光圈
LinePeriodsPerField	sensor的VTS
PixelPeriodsPerLine	sensor的HTS
PixelClockFreqMHZ	sensor的像素时钟频率(单位：兆赫兹)

常见问题定位及debug方法

若出现画面亮度闪烁、过冲、亮度不符合预期等问题时，建议通过抓取有问题场景的AE LOG进行分析，有助于快速定位问题、提高工作效率。

曝光统计同步测试功能

标定ISP模块前，需要驱动人员或tuning人员填写调试IQ XML中的SensorInfo参数。这个模块的涉及到曝光参数的设置，如设置错误可能发生曝光出错、闪烁等现象。建议配置完sensorinfo参数之后，开启SyncTest功能进行自测。sensorinfo参数含义说明参考《Rockchip_Tuning_Guide_ISP30》。

SyncTest功能通过循环设置N组不同曝光值，可测试sensor的曝光时间和曝光增益、及DCG切换生效帧数是否正确，还可用于测试曝光的线性度，从而确认曝光时间和曝光增益的寄存器值转换公式及相关参数是否正确。

SyncTest功能参数介绍如下：

【描述】

曝光与统计的同步测试功能，支持按照给定间隔帧数，循环设置N组不同的曝光值，用于debug及验证曝光分量（曝光时间、曝光增益）的生效帧数及sensor曝光参数设置是否正确。

【成员】

成员名称	描述
Enable	曝光与统计同步测试功能的使能
IntervalFrm	曝光切换间隔帧数
AlterExp	曝光切换参数

- AlterExp

根据模式的不同，分为LinearAE和HdrAE两套参数。

成员名称	描述
TimeValue	曝光时间值
GainValue	曝光增益值
IspDgainValue	Isp数字增益值
DcgMode	Dcg模式值
PIrisGainValue	P-iris等效增益值

如参数设置正确，LOG示例如下（仅截取LOG中的关键信息）。红框所示为曝光切换位置，可见亮度并无发生突变且与曝光值相匹配，无延迟或提前线性，此时可基本判断sensorinfo参数设置正确。

```
>>> Framenum=116 Cur gain=5.988304,time=0.019991,Meanluma=44.751110,piris=0
>>> Framenum=117 Cur gain=5.988304,time=0.019991,Meanluma=44.755554,piris=0
>>> Framenum=118 Cur gain=5.988304,time=0.019991,Meanluma=44.755554,piris=0
>>> Framenum=119 Cur gain=5.988304,time=0.019991,Meanluma=44.764446,piris=0
>>> Framenum=120 Cur gain=5.988304,time=0.019991,Meanluma=44.755554,piris=0
>>> Framenum=121 Cur gain=5.988304,time=0.019991,Meanluma=44.755554,piris=0
>>> Framenum=122 Cur gain=5.988304,time=0.019991,Meanluma=44.768890,piris=0
>>> Framenum=123 Cur gain=5.988304,time=0.019991,Meanluma=44.782223,piris=0
>>> Framenum=124 Cur gain=1.000000,time=0.019991,Meanluma=24.568890,piris=0
>>> Framenum=125 Cur gain=1.000000,time=0.019991,Meanluma=24.484444,piris=0
>>> Framenum=126 Cur gain=1.000000,time=0.019991,Meanluma=24.484444,piris=0
>>> Framenum=127 Cur gain=1.000000,time=0.019991,Meanluma=24.484444,piris=0
>>> Framenum=128 Cur gain=1.000000,time=0.019991,Meanluma=24.480000,piris=0
>>> Framenum=129 Cur gain=1.000000,time=0.019991,Meanluma=24.480000,piris=0
>>> Framenum=130 Cur gain=1.000000,time=0.019991,Meanluma=24.471111,piris=0
>>> Framenum=131 Cur gain=1.000000,time=0.019991,Meanluma=24.475555,piris=0
```

曝光变化时出现闪烁

可能导致曝光变化时出现闪烁的几种原因：

（1）CISExpUpdate模块中的gain、time生效时刻帧数错误。常见LOG示例如下（仅截取每帧关键LOG行）：

```
Cur gain=1.937500,time=0.010015,RawMeanluma=24.622223,YuvMeanluma=34.124443,IsConverged=0
Cur gain=1.937500,time=0.010015,RawMeanluma=37.795555,YuvMeanluma=54.217777,IsConverged=0
Cur gain=1.937500,time=0.010015,RawMeanluma=37.257778,YuvMeanluma=52.435555,IsConverged=0
Cur gain=1.328125,time=0.020000,RawMeanluma=37.288887,YuvMeanluma=52.480000,IsConverged=0
Cur gain=1.390625,time=0.020000,RawMeanluma=60.342224,YuvMeanluma=82.528893,IsConverged=0
Cur gain=1.453125,time=0.020000,RawMeanluma=46.471111,YuvMeanluma=63.831112,IsConverged=0
Cur gain=1.000000,time=0.030015,RawMeanluma=48.048889,YuvMeanluma=66.195557,IsConverged=0
Cur gain=1.187500,time=0.020000,RawMeanluma=65.511108,YuvMeanluma=87.622223,IsConverged=0
Cur gain=1.125000,time=0.020000,RawMeanluma=38.071110,YuvMeanluma=53.022221,IsConverged=0
Cur gain=1.062500,time=0.020000,RawMeanluma=42.928890,YuvMeanluma=60.355556,IsConverged=0
Cur gain=1.640625,time=0.010015,RawMeanluma=41.328888,YuvMeanluma=57.666668,IsConverged=0
Cur gain=1.593750,time=0.010015,RawMeanluma=25.453333,YuvMeanluma=35.293335,IsConverged=0
Cur gain=1.562500,time=0.010015,RawMeanluma=33.360001,YuvMeanluma=47.897778,IsConverged=0
Cur gain=1.531250,time=0.010015,RawMeanluma=32.595554,YuvMeanluma=46.084446,IsConverged=0
```

红框所标为亮度出错位置，可见随着曝光增长或降低，对应亮度与曝光变化趋势相反。通过观察，可发现亮度突变都发生在曝光时间和曝光增益同时变化后的第二帧。亮度的变化趋势与曝光时间变化趋势一致，因此可以判断gain、time的生效帧数出错，曝光时间和曝光增益的变化并未同时生效，导致亮度和曝光不匹配。可以通过修改gain、time生效帧数解决此问题。上述问题可以使用AE的syncTest功能进行复现，设置两组曝光（曝光增益和曝光时间不同），令其来回切换，查看LOG可知是否复现。

（2）驱动错误导致的亮度来回震荡无法收敛，常发生在高亮场景，常见LOG示例如下（仅截取每帧关键LOG行）

```
Framenum=3378 Cur Piris=128, Sgain=1.188502,Stime=0.000044,m
Framenum=3379 Cur Piris=128, Sgain=1.148154,Stime=0.000044,m
Framenum=3380 Cur Piris=128, Sgain=1.148154,Stime=0.000044,m
Framenum=3381 Cur Piris=128, Sgain=1.230269,Stime=0.000044,m
 Framenum=3382 Cur Piris=128, Sgain=1.000000,Stime=0.000059,
 Framenum=3383 Cur Piris=128, Sgain=1.035142,Stime=0.000059,
 Framenum=3384 Cur Piris=128, Sgain=1.035142,Stime=0.000059,
 Framenum=3385 Cur Piris=128, Sgain=1.071519,Stime=0.000059,
 Framenum=3386 Cur Piris=128, Sgain=1.273503,Stime=0.000044,
 Framenum=3387 Cur Piris=128, Sgain=1.188502,Stime=0.000044,
 Framenum=3388 Cur Piris=128, Sgain=1.148154,Stime=0.000044,
 Framenum=3389 Cur Piris=128, Sgain=1.148154,Stime=0.000044,
 Framenum=3390 Cur Piris=128, Sgain=1.230269,Stime=0.000044,
 Framenum=3391 Cur Piris=128, Sgain=1.000000,Stime=0.000059,
Framenum=3392 Cur Piris=128, Sgain=1.000000,Stime=0.000059,m
Framenum=3393 Cur Piris=128, Sgain=1.000000,Stime=0.000059,m
 Framenum=3394 Cur Piris=128, Sgain=1.035142,Stime=0.000059,
 Framenum=3395 Cur Piris=128, Sgain=1.273503,Stime=0.000044,
 Framenum=3396 Cur Piris=128, Sgain=1.188502,Stime=0.000044,
 Framenum=3397 Cur Piris=128, Sgain=1.148154,Stime=0.000044,
 Framenum=3398 Cur Piris=128, Sgain=1.148154,Stime=0.000044,
 Framenum=3399 Cur Piris=128, Sgain=1.230269,Stime=0.000044,
```

如图，可观察到曝光在59ms和44ms之间来回震荡，具体查看59ms和44ms对应的亮度均值会发现，二者的亮度之比与曝光之比差距较大，线性度有问题。第一步需要进行sensor的驱动检测，在驱动打印曝光寄存器值，查看寄存器值是否与log中的曝光一致。上述问题可以使用AE的syncTest功能进行复现，设置多组曝光（包含出现问题的曝光），令其来回切换，查看LOG中每帧曝光对应的亮度变化是否满足线性。

```
rk_aiq_ae_algo.cpp:6405: ================================ HDR-AE (enter)==============================
rk_aiq_ae_algo.cpp:6425: AecRun: SMeanLuma=25.166803, MMeanLuma=85.886871,LMeanLuma=0.000000,TmoMeanluma=35.152767,Isconverged
rk_aiq_ae_algo.cpp:6434: >>> Framenum=3385 Cur Piris=128, Sgain=1.071519,Stime=0.000059,mgain=1.071519,mtime=0.000207,lgain=0.
rk_aiq_ae_algo.cpp:3564: S-HighLightLuma=72.000000,S-Target=100.000000,S-GlobalLuma=25.166803,S-Target=19.998999
rk_aiq_ae_algo.cpp:3909: L-LowLightLuma=56.696957,L-Target=49.991318,L-GlobalLuma=85.886871,L-Target=79.985535
rk_aiq_ae_algo.cpp:5385: AecHdrClmExecute: sgain=1.000000,stime=0.000051,mgain=1.000000,mtime=0.000220,lgain=0.000000,ltime=0.
rk_aiq_ae_algo.cpp:6555: calc result:piris=128,sgain=1.148154,stime=0.000044,mgain=1.071519,mtime=0.000207,lgain=0.000000,ltir
rk_aiq_ae_algo.cpp:6559: ================================ (exit)==============================

rk_aiq_algo_ae_itf.cpp:256: Cur-Exp: FrmId=3386,S-gain=0x7,S-time=0xc,M-gain=0x2,M-time=0x38,L-gain=0x0,L-time=0x0,envChange=1
rk_aiq_algo_ae_itf.cpp:264: Last-Res:FrmId=3385,S-gain=0x4,S-time=0xc,M-gain=0x2,M-time=0x38,L-gain=0x0,L-time=0x0

rk_aiq_ae_algo.cpp:6405: ================================ HDR-AE (enter)==============================
rk_aiq_ae_algo.cpp:6425: AecRun: SMeanLuma=15.018992, MMeanLuma=85.981834,LMeanLuma=0.000000,TmoMeanluma=31.430223,Isconverged
rk_aiq_ae_algo.cpp:6434: >>> Framenum=3386 Cur Piris=128, Sgain=1.273503,Stime=0.000044,mgain=1.071519,mtime=0.000207,lgain=0.
rk_aiq_ae_algo.cpp:3564: S-HighLightLuma=43.000000,S-Target=100.000000,S-GlobalLuma=15.018992,S-Target=19.999107
rk_aiq_ae_algo.cpp:3909: L-LowLightLuma=56.738033,L-Target=49.991318,L-GlobalLuma=85.981834,L-Target=79.985535
rk_aiq_ae_algo.cpp:5385: AecHdrClmExecute: sgain=1.000000,stime=0.000057,mgain=1.000000,mtime=0.000220,lgain=0.000000,ltime=0.
rk_aiq_ae_algo.cpp:6555: calc result:piris=128,sgain=1.273503,stime=0.000044,mgain=1.071519,mtime=0.000207,lgain=0.000000,ltir
rk_aiq_ae_algo.cpp:6559: ================================ (exit)==============================
```

（3）AE后续模块导致的闪烁，常见LOG示例如下（仅截取每帧关键LOG行）：

```
AecRun: SMeanLuma=12.698849, MMeanLuma=240.257675,LMeanLuma=0.000000,TmoMeanluma=104.390663,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=12.944373, MMeanLuma=244.828003,LMeanLuma=0.000000,TmoMeanluma=104.161766,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=12.950768, MMeanLuma=245.728897,LMeanLuma=0.000000,TmoMeanluma=102.967392,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=10.402813, MMeanLuma=222.482101,LMeanLuma=0.000000,TmoMeanluma=79.687981,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=8.134911, MMeanLuma=159.046036,LMeanLuma=0.000000,TmoMeanluma=60.763428,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=6.737852, MMeanLuma=127.505112,LMeanLuma=0.000000,TmoMeanluma=54.783249,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=6.527493, MMeanLuma=123.283249,LMeanLuma=0.000000,TmoMeanluma=54.739769,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=6.310742, MMeanLuma=119.683502,LMeanLuma=0.000000,TmoMeanluma=63.102303,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=5.210998, MMeanLuma=95.413681,LMeanLuma=0.000000,TmoMeanluma=53.315216,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=5.067775, MMeanLuma=86.629799,LMeanLuma=0.000000,TmoMeanluma=49.849743,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=4.646420, MMeanLuma=78.632355,LMeanLuma=0.000000,TmoMeanluma=46.617645,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=4.376598, MMeanLuma=76.865089,LMeanLuma=0.000000,TmoMeanluma=45.372761,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=4.205883, MMeanLuma=74.497444,LMeanLuma=0.000000,TmoMeanluma=43.842072,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=3.946291, MMeanLuma=74.496162,LMeanLuma=0.000000,TmoMeanluma=43.543480,Isconverged=0,Longfrm=0
AecRun: SMeanLuma=3.789642, MMeanLuma=74.514069,LMeanLuma=0.000000,TmoMeanluma=43.231457,Isconverged=0,Longfrm=0
```

从LOG中可知左侧SMeanLuma和MMeanLuma递减的过程中，TmoMeanluma模块输出的亮度发生了突变，发生这种情况时需至TMO模块进行debug。

AWB

概述

AWB模块的功能是通过改变拍摄设备的色彩通道的增益，对色温环境所造成的颜色偏差和拍摄设备本身所固有的色彩通道增益的偏差进行统一补偿，从而让获得的图像能正确反映物体的真实色彩。

重要概念

- 色温：色温是按绝对黑体来定义的， 光源的辐射在可见区和绝对黑体的辐射完全相同时，此时黑体的温度就称此光源的色温。
- 白平衡：在不同色温的光源下，白色在传感器中的响应会偏蓝或偏红。白平衡算法通过调整 R, G, B 三个颜色通道的强度，使白色真实呈现。

功能描述

AWB 模块有WB 信息统计及 AWB 策略控制算法两部分组成。

功能级API参考

rk_aiq_uapi2_setWBMode

【描述】
设置白平衡工作模式。

【语法】

```
XCamReturn rk_aiq_uapi2_setWBMode(const rk_aiq_sys_ctx_t* ctx, opMode_t mode);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
mode	白平衡工作模式	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【注意】

- 若设置为手动模式，白平衡增益值为当前手动白平衡参数控制。若需要切换手动模式的同时设置特定增益值，可以使用rk_aiq_uapi2_setMWBGain接口。

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_getWBMode

【描述】
获取白平衡工作模式。

【语法】

```
XCamReturn rk_aiq_uapi2_getWBMode(const rk_aiq_sys_ctx_t* ctx, opMode_t *mode);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
mode	白平衡工作模式	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_lockAWB

【描述】
锁定当前白平衡参数。

【语法】

```
XCamReturn rk_aiq_uapi2_lockAWB(const rk_aiq_sys_ctx_t* ctx);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_unlockAWB

【描述】
解锁已被锁定的白平衡参数。

【语法】

```
XCamReturn rk_aiq_uapi2_unlockAWB(const rk_aiq_sys_ctx_t* ctx);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_setMWBScene

【描述】
设置白平衡场景。

【语法】

```
XCamReturn rk_aiq_uapi2_setMWBScene(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_wb_scene_t scene);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
scene	白平衡场景	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_getMWBScene

【描述】
获取白平衡场景。

【语法】

```
XCamReturn rk_aiq_uapi2_getMWBScene(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_wb_scene_t *scene);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
scene	白平衡场景	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_setMWBGain

【描述】
设置手动白平衡增益系数。

【语法】

```
XCamReturn rk_aiq_uapi2_setMWBGain(const rk_aiq_sys_ctx_t* ctx, rk_aiq_wb_gain_t *gain);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
gain	白平衡增益系数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_getWBGain

【描述】
获取白平衡增益系数。手动白平衡和自动白平衡均用该函数

【语法】

```
XCamReturn rk_aiq_uapi2_getWBGain(const rk_aiq_sys_ctx_t* ctx, rk_aiq_wb_gain_t *gain);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
gain	白平衡增益系数	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_setMWBCT

【描述】
设置手动白平衡色温参数。

【语法】

```
XCamReturn rk_aiq_uapi2_setMWBCT(const rk_aiq_sys_ctx_t* ctx, unsigned int ct);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
ct	白平衡色温参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_getWBCT

【描述】
获取白平衡色温。自动和手动均用该函数

【语法】

```
XCamReturn rk_aiq_uapi2_getWBCT(const rk_aiq_sys_ctx_t* ctx, unsigned int *ct);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
ct	白平衡色温	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_setAwbGainOffsetAttrib

【描述】
设置自动白平衡gain的偏移。

【语法】

```
XCamReturn rk_aiq_uapi2_setAwbGainOffsetAttrib(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_uapiV2_wb_awb_wbGainOffset_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	wbgain偏移参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_getAwbGainOffsetAttrib

【描述】
获取自动白平衡gain的偏移

【语法】

```
XCamReturn rk_aiq_uapi2_getAwbGainOffsetAttrib(const rk_aiq_sys_ctx_t* ctx,
CalibDbV2_Awb_gain_offset_cfg_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	wbgain偏移参数	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_setAwbGainAdjustAttrib

【描述】
设置自动白平衡模式下的色调调整参数。

【语法】

```
XCamReturn rk_aiq_uapi2_setAwbGainAdjustAttrib(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_uapiV2_wb_awb_wbGainAdjust_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	白平衡色调调整参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_getAwbGainAdjustAttrib

【描述】
获取自动白平衡模式下的色调调整参数。

【语法】

```
XCamReturn rk_aiq_uapi2_getAwbGainAdjustAttrib(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_uapiV2_wb_awb_wbGainAdjust_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	白平衡色调调整参数	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_setAwbV30AllAttrib

【描述】
设置白平衡API支持的全部参数。

【语法】

```
XCamReturn rk_aiq_uapi2_setAwbV30AllAttrib(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_uapiV2_wbV30_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	白平衡API支持的全部参数。	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_uapi2_getAwbV30AllAttrib

【描述】
获取白平衡API支持的全部参数。

【语法】

```
XCamReturn rk_aiq_uapi2_getAwbV30AllAttrib(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_uapiV2_wbV30_attrib_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	wbgain偏移参数	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

功能级API数据类型

rk_aiq_wb_op_mode_t

【说明】
定义白平衡工作模式

【定义】

```
typedef enum rk_aiq_wb_op_mode_s {
    RK_AIQ_WB_MODE_INVALID          = 0,
    RK_AIQ_WB_MODE_MANUAL           = 1,
    RK_AIQ_WB_MODE_AUTO             = 2,
    RK_AIQ_WB_MODE_MAX
} rk_aiq_wb_op_mode_t;
```

【成员】

成员名称	描述
RK_AIQ_WB_MODE_MANUAL	白平衡手动模式
RK_AIQ_WB_MODE_AUTO	白平衡自动模式

rk_aiq_wb_mwb_mode_t

【说明】
定义手动白平衡模式类型

【定义】

```
typedef enum rk_aiq_wb_mwb_mode_e {
    RK_AIQ_MWB_MODE_INVAILD             = 0,
    RK_AIQ_MWB_MODE_CCT                 = 1,
    RK_AIQ_MWB_MODE_WBGAIN              = 2,
    RK_AIQ_MWB_MODE_SCENE               = 3,
} rk_aiq_wb_mwb_mode_t;
```

【成员】

成员名称	描述
RK_AIQ_MWB_MODE_CCT	色温
RK_AIQ_MWB_MODE_WBGAIN	增益系数
RK_AIQ_MWB_MODE_SCENE	场景

rk_aiq_wb_gain_t

【说明】
定义白平衡增益参数

【定义】

```
typedef struct rk_aiq_wb_gain_s {
    float rgain;
    float grgain;
    float gbgain;
    float bgain;
} rk_aiq_wb_gain_t;
```

【成员】

成员名称	描述
rgain	R通道增益
grgain	G通道增益
gbgain	GB通道增益
bgain	B通道增益

rk_aiq_wb_scene_t

【说明】
定义白平衡增益参数

【定义】

```
typedef enum rk_aiq_wb_scene_e {
    RK_AIQ_WBCT_INCANDESCENT = 0,
    RK_AIQ_WBCT_FLUORESCENT,
    RK_AIQ_WBCT_WARM_FLUORESCENT,
    RK_AIQ_WBCT_DAYLIGHT,
    RK_AIQ_WBCT_CLOUDY_DAYLIGHT,
    RK_AIQ_WBCT_TWILIGHT,
    RK_AIQ_WBCT_SHADE
} rk_aiq_wb_scene_t;
```

【成员】

成员名称	描述
RK_AIQ_WBCT_INCANDESCENT	白炽灯
RK_AIQ_WBCT_FLUORESCENT	荧光灯
RK_AIQ_WBCT_WARM_FLUORESCENT	暖荧光灯
RK_AIQ_WBCT_DAYLIGHT	日光
RK_AIQ_WBCT_CLOUDY_DAYLIGHT	阴天
RK_AIQ_WBCT_TWILIGHT	暮光
RK_AIQ_WBCT_SHADE	阴影

rk_aiq_wb_cct_t

【说明】
定义白平衡增益参数

【定义】

```
typedef struct rk_aiq_wb_cct_s {
    float CCT;
    float CCRI;
} rk_aiq_wb_cct_t;
```

【成员】

成员名称	描述
CCT	相关色温
CCRI	相关显色指数

rk_aiq_wb_mwb_attrib_t

【说明】
定义手动白平衡属性

【定义】

```
typedef struct rk_aiq_wb_mwb_attrib_s {
    rk_aiq_wb_mwb_mode_t mode;
    union MWBPara_u {
        rk_aiq_wb_gain_t gain;
        rk_aiq_wb_scene_t scene;
        rk_aiq_wb_cct_t cct;
    } para;
} rk_aiq_wb_mwb_attrib_t;
```

【成员】

成员名称	描述
mode	模式选择
para	模式对应的参数配置

rk_aiq_uapiV2_wb_awb_wbGainOffset_t

【说明】
定义自动白平衡gain偏移

【定义】

```
typedef struct rk_aiq_uapiV2_wb_awb_wbGainOffset_s{
  rk_aiq_uapi_sync_t sync;
  CalibDbV2_Awb_gain_offset_cfg_t gainOffset;
}rk_aiq_uapiV2_wb_awb_wbGainOffset_t;
```

【成员】

成员名称	描述
sync	参考rk_aiq_uapi_sync_t定义说明，参见 **“概述/API说明”** 章节
gainOffset	参靠后面CalibDbV2_Awb_gain_offset_cfg_t描述

CalibDbV2_Awb_gain_offset_cfg_t

【说明】
定义自动白平衡gain偏移

【定义】

```
typedef struct CalibDbV2_Awb_gain_offset_cfg_s{
  bool enable;
  float offset[4];
}CalibDbV2_Awb_gain_offset_cfg_t;
```

【成员】

成员名称	描述
enable	使能开关 取值0或1，分别代表不使能、使能
offset	wbgain与offset相加，对应R GR GB B通道的偏移 取值范围由wbgain与offset相加值确定，wbgain与offset相加后范围在[0,8]（ISP21）

rk_aiq_uapiV2_wb_awb_wbGainAdjust_t

【说明】

定义自动白平衡色调调整参数

【定义】

```
typedef struct rk_aiq_uapiV2_wb_awb_wbGainAdjustLut_s {
  float lumaValue;
  int ct_grid_num;
  int cri_grid_num;
  float ct_in_range[2];//min,max, equal distance sapmle
  float cri_in_range[2];//min,max
  float *ct_lut_out;//size is ct_grid_num*cri_grid_num
  float *cri_lut_out;
} rk_aiq_uapiV2_wb_awb_wbGainAdjustLut_t;

typedef struct rk_aiq_uapiV2_wb_awb_wbGainAdjust_s {
  rk_aiq_uapi_sync_t sync;
  bool enable;
  rk_aiq_uapiV2_wb_awb_wbGainAdjustLut_t *lutAll;
  int lutAll_len;
} rk_aiq_uapiV2_wb_awb_wbGainAdjust_t;
```

【成员】

成员名称	描述
sync	参考rk_aiq_uapi_sync_t定义说明，参见 **“概述/API说明”** 章节
enable	色调调整使能 取值0或1，分别代表不使能、使能
lutAll	不同的环境亮度可以配置不同的输出色温表
lutAll_len	指定色温表的个数
lutAll.lumaValue	环境亮度 取值范围0-255000
lutAll.ct_grid_num	输入色温表之色温的采样点数 取值不限
lutAll.ct_in_range	输入色温表之色温的范围 取值不限
lutAll.cri_grid_num	输入色温表之显色指数的采样点数 取值不限
lutAll.cri_in_range	输入色温表之显色指数的范围 取值不限
lutAll.ct_out	如工具界面图所示每个圆点的ct值，从左到右（色调从冷到暖），即**CT从小到大** 取值范围0-255000
lutAll.cri_out	如工具界面图所示每个圆点的cri，从下到上（色调从紫到绿），即**CRI从负数到正数** 取值范围不限

更多说明参考《Rockchip_Color_Optimization_Guide》文档里面的WBGain色调调整章节

rk_aiq_uapiV2_wbV30_awb_attrib_t

【说明】
定义自动白平衡api参数

【定义】

```
typedef struct rk_aiq_uapiV2_wbV30_awb_attrib_s {
    rk_aiq_uapiV2_wb_awb_wbGainAdjust_t wbGainAdjust;
    CalibDbV2_Awb_gain_offset_cfg_t wbGainOffset;
} rk_aiq_uapiV2_wbV30_awb_attrib_t;
```

【成员】

成员名称	描述
stAuto.wbGainAdjust	参考前述CalibDbV2_Awb_gain_offset_cfg_t
stAuto.wbGainOffset	参考前述rk_aiq_uapiV2_wb_awb_wbGainAdjust_t

rk_aiq_uapiV2_wbV30_attrib_t

【说明】
定义白平衡API支持的全部参数。

【定义】

```
typedef struct rk_aiq_uapiV2_wbV30_attrib_s {
    rk_aiq_uapi_sync_t sync;
    bool byPass;
    rk_aiq_wb_op_mode_t mode;
    rk_aiq_wb_mwb_attrib_t stManual;
    rk_aiq_uapiV2_wbV30_awb_attrib_t stAuto;
} rk_aiq_uapiV2_wbV30_attrib_t;
```

【成员】

成员名称	描述
sync	参考rk_aiq_uapi_sync_t说明，参见 **“概述/API说明”** 章节
bypass	取值0或1 0表示做白平衡校正，使用的白平衡增益由表格后面的参数控制控制 1表示不执行白平衡校正
mode	自动或手动白平衡模式控制参数，参考前述rk_aiq_wb_op_mode_t
stManual	手动白平衡参数，参考前述rk_aiq_wb_mwb_attrib_t
stAuto	自动白平衡参数，参考前述rk_aiq_uapiV2_wbV21_awb_attrib_t

模块级API参考

rk_aiq_user_api2_awbV30_SetAllAttrib

【描述】
设置白平衡API支持的全部参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_awbV30_SetAllAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_uapiV2_wbV30_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	白平衡API支持的全部参数	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awbV30_GetAllAttrib

【描述】
获取白平衡API支持的全部参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_awbV30_GetAllAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_uapiV2_wbV30_attrib_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	白平衡API支持的全部参数	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awb_GetCCT

【描述】
获取白平衡色温。自动和手动均用该函数

【语法】

```
XCamReturn
rk_aiq_user_api2_awb_GetCCT(const rk_aiq_sys_ctx_t* sys_ctx, rk_aiq_wb_cct_t
*cct);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
cct	白平衡的色温参数	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awb_QueryWBInfo

【描述】
获取白平衡增益系数，色温。

【语法】

```
XCamReturn
rk_aiq_user_api2_awb_QueryWBInfo(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_wb_querry_info_t *wb_querry_info);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
wb_querry_info	颜色相关状态参数	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awb_Lock

【描述】
锁定当前白平衡参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_awb_Lock(const rk_aiq_sys_ctx_t* sys_ctx);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awb_Unlock

【描述】
解锁已被锁定的白平衡参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_awb_Lock(const rk_aiq_sys_ctx_t* sys_ctx);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awb_SetWpModeAttrib

【描述】
设置白平衡工作模式。

【语法】

```
XCamReturn
rk_aiq_user_api2_awb_SetWpModeAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_uapiV2_wb_opMode_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	白平衡工作模式	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awb_GetWpModeAttrib

【描述】
获取白平衡工作模式。

【语法】

```
XCamReturn
rk_aiq_user_api2_awb_GetWpModeAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_uapiV2_wb_opMode_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	白平衡工作模式	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awb_SetMwbAttrib

【描述】
设置手动白平衡参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_awb_SetMwbAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_wb_mwb_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	手动白平衡参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awb_GetMwbAttrib

【描述】
获取手动白平衡参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_awb_GetMwbAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_wb_mwb_attrib_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	手动白平衡参数	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awb_SetWbGainAdjustAttrib

【描述】
设置自动白平衡模式下的色调调整参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_awb_SetWbGainAdjustAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_uapiV2_wb_awb_wbGainAdjust_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	色调调整参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awb_GetWbGainAdjustAttrib

【描述】
获取自动白平衡模式下的色调调整参数。

【语法】

```
XCamReturn
rk_aiq_user_api2_awb_GetWbGainAdjustAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_uapiV2_wb_awb_wbGainAdjust_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	色调调整参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awb_SetWbGainOffsetAttrib

【描述】
设置自动白平衡gain的偏移

【语法】

```
XCamReturn
rk_aiq_user_api2_awb_SetWbGainOffsetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_uapiV2_wb_awb_wbGainOffset_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	色调调整参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

rk_aiq_user_api2_awb_GetWbGainOffsetAttrib

【描述】
获取自动白平衡gain的偏移

【语法】

```
XCamReturn
rk_aiq_user_api2_awb_GetWbGainOffsetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_uapiV2_wb_awb_wbGainOffset_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	色调调整参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_awb.h、rk_aiq_uapi2_awb_int.h
- 库文件：librkaiq.so

【示例】

- 参考sample_awb_module.cpp

模块级API数据类型

rk_aiq_wb_querry_info_t

【说明】
定义白平衡查询信息

【定义】

```
typedef struct rk_aiq_wb_querry_info_s {
    rk_aiq_wb_gain_t gain;
    rk_aiq_wb_cct_t cctGloabl;
    bool awbConverged;
    uint32_t LVValue;
} rk_aiq_wb_querry_info_t;
```

【成员】

成员名称	描述
gain	增益
cctGloabl	全局色温参数
awbConverged	白平衡是否收敛
LVValue	相关环境亮度

rk_aiq_wb_op_mode_t

【说明】
定义白平衡工作模式

【定义】

```
typedef struct rk_aiq_uapiV2_wb_opMode_s {
  rk_aiq_uapi_sync_t sync;
  rk_aiq_wb_op_mode_t mode;
} rk_aiq_uapiV2_wb_opMode_t;
```

【成员】

成员名称	描述
sync	参考rk_aiq_uapi_sync_t说明，参见 **“概述/API说明”** 章节
mode	参考rk_aiq_wb_op_mode_t说明

rk_aiq_wb_mwb_attrib_t

【说明】
定义手动白平衡属性

【定义】

```
typedef struct rk_aiq_wb_mwb_attrib_s {
     rk_aiq_uapi_sync_t sync;
    rk_aiq_wb_mwb_mode_t mode;
    union MWBPara_u {
        rk_aiq_wb_gain_t gain;
        rk_aiq_wb_scene_t scene;
        rk_aiq_wb_cct_t cct;
    } para;
} rk_aiq_wb_mwb_attrib_t;
```

【成员】

成员名称	描述
sync	参考rk_aiq_uapi_sync_t定义，参见 **“概述/API说明”** 章节
mode	模式选择
para	模式对应的参数配置

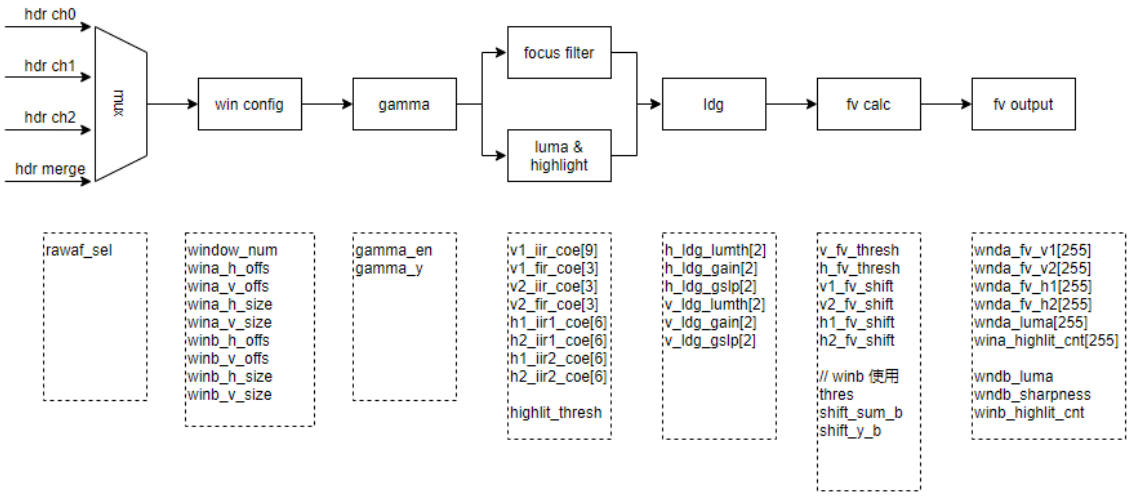
AF

概述

AF模块的功能是指调整相机镜头，使被拍物成像清晰的过程。

功能描述

AF 模块由AF 信息统计及AF控制算法两部分组成。
下图为AF信息统计模块框图

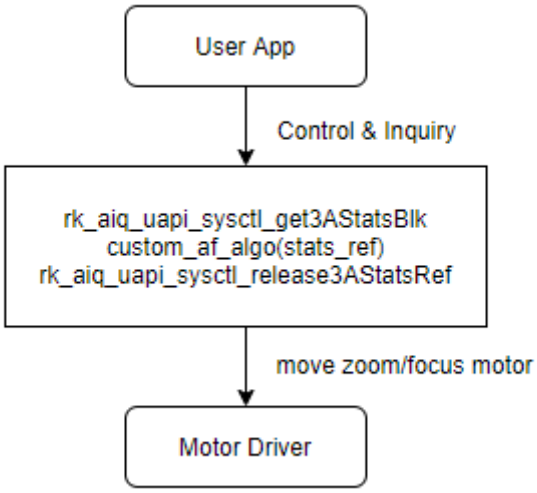


其中，ch0/ch1/ch2对应hdr模式下S/M/L帧，debayer为hdr模式下的合成帧。

windowA的输出主要包含15*15的V1/V2/H1/H2 FV信息、亮度信息和高亮计数。

windowB的输出主要包含一个FV信息、亮度信息和高亮计数。

开发用户AF算法



参考代码位置：sdk: external/camera_engine_rkaiq/rkisp_demo/demo/af_algo_demo

功能级API参考

rk_aiq_uapi2_setFocusMode

【描述】 配置对焦模式。

【语法】

```
XCamReturn rk_aiq_uapi2_setFocusMode(const rk_aiq_sys_ctx_t* ctx, opMode_t
mode);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
mode	对焦模式	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getFocusMode

【描述】 获取当前对焦模式。

【语法】

```
XCamReturn rk_aiq_uapi2_getFocusMode(const rk_aiq_sys_ctx_t* ctx, opMode_t
*mode);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
mode	对焦模式	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setFocusWin

【描述】 设置自动对焦窗口。

【语法】

```
XCamReturn rk_aiq_uapi2_setFocusWin(const rk_aiq_sys_ctx_t* ctx, paRect_t
*rect);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
rect	对焦窗口，取值范围由sensor输入大小确定	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getFocusWin

【描述】 设置自动对焦窗口。

【语法】

```
XCamReturn rk_aiq_uapi2_getFocusWin(const rk_aiq_sys_ctx_t* ctx, paRect_t
*rect);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
rect	对焦窗口	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_lockFocus

【描述】 锁定自动对焦，对焦暂停动作。

【语法】

```
XCamReturn rk_aiq_uapi2_lockFocus(const rk_aiq_sys_ctx_t* ctx);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_unlockFocus

【描述】 解锁自动对焦，对焦继续动作。

【语法】

```
XCamReturn rk_aiq_uapi2_unlockFocus(const rk_aiq_sys_ctx_t* ctx);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_oneshotFocus

【描述】 触发单次对焦，对焦完成后，不会继续对焦。

【语法】

```
XCamReturn rk_aiq_uapi2_oneshotFocus(const rk_aiq_sys_ctx_t* ctx);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_manualTrigerFocus

【描述】 手动触发对焦，对焦完成后，继续监视画面是否模糊，如果画面模糊，再次执行对焦。

【语法】

```
XCamReturn rk_aiq_uapi2_manualTrigerFocus(const rk_aiq_sys_ctx_t* ctx);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_trackingFocus

【描述】 继续监视画面是否模糊，如果画面模糊，再次执行对焦，一般执行rk_aiq_uapi2_oneshotFocus后使用。

【语法】

```
XCamReturn rk_aiq_uapi2_trackingFocus(const rk_aiq_sys_ctx_t* ctx);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getSearchResult

【描述】 获取对焦结果。

【语法】

```
XCamReturn rk_aiq_uapi2_getSearchResult(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_af_result_t* result);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
result	对焦结果，具体参考结构体rk_aiq_af_result_t的说明	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getZoomRange

【描述】 获取变焦范围值，用于限制rk_aiq_uapi_setOpZoomPosition输入的zoom code参数。

【语法】

```
XCamReturn rk_aiq_uapi2_getZoomRange(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_af_zoomrange* range);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
range	zoom可移动范围，具体参考结构体rk_aiq_af_zoomrange	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setOpZoomPosition

【描述】 设置zoom位置，修改变焦位置。

【语法】

```
XCamReturn rk_aiq_uapi2_setOpZoomPosition(const rk_aiq_sys_ctx_t* ctx, int pos);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
pos	zoom position，取值范围由rk_aiq_uapi2_getZoomRange确定	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getOpZoomPosition

【描述】 获取zoom位置。

【语法】

```
XCamReturn rk_aiq_uapi2_getOpZoomPosition(const rk_aiq_sys_ctx_t* ctx, int
*pos);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
pos	zoom position	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_endOpZoomChange

【描述】 结束zoom设备位置的设置，在rk_aiq_uapi_setOpZoomPosition之后调用，被调用后执行聚焦动作。

【语法】

```
XCamReturn rk_aiq_uapi2_endOpZoomChange(const rk_aiq_sys_ctx_t* ctx);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getFocusRange

【描述】 获取聚焦范围值，用于限制rk_aiq_uapi_setFocusPosition输入的focus code参数。

【语法】

```
XCamReturn rk_aiq_uapi2_getFocusRange(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_af_focusrange* range);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
range	focus可移动范围，具体参考结构体rk_aiq_af_focusrange	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setFocusPosition

【描述】 设置手动对焦模式下的对焦位置。

【语法】

```
XCamReturn rk_aiq_uapi2_setFocusPosition(const rk_aiq_sys_ctx_t* ctx, unsigned
short code);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
code	对焦code值，取值范围由rk_aiq_uapi2_getFocusRange确定	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h

- 库文件：librkaiq.so

rk_aiq_uapi2_getFocusPosition

【描述】 获取当前对焦位置。

【语法】

```
XCamReturn rk_aiq_uapi2_getFocusPosition(const rk_aiq_sys_ctx_t* ctx, unsigned short *code);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
code	对焦code值	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_startZoomCalib

【描述】 执行电动马达模组校正。

【语法】

```
XCamReturn rk_aiq_uapi2_startZoomCalib(const rk_aiq_sys_ctx_t* ctx);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_resetZoom

【描述】 执行电动马达模组复位。

【语法】

```
XCamReturn rk_aiq_uapi2_resetZoom(const rk_aiq_sys_ctx_t* ctx);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

功能级API数据类型

opMode_t

【说明】
定义AF信息统计工作模式

【定义】

```
typedef enum opMode_e {
    OP_AUTO   = 0,
    OP_MANUAL = 1,
    OP_SEMI_AUTO = 2,
    OP_INVAL
} opMode_t;
```

【成员】

成员名称	描述
OP_AUTO	自动对焦模式，运行内置AF算法
OP_MANUAL	手动对焦模式，停止内置AF算法
OP_SEMI_AUTO	半自动对焦模式，set zoom position api调用后执行一次自动对焦

rk_aiq_af_zoomrange

【说明】 定义zoom取值范围

【定义】

```
typedef struct {
    int min_pos;
    int max_pos;
    float min_fl;
    float max_fl;
} rk_aiq_af_zoomrange;
```

【成员】

成员名称	描述
min_pos	zoom最小值
max_pos	zoom最大值
min_fl	最小焦距
max_fl	最大焦距

rk_aiq_af_focusrange

【说明】 定义focus取值范围

【定义】

```
typedef struct {
    int min_pos;
    int max_pos;
} rk_aiq_af_focusrange;
```

【成员】

成员名称	描述
min_pos	focus最小值
max_pos	focus最大值

rk_aiq_af_result_t

【说明】 定义af搜索结果

【定义】

```
typedef enum rk_aiq_af_sec_stat_e
{
    RK_AIQ_AF_SEARCH_INVAL   = 0,
    RK_AIQ_AF_SEARCH_RUNNING = 1,
    RK_AIQ_AF_SEARCH_END     = 2
} rk_aiq_af_sec_stat_t;

typedef struct {
    rk_aiq_af_sec_stat_t stat;
    int32_t final_pos;
} rk_aiq_af_result_t;
```

【成员】

成员名称	描述
stat	对焦状态
final_pos	最终focus位置

模块级API参考

rk_aiq_user_api2_af_SetAttrib

【描述】
设置对焦属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_af_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_af_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	对焦的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_af.h
- 库文件：librkaiq.so

rk_aiq_user_api2_af_GetAttrib

【描述】
获取对焦属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_af_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_af_attrib_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	对焦的参数属性	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_af.h
- 库文件：librkaiq.so

模块级API数据类型

RKAIQ_AF_MODE

【说明】
定义对焦工作模式

【定义】

```
typedef enum _RKAIQ_AF_MODE
{
    RKAIQ_AF_MODE_NOT_SET = -1,
    RKAIQ_AF_MODE_AUTO,
    RKAIQ_AF_MODE_MACRO,
    RKAIQ_AF_MODE_INFINITY,
    RKAIQ_AF_MODE_FIXED,
    RKAIQ_AF_MODE_EDOF,
    RKAIQ_AF_MODE_CONTINUOUS_VIDEO,
    RKAIQ_AF_MODE_CONTINUOUS_PICTURE,
    RKAIQ_AF_MODE_ONESHOT_AFTER_ZOOM,
} RKAIQ_AF_MODE;
```

【成员】

成员名称	描述
RKAIQ_AF_MODE_NOT_SET	对焦模式未设置
RKAIQ_AF_MODE_AUTO	自动对焦模式
RKAIQ_AF_MODE_MACRO	微距对焦模式
RKAIQ_AF_MODE_INFINITY	远距对焦模式
RKAIQ_AF_MODE_FIXED	固定对焦模式
RKAIQ_AF_MODE_EDOF	景深对焦模式
RKAIQ_AF_MODE_CONTINUOUS_VIDEO	平滑持续对焦模式
RKAIQ_AF_MODE_CONTINUOUS_PICTURE	快速持续对焦模式
RKAIQ_AF_MODE_ONESHOT_AFTER_ZOOM	半自动对焦模式，set zoom position调用后，自动触发一次对焦

RKAIQ_AF_HWVER

【说明】
定义对焦工作模式

【定义】

```
typedef enum _RKAIQ_AF_HWVER
{
    RKAIQ_AF_HW_V20 = 0,
    RKAIQ_AF_HW_V30,
    RKAIQ_AF_HW_VMAX
} RKAIQ_AF_HWVER;
```

【成员】

成员名称	描述
RKAIQ_AF_HW_V20	AF硬件版本 2.0
RKAIQ_AF_HW_V30	AF硬件版本 3.0

rk_aiq_af_algo_meas_v30_t

【说明】
定义AF信息统计工作模式

【定义】

```
typedef struct {
    unsigned char af_en;
    unsigned char rawaf_sel;
    unsigned char gamma_en;
    unsigned char gaus_en;
    unsigned char v1_fir_sel;
    unsigned char hiir_en;
```

```
    unsigned char viir_en;
    unsigned char v1_fv_outmode;    // 0 square, 1 absolute
    unsigned char v2_fv_outmode;    // 0 square, 1 absolute
    unsigned char h1_fv_outmode;    // 0 square, 1 absolute
    unsigned char h2_fv_outmode;    // 0 square, 1 absolute
    unsigned char ldg_en;
    unsigned char accu_8bit_mode;   // fix to 1
    unsigned char ae_mode;
    unsigned char y_mode;           // fix to 0
    unsigned char line_en[RKAIQ_RAWAF_LINE_NUM];
    unsigned char line_num[RKAIQ_RAWAF_LINE_NUM];

    unsigned char window_num;
    unsigned short wina_h_offs;
    unsigned short wina_v_offs;
    unsigned short wina_h_size;
    unsigned short wina_v_size;
    unsigned short winb_h_offs;
    unsigned short winb_v_offs;
    unsigned short winb_h_size;
    unsigned short winb_v_size;

    unsigned short gamma_y[RKAIQ_RAWAF_GAMMA_NUM];

    // [old version param]
    unsigned short thres;
    unsigned char shift_sum_a;
    unsigned char shift_sum_b;
    unsigned char shift_y_a;
    unsigned char shift_y_b;

    /**********[Vertical IIR (v1 & v2)]************/
    short v1_iir_coe[9];
    short v1_fir_coe[3];
    short v2_iir_coe[3];
    short v2_fir_coe[3];

    /**********[Horizontal IIR (h1 & h2)]************/
    short h1_iir1_coe[6];
    short h2_iir1_coe[6];
    short h1_iir2_coe[6];
    short h2_iir2_coe[6];

    /**********[Focus value statistic param]**********/
    // level depended gain
    // input8 lumi, output8bit gain
    unsigned char h_ldg_lumth[2];    //luminance thresh
    unsigned char h_ldg_gain[2];     //gain for [minLum,maxLum]
    unsigned short h_ldg_gslp[2];    //[slope_low,-slope_high]
    unsigned char v_ldg_lumth[2];
    unsigned char v_ldg_gain[2];
    unsigned short v_ldg_gslp[2];

    // coring
    unsigned short v_fv_thresh;
    unsigned short h_fv_thresh;

    // left shift, more needed if outmode=square
```

```
    unsigned char v1_fv_shift; //only for sel1
    unsigned char v2_fv_shift;
    unsigned char h1_fv_shift;
    unsigned char h2_fv_shift;

    /**********[High light]**********/
    unsigned short highlit_thresh;
}  rk_aiq_af_algo_meas_v30_t;
```

【成员】

成员名称	描述
af_en	是否使能AF 信息统计，0为关闭，1为打开
rawaf_sel	选择AF信息统计的通道，取值范围0-3，对应hdr模式的长/中/短/合成帧通道选择，一般AF选择中帧通道，非hdr模式设置为0，hdr模式设置为1
gamma_en	gamma模块使能开关，0为关闭，1为打开
gaus_en	需固定设置为1
v1_fir_sel	需固定设置为1
hiir_en	H1/H2通道使能开关，0为关闭，1为打开
viir_en	V1/V2通道使能开关，0为关闭，1为打开。需要注意gamma_en打开时，viir_en必须设置为1
v1_fv_outmode	V1通道FV输出模式选择，0为平方模式，1为绝对值模式
v2_fv_outmode	V2通道FV输出模式选择，0为平方模式，1为绝对值模式
h1_fv_outmode	H1通道FV输出模式选择，0为平方模式，1为绝对值模式
h2_fv_outmode	H2通道FV输出模式选择，0为平方模式，1为绝对值模式
ldg_en	LDG功能使能开关，0为关闭，1为打开
accu_8bit_mode	需固定设置为1
ae_mode	当ae_mode设置为1，RAWAF使能15x15亮度均值统计，复用了RAWAE_BIG模块的逻辑
y_mode	需固定设置为0
line_en	目前暂未生效
line_num	目前暂未生效
window_num	生效的窗口数，window_num为1时，wina(主窗口)生效；window_num为2时，wina(主窗口)和winb(独立窗口)生效
wina_h_offs	wina(主窗口)左上角第一个像素的水平坐标，该值必须大于等于2
wina_v_offs	wina(主窗口)左上角第一个像素的垂直坐标，该值必须大于等于1
wina_h_size	wina(主窗口)的窗口宽度，该值必须小于图像宽度-2-wina_h_offs；同时该值必须为15的倍数；
wina_v_size	wina(主窗口)的窗口高度，该值必须小于图像高度-2-wina_v_offs；同时该值必须为15的倍数；
winb_h_offs	winb(独立窗口)左上角第一个像素的水平坐标，该值必须大于等于2
winb_v_offs	winb(独立窗口)左上角第一个像素的垂直坐标，该值必须大于等于1
winb_h_size	winb(独立窗口)的窗口宽度，该值必须小于图像宽度-2-wina_h_offs
winb_v_size	winb(独立窗口)的窗口高度，该值必须小于图像高度-2-wina_v_offs

成员名称	描述
gamma_y	gamma table的y值，取值范围0-1023；x坐标分段为0 to 1023: 16 16 16 16 32 32 32 32 64 64 64 128 128 128 128 128
thres	win b(独立窗口)的AF统计阈值，计算出的fv值小于该值时，fv值改为0，可减少噪声的影响，取值范围为0-0xFFFF
shift_sum_a	目前无法使用，固定设置为0即可
shift_sum_b	win b(独立窗口)的fv值的shit bit值，会按照该值将fv值向右移位，避免得到的fv值溢出，取值范围为0-7
shift_y_a	目前无法使用，固定设置为0即可
shift_y_b	win b(独立窗口)的luma值的shit bit值，会按照该值将luma值向右移位，避免得到的luma值溢出，取值范围为0-7
v1_iir_coe[9]	用于V1通道的3X3 IIR系数，按照AF滤波器系数生成工具的输出进行设置
v1_fir_coe[3]	用于V1通道的1x3 FIR系数，按照AF滤波器系数生成工具的输出进行设置
v2_iir_coe[3]	用于V2通道的1x3 IIR系数，按照AF滤波器系数生成工具的输出进行设置
v2_fir_coe[3]	用于V2通道的1x3 FIR系数，按照AF滤波器系数生成工具的输出进行设置
h1_iir1_coe[6]	用于H1通道的1X6 IIR1系数，按照AF滤波器系数生成工具的输出进行设置
h2_iir1_coe[6]	用于H2通道的1X6 IIR1系数，按照AF滤波器系数生成工具的输出进行设置
h1_iir2_coe[6]	用于H1通道的1X6 IIR2系数，按照AF滤波器系数生成工具的输出进行设置
h2_iir2_coe[6]	用于H2通道的1X6 IIR2系数，按照AF滤波器系数生成工具的输出进行设置
h_ldg_lumth[2]	用于H1/H2通道的ldg模块的亮度阈值系数，0为左边暗区设置，1为右边高亮区设置，取值范围为0~255
h_ldg_gain[2]	用于H1/H2通道的ldg模块的最小gain值，0为左边暗区设置，1为右边高亮区设置，取值范围为0~255
h_ldg_gslp[2]	用于H1/H2通道的ldg模块的斜率系数，0为左边暗区设置，1为右边高亮区设置，取值范围为0~65535
v_ldg_lumth[2]	用于V1/V2通道的ldg模块的亮度阈值系数，0为左边暗区设置，1为右边高亮区设置，取值范围为0~255
v_ldg_gain[2]	用于V1/V2通道的ldg模块的最小gain值，0为左边暗区设置，1为右边高亮区设置，取值范围为0~255
v_ldg_gslp[2]	用于V1/V2通道的ldg模块的最小gain值，0为左边暗区设置，1为右边高亮区设置，取值范围为0~255
v_fv_thresh	用于V1/V2通道的AF统计阈值，计算出的fv值小于该值时，fv值改为0，可减少噪声的影响，取值范围为0-0x0FFF
h_fv_thresh	用于H1/H2通道的AF统计阈值，计算出的fv值小于该值时，fv值改为0，可减少噪声的影响，取值范围为0-0x0FFF

成员名称	描述
v1_fv_shift	用于V1通道的fv值的shit bit值，会按照该值将fv值向右移位，避免得到的fv值溢出，取值范围为0-7
v2_fv_shift	用于V2通道的fv值的shit bit值，会按照该值将fv值向右移位，避免得到的fv值溢出，取值范围为0-7
h1_fv_shift	用于H1通道的fv值的shit bit值，会按照该值将fv值向右移位，避免得到的fv值溢出，取值范围为0-7
h2_fv_shift	用于H2通道的fv值的shit bit值，会按照该值将fv值向右移位，避免得到的fv值溢出，取值范围为0-7
highlit_thresh	表示高亮统计的阈值，当高于该值则认为是高亮点，纳入统计，只累加每个区域的高亮点的个数，取值范围为0-0x0FFF

rk_aiq_af_attrib_t

【说明】
对焦配置信息

【定义】

```
typedef struct rk_aiq_af_attrib_s {
    rk_aiq_uapi_sync_t sync;

    RKAIQ_AF_MODE AfMode;
    RKAIQ_AF_HWVER AfHwVer;

    bool contrast_af;
    bool laser_af;
    bool pdaf;

    int h_offs;
    int v_offs;
    unsigned int h_size;
    unsigned int v_size;

    short fixedModeDefCode;
    short macroModeDefCode;
    short infinityModeDefCode;

    union {
        rk_aiq_af_algo_meas_v20_t manual_meascfg;
        rk_aiq_af_algo_meas_v30_t manual_meascfg_v30;
    };
} rk_aiq_af_attrib_t;
```

【成员】

成员名称	描述
sync	同步异步API相关信息，参考rk_aiq_uapi_sync_t定义，参见 **“概述/API说明”** 章节
AfMode	对焦模式
contrast_af	使能反差对焦
laser_af	使能激光对焦
pdaf	使能相位对焦
h_offs	对焦窗口起始水平坐标
v_offs	对焦窗口起始垂直坐标
h_size	对焦窗口宽度
v_size	对焦窗口高度
fixedModeDefCode	固定对焦模式下对焦code值
macroModeDefCode	微距对焦模式下终止code值，对焦范围为0-该值
infinityModeDefCode	远距距对焦模式下起始code值，对焦范围为该值-64
manual_meascfg	AF2.0 自定义对焦统计信息配置
manual_meascfg_v30	AF3.0 自定义对焦统计信息配置

其它说明

VCM马达模组驱动验证

1) vcm驱动的起动电流、终止电流是否设置正确，

首先要从模组厂获取起动电流、终止电流的相关信息。

其次选取几个模组，确认这些信息是否正确，方法如下：

dts中将起始电流，终止电流设置为VCM可支持的最大范围。

对焦模式切换为手动模式，从64开始逐步调整vcm position，当lens开始移动，远焦物体(10米以上)清晰时，记录当前的position值，

起始电流为(vcm_max_mA - vcm_min_mA) * (64 - curPos) / 64 。

继续调整vcm位置，当近焦物体(10cm或20cm)清晰时，记录当前的position值，

终止电流为(vcm_max_mA - vcm_min_mA) * (64 - curPos) / 64。

2) 不同方向移动vcm，最终停留位置是否稳定。

对焦模式切换为手动模式，选取一个position，从0移动到该位置和从64移动到该位置，比较两次的图像清晰程度是否一致，AF 统计值　是否接近。

3. 移动镜头所需要的时间是否正确。

电动马达模组驱动验证

1) 多次单步移动和一次多步移动最终停留位置是否相同
 1）首先选定一个zoom/focus马达位置，使图像达到最清晰的状态，记录当前马达位置，并抓取一张图像A；
 2）让zoom或focus马达后退一定的步数，然后单步移动zoom或focus马达，直到到达记录的zoom/focus马达位置，抓取一张图像B；
 3）让zoom或focus马达再次后退一定的步数，一次移动zoom或focus马达，到达记录的zoom/focus马达位置，抓取一张图像C；
 4）比较图像A、图像B和图像C的清晰程度是否一致，视野范围是否一致；
2) 马达随机移动最终停留位置是否相同
 1）首先选定一个zoom/focus马达位置，使图像达到最清晰的状态，记录当前马达位置，并抓取一张图像A；
 2）其次通过脚本让zoom或focus马达随机移动，移动400到500次后，回到最初的zoom/focus马达位置，抓取一张图像B；
 3）比较图像A与图像B，判断清晰程度是否一致，视野范围是否一致；

3) 马达同时移动最终停留位置是否相同
 1）首先选定一个zoom/focus马达位置，使图像达到最清晰的状态，记录当前马达位置，并抓取一张图像A；
 2）其次通过脚本让zoom和focus马达同时进行随机移动，移动400到500次后，回到最初的zoom/focus马达位置，抓取一张图像B；
 3）比较图像A与图像B，判断清晰程度是否一致，视野范围是否一致；

IMGPROC

概述

imgproc 是指影响图像效果的模块。

Merge

功能描述

Merge是将多帧图像合成为一帧的模块。

重要概念

- 在且仅在HDR模式下生效。

功能级API参考

模块级API参考

rk_aiq_user_api2_amerge_SetAttrib

【描述】
设置merge属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_amerge_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
amerge_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	merge的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_amerge.h
- 库文件：librkaiq.so

rk_aiq_user_api2_amerge_GetAttrib

【描述】
获取merge属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_amerge_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
amerge_attrib_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	merge的参数属性	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_amerge.h
- 库文件：librkaiq.so

模块级API数据类型

merge_OpMode_t

【说明】
定义merge工作模式

【定义】

```
typedef enum merge_OpModeV21_e {
    MERGE_OPMODE_API_OFF = 0,
    MERGE_OPMODE_MANU    = 1,
} merge_OpModeV21_t;
```

【成员】

成员名称	描述
MERGE_OPMODE_API_OFF	api关闭模式，此时算法处于AUTO模式，使用Json文件中参数
MERGE_OPMODE_MANU	手动模式

MergeBaseFrame_t

【说明】
定义融合时基准帧属性

【定义】

```
typedef enum MergeBaseFrame_e {
    BASEFRAME_LONG        = 0,
    BASEFRAME_SHORT       = 1,
} MergeBaseFrame_t;
```

【成员】

成员名称	描述
BASEFRAME_LONG	融合时，以长帧为基准
BASEFRAME_SHORT	融合时，以短帧为基准

MergeCurrCtlData_t

【说明】
定义Merge控制参数属性

【定义】

```
typedef struct MergeCurrCtlData_s {
    float Envlv;
    float MoveCoef;
} MergeCurrCtlData_t;
```

【成员】

成员名称	描述
Envlv	当前环境亮度，取值范围：[0,1]，精度0.000001
MoveCoef	当前运动系数，取值范围：[0,1]，精度0.000001，本值暂时为定值。

mMergeOECurveV21_t

【说明】
定义手动Merge过曝曲线属性

【定义】

```
typedef struct mMergeOECurveV21_s {
    float Smooth;
    float Offset;
} mMergeOECurveV21_t;
```

【成员】

成员名称	描述
Smooth	过曝曲线的斜率，取值范围[0,1]，默认值为0.4，精度0.01。
Offset	过曝曲线的偏移值，取值范围[108,280]，默认值为210，精度0.1。

mMergeMDCurveV21_t

【说明】
定义长帧模式下运动曲线属性

【定义】

```
typedef struct mMergeMDCurveV21_s {
    float LM_smooth;
    float LM_offset;
    float MS_smooth;
    float MS_offset;
} mMergeMDCurveV21_t;
```

【成员】

成员名称	描述
LM_smooth	长帧和中帧之间运动曲线斜率，取值范围为[0,1]，默认值为0.4，精度0.01。在RK356x平台下，该值无效。
LM_offset	长帧和中帧之间运动曲线偏移值，取值范围为[0.26,1]，默认值为0.38，精度0.01。在RK356x平台下，该值无效。
MS_smooth	中帧和短帧之间运动曲线斜率，取值范围为[0,1]，默认值为0.4，精度0.01。
MS_offset	中帧和短帧之间运动曲线偏移值，取值范围为[0.26,1]，默认值为0.38，精度0.01。

mMergeAttrV21_t

【说明】
定义RK3588中手动merge属性

【定义】

```
typedef struct mMergeAttrV21_s {
    mMergeOECurveV21_t OECurve;
    mMergeMDCurveV21_t MDCurve;
} mMergeAttrV21_t;
```

【成员】

成员名称	描述
OECurve	过曝曲线参数
MDCurve	运动曲线参数

mergeAttrV21_t

【说明】
RK356x芯片下merge属性配置

【定义】

```
typedef struct mergeAttrV21_s {
    merge_OpModeV21_t    opMode;
    mMergeAttrV21_t      stManual;
    MergeCurrCtlData_t   CtlInfo;
} mergeAttrV21_t;
```

【成员】

成员名称	描述
opMode	模式选择
stManual	手动merge属性
CtlInfo	控制量参数

mLongFrameModeData_t

【说明】
定义长帧模式下，控制参数属性

【定义】

```
typedef struct mLongFrameModeData_s {
    mMergeOECurveV21_t OECurve;
    mMergeMDCurveV21_t MDCurve;
} mLongFrameModeData_t;
```

【成员】

成员名称	描述
OECurve	过曝曲线参数
MDCurve	运动曲线参数

mMergeMDCurveV30Short_t

【说明】
定义短帧模式下，运动曲线属性

【定义】

```
typedef struct mMergeMDCurveV30Short_s{
    float Coef;
    float ms_thd0;
    float lm_thd0;
} mMergeMDCurveV30Short_t;
```

【成员】

成员名称	描述
Coef	控制系数，取值范围[0,1]，默认值为0.05，精度0.0001。
ms_thd0	中短帧控制系数，取值范围[0,1]，默认值为0.0，精度0.1。
lm_thd0	长中帧控制系数，取值范围[0,1]，默认值为0.0，精度0.1。

mShortFrameModeData_t

【说明】
定义短帧模式下，控制参数属性

【定义】

```
typedef struct mShortFrameModeData_s {
    mMergeOECurveV21_t      OECurve;
    mMergeMDCurveV30Short_t MDCurve;
} mShortFrameModeData_t;
```

【成员】

成员名称	描述
OECurve	过曝曲线参数
MDCurve	运动曲线参数

mMergeAttrV30_t

【说明】
定义RK3588中手动merge属性

【定义】

```
typedef struct mMergeAttrV30_s {
    MergeBaseFrame_t      BaseFrm;
    mLongFrameModeData_t  LongFrmModeData;
    mShortFrameModeData_t ShortFrmModeData;
} mMergeAttrV30_t;
```

【成员】

成员名称	描述
BaseFrm	融合基准帧
LongFrmModeData	基准值为长帧时，控制数据数据
ShortFrmModeData	基准值为短帧时，控制数据数据

mergeAttrV30_t

【说明】
RK3588芯片下merge属性配置

【定义】

```
typedef struct mergeAttrV30_s {
    merge_OpModeV21_t    opMode;
    mMergeAttrV30_t      stManual;
    MergeCurrCtlData_t   CtlInfo;
} mergeAttrV30_t;
```

【成员】

成员名称	描述
opMode	模式选择
stManual	手动merge属性
CtlInfo	控制量参数

mergeAttr_t

【说明】
merge属性配置

【定义】

```
typedef struct mergeAttr_s {
    rk_aiq_uapi_sync_t  sync;
    mergeAttrV21_t      attrV21;
    mergeAttrV30_t      attrV30;
} mergeAttr_t;
```

【成员】

成员名称	描述
sync	参考rk_aiq_uapi_sync_t定义，参见 **“概述/API说明”** 章节
attrV21	RK356x芯片下merge属性，在RK3588芯片中无效
attrV30	RK3588芯片下merge属性，在RK356x芯片中无效

DRC

功能描述

DRC(动态范围压缩, High Dynamic Range Compression)，其作用是将高比特位的图像压缩到低比特位图像。

重要概念

- 在线性或者HDR模式下均可使用DRC。

功能级API参考

rk_aiq_uapi2_setDrcGain

【描述】

设置DrcGain相关参数。

【语法】

```
XCamReturn rk_aiq_uapi2_setDrcGain(const rk_aiq_sys_ctx_t* ctx, float Gain,
float Alpha, float Clip);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
Gain	Gain值 取值范围：[1,8]	输入
Alpha	Alpha值 取值范围：[0,1]	输入
Clip	Clip值 取值范围：[0,64]	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getDrcGain

【描述】

获取DrcGain相关参数。

【语法】

```
XCamReturn rk_aiq_uapi2_getDrcGain(const rk_aiq_sys_ctx_t* ctx, float Gain,
float Alpha, float Clip);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
Gain	Gain值 取值范围：[1,8]	输出
Alpha	Alpha值 取值范围：[0,1]	输出
Clip	Clip值 取值范围：[0,64]	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setDrcHiLit

【描述】

设置DrcHiLit参数。

【语法】

```
XCamReturn rk_aiq_uapi2_setDrcHiLit(const rk_aiq_sys_ctx_t* ctx, float
Strength);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
Strength	强度 取值范围：[0,1]	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getDrcHiLit

【描述】

获取DrcHiLit参数。

【语法】

```
XCamReturn rk_aiq_uapi2_getDrcHiLit(const rk_aiq_sys_ctx_t* ctx, float
Strength);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
Strength	强度 取值范围：[0,1]	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setDrcLocalData

【描述】

设置DrcLocalData相关参数。

在调用本api时不需要调用rk_aiq_uapi2_enableDrc。同时，本api不能与DRC其他功能级设置api同时调用。

【语法】

```
XCamReturn rk_aiq_uapi2_setDrcLocalData(const rk_aiq_sys_ctx_t* ctx, float
LocalWeit, float GlobalContrast, float LoLitContrast, int LocalAutoEnable, float
LocalAutoWeit);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
LocalWeit	LocalWeit值 取值范围：[0,1]	输入
GlobalContrast	GlobalContrast值 取值范围：[0,1]	输入
LoLitContrast	LoLitContrast值 取值范围：[0,1]	输入
LocalAutoEnable	自动Local开关 取值范围：[0,1]	输入
LocalAutoWeit	自动LocalWeit值 取值范围：[0,1]	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getDrcLocalData

【描述】

获取DrcLocalData相关参数。

【语法】

```
XCamReturn rk_aiq_uapi2_getDrcLocalData(const rk_aiq_sys_ctx_t* ctx, float*
LocalWeit, float* GlobalContrast, float* LoLitContrast, int* LocalAutoEnable,
float* LocalAutoWeit);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
LocalWeit	LocalWeit值 取值范围：[0,1]	输出
GlobalContrast	GlobalContrast值 取值范围：[0,1]	输出
LoLitContrast	LoLitContrast值 取值范围：[0,1]	输出
ocalAutoEnable	自动Local开关 取值范围：[0,1]	输出
LocalAutoWeit	自动LocalWeit值 取值范围：[0,1]	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

模块级API参考

rk_aiq_user_api2_adrc_SetAttrib

【描述】
设置DRC软件属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_adrc_SetAttrib(RkAiqAlgoContext* ctx,
                                drc_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
attr	DRC软件属性结构体	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_adrc.h
- 库文件：librkaiq.so

【说明】

rk_aiq_user_api2_adrc_GetAttrib

【描述】
获取DRC软件属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_adrc_GetAttrib(RkAiqAlgoContext* ctx,
                                drc_attrib_t* attr);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
attr	DRC软件属性结构体	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_adrc.h
- 库文件：librkaiq.so

【说明】

模块级API数据类型

AdrcVersion_t

【说明】
定义DRC版本属性

【定义】

```
typedef enum AdrcVersion_e {
    ADRC_VERSION_356X = 0,
    ADRC_VERSION_3588 = 1,
    ADRC_VERSION_MAX
} AdrcVersion_t;
```

【成员】

成员名称	描述
ADRC_VERSION_356X	RK356x
ADRC_VERSION_3588	RK3588
ADRC_VERSION_MAX	最大版本模式

Drc_OpMode_t

【说明】
定义DRC工作模式

【定义】

```
typedef enum drc_OpMode_e {
    DRC_OPMODE_API_OFF   = 0,
    DRC_OPMODE_MANU      = 1,
    DRC_OPMODE_DRC_GAIN  = 2,
    DRC_OPMODE_HILIT     = 3,
    DRC_OPMODE_LOCAL_TMO = 4,
} drc_OpMode_t;
```

【成员】

成员名称	描述
DRC_OPMODE_API_OFF	api关闭模式
DRC_OPMODE_MANU	手动模式
DRC_OPMODE_AUTO	自动模式
DRC_OPMODE_DRC_GAIN	DrcGain模式，调整DrcGain部分参数
DRC_OPMODE_HILIT	HiLit模式，调整HiLit部分参数
DRC_OPMODE_LOCAL_TMO	LocalTMO模式，调整LocalTMO部分参数
DRC_OPMODE_COMPRESS	Compress模式，调整Compress部分参数

mDrcGain_t

【说明】
定义手动DrcGain参数属性

【定义】

```
typedef struct mDrcGain_s {
    float DrcGain;
    float Alpha;
    float Clip;
} mDrcGain_t;
```

【成员】

成员名称	描述
DrcGain	手动DrcGain值，取值范围[1,8]，默认值为1，精度0.01
Alpha	手动Alpha值，取值范围[0,1]，默认值为0.2，精度0.01
Clip	手动Clip值，取值范围[0,64]，默认值为16，精度0.01

mDrcHiLit_t

【说明】
定义手动Drc HiLit参数属性

【定义】

```
typedef struct mDrcHiLit_s {
    float Strength;
} mDrcHiLit_t;
```

【成员】

成员名称	描述
Strength	手动Strength值，取值范围[0,1]，默认值为0，精度0.01

mLocalDataV21_t

【说明】
定义RK356x下，手动Drc Local参数属性

【定义】

```
typedef struct mLocalDataV21_s {
    float LocalWeit;
    float GlobalContrast;
    float LoLitContrast;
} mLocalDataV21_t;
```

【成员】

成员名称	描述
LocalWeit	手动LocalWeit值，取值范围[0,1]，默认值为1，精度0.01
GlobalContrast	手动GlobalContrast值，取值范围[0,1]，默认值为0，精度0.01
LoLitContrast	手动LoLitContrast值，取值范围[0,1]，默认值为0，精度0.01

mLocalDataV30_t

【说明】
定义RK356x下，手动Drc Local参数属性

【定义】

```
typedef struct mLocalDataV30_s {
    float         LocalWeit;
    int           LocalAutoEnable;
    float         LocalAutoWeit;
    float         GlobalContrast;
    float         LoLitContrast;
} mLocalDataV30_t;
```

【成员】

成员名称	描述
LocalWeit	手动LocalWeit值，取值范围[0,1]，默认值为1，精度0.01。
LocalAutoEnable	自动LocalWeit开关，取值范围[0,1]，默认值为1，精度1。
LocalAutoWeit	自动LocalWeit值，取值范围[0,1]，默认值为0.4，精度0.01。
GlobalContrast	手动GlobalContrast值，取值范围[0,1]，默认值为0，精度0.01。
LoLitContrast	手动LoLitContrast值，取值范围[0,1]，默认值为0，精度0.01。

mDrcLocalV21_t

【说明】
定义RK356x下，手动Drc Local参数属性

【定义】

```
typedef struct mDrcLocalV21_s {
    mLocalDataV21_t LocalData;
    float           curPixWeit;
    float           preFrameWeit;
    float           Range_force_sgm;
    float           Range_sgm_cur;
    float           Range_sgm_pre;
    int             Space_sgm_cur;
    int             Space_sgm_pre;
} mDrcLocalV21_t;
```

【成员】

成员名称	描述
LocalData	手动LocalData设置
curPixWeit	当前点的双边权重，取值范围[0,1]，默认值为0.37，精度0.001。
preFrameWeit	前帧双边权重，取值范围[0,1]，默认值为0.8，精度0.001。
Range_force_sgm	双边值域 sigma 的倒数，取值范围[0,1]，默认值为0，精度0.0001。
Range_sgm_cur	前帧双边空域sigma的倒数，取值范围[0,1]，默认值为0.2，精度0.0001。
Range_sgm_pre	前一帧双边空域sigma的倒数，取值范围[0,1]，默认值为0.2，精度0.0001。
Space_sgm_cur	当前帧双边值域sigma的倒数，取值范围[0,4095]，默认值为4068，精度1。
Space_sgm_pre	前一帧双边值域sigma的倒数，取值范围[0,4095]，默认值为3068，精度1。

mDrcLocalV30_t

【说明】
定义RK3588下，手动Drc Local参数属性

【定义】

```
typedef struct mDrcLocalV30_s {
    mLocalDataV30_t LocalData;
    float           curPixWeit;
    float           preFrameWeit;
    float           Range_force_sgm;
    float           Range_sgm_cur;
    float           Range_sgm_pre;
    int             Space_sgm_cur;
    int             Space_sgm_pre;
} mDrcLocalV30_t;
```

【成员】

成员名称	描述
LocalData	手动LocalData设置
curPixWeit	当前点的双边权重，取值范围[0,1]，默认值为0.37，精度0.001
preFrameWeit	前帧双边权重，取值范围[0,1]，默认值为0.8，精度0.001
Range_force_sgm	双边值域 sigma 的倒数，取值范围[0,1]，默认值为0，精度0.0001
Range_sgm_cur	前帧双边空域sigma的倒数，取值范围[0,1]，默认值为0.2，精度0.0001
Range_sgm_pre	前一帧双边空域sigma的倒数，取值范围[0,1]，默认值为0.2，精度0.0001
Space_sgm_cur	当前帧双边值域sigma的倒数，取值范围[0,4095]，默认值为4068，精度1
Space_sgm_pre	前一帧双边值域sigma的倒数，取值范围[0,4095]，默认值为3068，精度1

CompressMode_t

【说明】
定义手动模式下，DrcCompress曲线工作模式

【定义】

```
typedef enum CompressMode_s {
    COMPRESS_AUTO     = 0,
    COMPRESS_MANUAL   = 1,
} CompressMode_t;
```

【成员】

成员名称	描述
COMPRESS_AUTO	手动模式下，Compress曲线自动模式
COMPRESS_MANUAL	手动模式下，Compress曲线手动模式

mDrcCompress_t

【说明】
定义手动DrcCompress参数属性

【定义】

```
typedef struct mDrcCompress_s {
    CompressMode_t Mode;
    uint16_t       Manual_curve[17];
} mDrcCompress_t;
```

【成员】

成员名称	描述
Mode	开关功能
Manual_curve	手动模式下，手动Compress曲线

mdrcAttr_V21_t

【说明】
定义RK356x芯片下手动Drc属性

【定义】

```
typedef struct mdrcAttr_V21_s {
    bool            Enable;
    mDrcGain_t      DrcGain;
    mDrcHiLit_t     HiLight;
    mDrcLocalV21_t LocalSetting;
    mDrcCompress_t CompressSetting;
    int             Scale_y[ADRC_Y_NUM];
    float           Edge_Weit;
    bool            OutPutLongFrame;
    int             IIR_frame;
} mdrcAttr_V21_t;
```

【成员】

成员名称	描述
Enable	手动Drc开关
DrcGain	手动DrcGain设置
HiLight	手动HiLit设置
LocalSetting	手动Local设置
CompressSetting	压缩曲线设置
Scale_y	增益修正scale表，取值范围[0,2048]，精度1.
Edge_Weit	边缘响应scale值，取值范围[0,1]，默认值0.02，精度0.01。用于降低高对比度边缘Artifact。
OutPutLongFrame	只输出长帧开关，0：关闭，1：开启。
IIR_frame	IIR滤波器帧数，取值范围[1,1000]，默认值为2。

mdrcAttr_V30_t

【说明】
定义RK3588芯片下手动Drc属性

【定义】

```
typedef struct mdrcAttr_V30_s {
    bool            Enable;
    mDrcGain_t      DrcGain;
    mDrcHiLit_t     HiLight;
    mDrcLocalV30_t LocalSetting;
    mDrcCompress_t CompressSetting;
    int             Scale_y[ADRC_Y_NUM];
    float           Edge_Weit;
    bool            OutPutLongFrame;
    int             IIR_frame;
} mdrcAttr_V30_t;
```

【成员】

成员名称	描述
Enable	手动Drc开关
DrcGain	手动DrcGain设置
HiLight	手动HiLit设置
LocalSetting	手动Local设置
CompressSetting	压缩曲线设置
Scale_y	增益修正scale表，取值范围[0,2048]，精度1.
Edge_Weit	边缘响应scale值，取值范围[0,1]，默认值0.02，精度0.01。用于降低高对比度边缘Artifact。
OutPutLongFrame	只输出长帧开关，0：关闭，1：开启。
IIR_frame	IIR滤波器帧数，取值范围[1,1000]，默认值为2。

DrcInfo_t

【说明】
定义DRC控制参数属性

【定义】

```
typedef struct DrcInfo_s {
    float EnvLv;
} DrcInfo_t;
```

【成员】

成员名称	描述
EnvLv	当前环境亮度

drcAttr_t

【说明】
定义DRC属性

【定义】

```
typedef struct drcAttr_s {
    rk_aiq_uapi_sync_t sync;
    AdrcVersion_t      Version;
    drc_OpMode_t       opMode;
    mdrcAttr_V21_t     stManualV21;
    mdrcAttr_V30_t     stManualV30;
    mDrcGain_t         stDrcGain;
    mDrcHiLit_t        stHiLit;
    mLocalDataV21_t    stLocalDataV21;
    mLocalDataV30_t    stLocalDataV30;
    DrcInfo_t          Info;
} drcAttr_t;
```

【成员】

成员名称	描述
sync	参考rk_aiq_uapi_sync_t定义，参见 **“概述/API说明”** 章节
Version	版本信息
opMode	api模式
stManualV21	RK356x芯片下手动Drc属性
stManualV30	RK3588芯片下手动Drc属性
stDrcGain	DrcGain属性
stHiLit	HiLit属性
stLocalDataV21	RK356x芯片下LocalData属性
stLocalDataV30	RK3588芯片下LocalData属性
Info	控制参数信息

Noise Removal

功能描述

图像噪声是指存在于图像数据中的不必要的或多余的干扰信息。图像去噪是减少数字图像中噪声的过程。

功能级API参考

rk_aiq_uapi2_setNRMode

【描述】 设置去噪模式。

【语法】

```
XCamReturn rk_aiq_uapi2_setNRMode(const rk_aiq_sys_ctx_t* ctx, opMode_t mode);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mode	工作模式	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getNRMode

【描述】 获取当前去噪模式。

【语法】

```
XCamReturn rk_aiq_uapi2_getNRMode(const rk_aiq_sys_ctx_t* ctx, opMode_t* mode);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mode	工作模式	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setANRStrth

【描述】 设置普通去噪强度。

【语法】

```
XCamReturn rk_aiq_uapi2_setANRStrth(const rk_aiq_sys_ctx_t* ctx, unsigned int
level);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
level	去噪强度，取值范围0.0-100.0, 默认值50。	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getANRStrth

【描述】 获取普通去噪强度。

【语法】

```
XCamReturn rk_aiq_uapi2_getANRStrth(const rk_aiq_sys_ctx_t* ctx, unsigned int*
level);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
level	去噪强度，取值范围0.0-100.0, 默认值50。	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setMSpaNRStrth

【描述】 设置空域去噪强度。

【语法】

```
XCamReturn rk_aiq_uapi2_setMSpaNRStrth(const rk_aiq_sys_ctx_t* ctx, bool on,
unsigned int level);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
on	开关	输入
level	去噪强度，取值范围0.0-100.0, 默认值50。	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getMSpaNRStrth

【描述】 获取空域去噪强度。

【语法】

```
XCamReturn rk_aiq_uapi2_getMSpaNRStrth(const rk_aiq_sys_ctx_t* ctx, bool *on,
unsigned int *level);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
on	开关	输出
level	去噪强度，取值范围0.0-100.0, 默认值50。	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setMTNRStrth

【描述】 设置时域去噪强度。

【语法】

```
XCamReturn rk_aiq_uapi2_setMTNRStrth(const rk_aiq_sys_ctx_t* ctx, bool on,
unsigned int level);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
on	开关	输入
level	去噪强度，取值范围0.0-100.0, 默认值50。	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getMTNRStrth

【描述】 获取时域去噪强度。

【语法】

```
XCamReturn rk_aiq_uapi2_getMTNRStrth(const rk_aiq_sys_ctx_t* ctx, bool *on,
unsigned int *level);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
on	开关	输出
level	去噪强度，取值范围0.0-100.0, 默认值50。	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

模块级API参考

rk_aiq_user_api2_abayer2dnrV2_SetAttrib

【描述】
设置去噪算法属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_abayer2dnrV2_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_bayer2dnr_attrib_v2_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去噪的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_abayer2dnr_v2.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_abayer2dnrV2_GetAttrib

【描述】
获取去噪算法属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_abayer2dnrV2_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_bayer2dnr_attrib_v2_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去噪的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_abayer2dnr_v2.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_abayer2dnrV2_SetStrength

【描述】
设置去噪力度。

【语法】

```
XCamReturn
rk_aiq_user_api2_abayer2dnrV2_SetStrength(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_bayer2dnr_strength_v2_t *pStrength)
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
pStrength	去噪强度相关结构体，结构体中percent取值范围0.0-1.0	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_abayer2dnr_v2.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_abayer2dnrV2_GetStrength

【描述】
获取去噪力度。

【语法】

```
XCamReturn
rk_aiq_user_api2_abayer2dnrV2_GetStrength(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_bayer2dnr_strength_v2_t *pStrength);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
pPercnt	去噪强度结构体指针，结构体中percent取值范围0.0-1.0	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_abayer2dnr_v2.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_abayertnrV2_SetAttrib

【描述】
设置去噪算法属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_abayertnrV2_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_bayertnr_attrib_v2_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去噪的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_abayertnr_v2.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_abayertnrV2_GetAttrib

【描述】
获取去噪算法属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_abayertnrV2_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_bayertnr_attrib_v2_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去噪的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_abayertnr_v2.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_abayertnrV2_SetStrength

【描述】
设置去噪力度。

【语法】

```
XCamReturn
rk_aiq_user_api2_abayertnrV2_SetStrength(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_bayertnr_strength_v2_t *pStrength);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
pStrength	去噪强度结构体指针，结构体中percent取值范围0.0-1.0	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_abayertnr_v2.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_abayertnrV2_GetStrength

【描述】
获取去噪力度。

【语法】

```
XCamReturn
rk_aiq_user_api2_abayertnrV2_GetStrength(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_bayertnr_strength_v2_t *pStrength);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
pStrength	去噪强度结构体指针，结构体中percent取值范围0.0-1.0	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_abayertnr_v2.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_aynrV3_SetAttrib

【描述】
设置去噪算法属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_aynrV3_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_ynr_attrib_v3_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去噪的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_aynr_v3.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_aynrV3_GetAttrib

【描述】
获取去噪算法属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_aynrV3_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_ynr_attrib_v3_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去噪的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_aynr_v3.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_aynrV3_SetStrength

【描述】
设置去噪力度。

【语法】

```
XCamReturn
rk_aiq_user_api2_aynrV3_SetStrength(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_ynr_strength_v3_t* pStrength);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
pStrength	去噪强度结构体指针，结构体中percent取值范围0.0-1.0	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_aynr_v3.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_aynrV3_GetStrength

【描述】
获取去噪力度。

【语法】

```
XCamReturn
rk_aiq_user_api2_aynrV3_GetStrength(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_ynr_strength_v3_t* pStrength);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
pStrength	去噪强度结构体指针，结构体中percent取值范围0.0-1.0	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_aynr_v3.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_acnrV2_SetAttrib

【描述】
设置去噪算法属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_acnrV2_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_cnr_attrib_v2_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去噪的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_acnr_v2.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_acnrV2_GetAttrib

【描述】
获取去噪算法属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_acnrV2_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_cnr_attrib_v2_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去噪的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_acnr_v2.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_acnrV2_SetStrength

【描述】
设置去噪力度。

【语法】

```
XCamReturn
rk_aiq_user_api2_acnrV2_SetStrength(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_cnr_strength_v2_t *pStrength);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
pStrength	去噪强度结构体指针，结构体中percent取值范围0.0-1.0	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_acnr_v2.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_acnrV2_GetStrength

【描述】
获取去噪力度。

【语法】

```
XCamReturn
rk_aiq_user_api2_acnrV2_GetStrength(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_cnr_strength_v2_t *pStrength);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
pStrength	去噪强度结构体指针，结构体中percent取值范围0.0-1.0	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_acnr_v2.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

模块级API数据类型

rk_aiq_bayer2dnr_attrib_v2_t

【说明】
定义去噪模块的参数

【定义】

```
typedef struct rk_aiq_bayer2dnr_attrib_v2_s {
    rk_aiq_uapi_sync_t sync;
    Abayer2dnr_OPMode_V2_t eMode;
    Abayer2dnr_Auto_Attr_V2_t stAuto;
    Abayer2dnr_Manual_Attr_V2_t stManual;
} rk_aiq_bayer2dnr_attrib_v2_t;
```

【成员】

成员名称	描述
sync	同步异步模式选择，参见 **“概述/API说明”** 章节
eMode	bayer2dnr去噪模块模式
stAuto	bayer2dnr去噪模块自动模式参数
stManual	bayer2dnr去噪模块手动模式参数

Abayer2dnr_OPMode_V2_t

【说明】
定义去噪模块的模式

【定义】

```
typedef enum Abayer2dnr_OPMode_V2_e {
    ABAYER2DNR_OP_MODE_INVALID          = 0,
    ABAYER2DNR_OP_MODE_AUTO             = 1,
    ABAYER2DNR_OP_MODE_MANUAL           = 2,
    ABAYER2DNR_OP_MODE_REG_MANUAL       = 3,
    ABAYER2DNR_OP_MODE_MAX
} Abayer2dnr_OPMode_V2_t;
```

【成员】

成员名称	描述
ABAYER2DNR_OP_MODE_INVALID	bayer域 2dnr去噪模块无效模式
ABAYER2DNR_OP_MODE_AUTO	bayer域 2dnr去噪模块自动模式
ABAYER2DNR_OP_MODE_MANUAL	bayer域 2dnr去噪模块手动模式的算法设置
ABAYER2DNR_OP_MODE_REG_MANUAL	bayer域 2dnr去噪模块手动模式的寄存器设置
ABAYER2DNR_OP_MODE_MAX	bayer域 2dnr去噪模块模式最大值，是一个无效模式

Abayer2dnr_Auto_Attr_V2_t

【说明】
定义去噪模块的自动属性

【定义】

```
typedef struct Abayer2dnr_Auto_Attr_V2_s
{
    RK_Bayer2dnr_Params_V2_t st2DParams;
    RK_Bayer2dnr_Params_V2_Select_t st2DSelect;

} Abayer2dnr_Auto_Attr_V2_t;
```

【成员】

成员名称	描述
st2DParams	bayer2dnr模块各个iso对应算法属性参数
st2DSelect	bayer2dnr模块根据当前iso计算出来属性参数

Abayer2dnr_Manual_Attr_V2_t

【说明】
定义去噪模块的手动属性

【定义】

```
typedef struct Abayer2dnr_Manual_Attr_V2_s
{
    RK_Bayer2dnr_Params_V2_Select_t st2DSelect;
    RK_Bayer2dnr_Fix_V2_t st2Dfix;

} Abayer2dnr_Manual_Attr_V2_t;
```

【成员】

成员名称	描述
st2DSelect	bayer2dnr手动模式下算法参数值
st2Dfix	bayer2dnr手动模式下寄存器值

RK_Bayer2dnr_Params_V2_t

【说明】

定义去噪模块的自动模式各个iso对应算法属性参数

【定义】

```
typedef struct RK_Bayer2dnr_Params_V2_s
{
    int enable;

    float iso[RK_BAYER2DNR_V2_MAX_ISO_NUM];

    int   lumapoint[16];
    int   sigma[RK_BAYER2DNR_V2_MAX_ISO_NUM][16];

    float filter_strength[RK_BAYER2DNR_V2_MAX_ISO_NUM];
    float edgesofts[RK_BAYER2DNR_V2_MAX_ISO_NUM];
    float ratio[RK_BAYER2DNR_V2_MAX_ISO_NUM];
    float weight[RK_BAYER2DNR_V2_MAX_ISO_NUM];
    int   gauss_guide[RK_BAYER2DNR_V2_MAX_ISO_NUM];

    int pix_diff[RK_BAYER2DNR_V2_MAX_ISO_NUM];
    int diff_thld[RK_BAYER2DNR_V2_MAX_ISO_NUM];

} RK_Bayer2dnr_Params_V2_t;
```

【成员】

成员名称	参数类型	描述
enable	调试参数	模块开关使能。1：模块打开，0：模块关闭。
ISO	调试参数	不同iso档位，对应不同调试参数。目前仅支持13档。
lumapoint	标定数据	不同pixel亮度，对应不同噪声sigma曲线点。共16个点。
sigma	标定数据	不同pixel亮度，对应不同噪声sigma曲线点。共16个点。
gauss_guide	调试参数	高斯导向是否使能。1：使能。0：关闭。
filter_strength	调试参数	去噪力度参数。取值范围[0, 16], 值越大，去噪力度越大。
edgesofts	调试参数	影响空域权重。取值范围[1, 16]，默认值为1。
ratio	调试参数	软阈值权重。取值范围[0, 1.0]。值越小，去噪力度越大。
weight	调试参数	滤波输出权重，值越大，去噪力度越大。
pix_diff	调试参数	双边滤波的 5x5 窗口像素差值门限;默认配置值 0x3fff.
diff_thld	调试参数	双边滤波的计算欧式距离的平方差门限，默认配置值 0x3ff;

RK_Bayer2dnr_Params_V2_Select_t

【说明】
定义去噪模块的手动模式下算法属性

【定义】

```
typedef struct RK_Bayer2dnr_Params_V2_Select_s
{
    int enable;

    int lumapoint[16];
    int sigma[16];

    float filter_strength;
    float edgesofts;
    float ratio;
    float weight;
    int gauss_guide;

    int pix_diff;
    int diff_thld;

} RK_Bayer2dnr_Params_V2_Select_t;
```

【成员】

成员名称	参数类型	描述
enable	调试参数	模块开关使能。1：模块打开，0：模块关闭。
lumapoint	标定数据	不同pixel亮度，对应不同噪声sigma曲线点。共16个点。
sigma	标定数据	不同pixel亮度，对应不同噪声sigma曲线点。共16个点。
gauss_guide	调试参数	高斯导向是否使能。1：使能。0：关闭。
filter_strength	调试参数	去噪力度参数。取值范围[0, 16], 值越大，去噪力度越大。
edgesofts	调试参数	影响空域权重。取值范围[1, 16]，默认值为1。
ratio	调试参数	软阈值权重。取值范围[0, 1.0]。值越小，去噪力度越大。
weight	调试参数	滤波输出权重，值越大，去噪力度越大。
pix_diff	调试参数	双边滤波的 5x5 窗口像素差值门限;默认配置值 0x3fff.
diff_thld	调试参数	双边滤波的计算欧式距离的平方差门限，默认配置值 0x3ff;

RK_Bayer2dnr_Fix_V2_t

【说明】
定义去噪模块的手动模式下寄存器配置

【定义】

```
typedef struct RK_Bayer2dnr_Fix_V2_s {

    //ISP_BAYNR_3A00_CTRL
    uint8_t baynr_lg2_mode;
    uint8_t baynr_gauss_en;
    uint8_t baynr_log_bypass;
    uint8_t baynr_en;

    // ISP_BAYNR_3A00_DGAIN0-2
    uint16_t baynr_dgain[3];

    // ISP_BAYNR_3A00_PIXDIFF
    uint16_t baynr_pix_diff;

    // ISP_BAYNR_3A00_THLD
    uint16_t baynr_diff_thld;
    uint16_t baynr_softthld;

    // ISP_BAYNR_3A00_W1_STRENG
    uint16_t bltflt_streng;
    uint16_t baynr_reg_w1;

    // ISP_BAYNR_3A00_SIGMAX0-15
    uint16_t sigma_x[16];

    // ISP_BAYNR_3A00_SIGMAY0-15
    uint16_t sigma_y[16];
```

```
    // ISP_BAYNR_3A00_WRIT_D
    uint16_t weit_d[3];

    uint16_t lg2_lgoff;
    uint16_t lg2_off;

    uint32_t dat_max;
} RK_Bayer2dnr_Fix_V2_t;
```

【成员】

参数	位宽	参数说明
baynr_gauss_en	1	高斯3x3导向滤波开关使能位; 1:打开滤波; 0：关闭滤波; 默认配置值为0;
baynr_log_bypass	1	log变换是否执行的控制位,hdr时硬件内部强制关闭; 1:打开log变换bypass,就是关闭log变换; 0:关闭log变换bypass,就是打开log转换; 默认值为0;
baynr_en	1	Bayer 降噪的开关使能位; 1: 打开降噪; 0: 关闭降噪; 默认配置值为 0;
baynr_dgain	18	HDR 模式下,对应三帧的增益的参数,小数 10bit.
baynr_pix_diff	14	双边滤波的 5x5 窗口像素差值门限; 默认配置值 0x3fff
baynr_diff_thld	10	双边滤波的计算欧式距离的平方差门限, 默认配置值 0x3ff;
baynr_softthld	10	软阈值的权重值,值越大阈值越大,保留的噪声越多, 小数位 10bit,默认配置值 0xa;
bltflt_streng	12	双边滤波的降噪强度; 小数 8bit; 默认配置值 0xa3
baynr_reg_w1	10	滤波输出权重值,值越大使用滤波输出值越多,越小使用原始值越多. 小数位 10bit, 默认配置值 0x3ff;
sigma_x	16	噪声曲线的 16 个亮度水平等级，配置时要保证两个值的 插值为 2 的幂次方;
sigma_y	16	噪声曲线的 16 个噪声 sigma 等级; 配置参数由标定得到.
weit_d	10	空域权重值,分别表示不同位置的权重值. 小数 10bit; 默认配置值分别是 0x178, 0x249, 0x31d;

rk_aiq_bayer2dnr_strength_v2_t

【说明】
定义去噪模块的去噪强度配置结构体

【定义】

```
typedef struct rk_aiq_bayer2dnr_strength_v2_s {
    rk_aiq_uapi_sync_t sync;
    float percent;
} rk_aiq_bayer2dnr_strength_v2_t;
```

【成员】

成员名称	描述
sync	同步异步模式选择，参见 **“概述/API说明”** 章节
percent	去噪强度，取值范围0.0-1.0。

rk_aiq_bayertnr_attrib_v2_t

【说明】
定义去噪模块参数

【定义】

```
typedef struct rk_aiq_bayertnr_attrib_v2_s {
    rk_aiq_uapi_sync_t sync;
    Abayertnr_OPMode_V2_t eMode;
    Abayertnr_Auto_Attr_V2_t stAuto;
    Abayertnr_Manual_Attr_V2_t stManual;
} rk_aiq_bayertnr_attrib_v2_t;
```

【成员】

成员名称	描述
sync	同步异步模式选择，参见 **“概述/API说明”** 章节
eMode	bayertnr去噪模块模式
stAuto	bayertnr去噪模块自动模式参数
stManual	bayertnr去噪模块手动模式参数

Abayertnr_OPMode_V2_t

【说明】
定义去噪模块的模式

【定义】

```
typedef enum Abayertnr_OPMode_V2_e {
    ABAYERTNRV2_OP_MODE_INVALID           = 0,
    ABAYERTNRV2_OP_MODE_AUTO              = 1,
    ABAYERTNRV2_OP_MODE_MANUAL            = 2,
    ABAYERTNRV2_OP_MODE_REG_MANUAL        = 3,
    ABAYERTNRV2_OP_MODE_MAX
} Abayertnr_OPMode_V2_t;
```

【成员】

成员名称	描述
ABAYERTNRV2_OP_MODE_INVALID	bayer域 tnr去噪模块无效模式
ABAYERTNRV2_OP_MODE_AUTO	bayer域 tnr去噪模块自动模式
ABAYERTNRV2_OP_MODE_MANUAL	bayer域 tnr去噪模块手动模式的算法设置
ABAYERTNRV2_OP_MODE_REG_MANUAL	bayer域 tnr去噪模块手动模式的寄存器设置
ABAYERTNRV2_OP_MODE_MAX	bayer域 tnr去噪模块模式最大值，是一个无效模式

Abayertnr_Auto_Attr_V2_t

【说明】
定义去噪模块的自动属性

【定义】

```
typedef struct Abayertnr_Auto_Attr_V2_s
{
    int bayernr3DEn;

    RK_Bayertnr_Params_V2_t st3DParams;
    RK_Bayertnr_Params_V2_Select_t st3DSelect;

} Abayertnr_Auto_Attr_V2_t;
```

【成员】

成员名称	描述
bayernr3DEn	bayer3dnr模块使能开关
st3DParams	bayer3dnr模块自动模式各个iso对应算法属性参数
st3DSelect	bayer3dnr模块根据当前iso计算出来属性参数

Abayertnr_Manual_Attr_V2_t

【说明】
定义去噪模块的手动属性

【定义】

```
typedef struct Abayertnr_Manual_Attr_V2_s
{
    int bayernr3DEn;
    RK_Bayertnr_Params_V2_Select_t st3DSelect;

    RK_Bayertnr_Fix_V2_t st3DFix;
} Abayertnr_Manual_Attr_V2_t;
```

【成员】

成员名称	描述
bayernr3DEn	bayer3dnr模块使能开关
st3DSelect	bayer3dnr手动模式下算法参数值
st3Dfix	bayer3dnr手动模式下寄存器值

RK_Bayertnr_Params_V2_t

【说明】
定义去噪模块的自动模式各个iso对应算法属性参数

【定义】

```
typedef struct RK_Bayertnr_Params_V2_s
{
    int enable;

    float iso[RK_BAYERNR_V2_MAX_ISO_NUM];

    //calib
    int   lumapoint[16];
    int   sigma[RK_BAYERNR_V2_MAX_ISO_NUM][16];
    int   lumapoint2[16];
    int   lo_sigma[RK_BAYERNR_V2_MAX_ISO_NUM][16];
    int   hi_sigma[RK_BAYERNR_V2_MAX_ISO_NUM][16];

    //tuning
    int   thumbds[RK_BAYERNR_V2_MAX_ISO_NUM];

    int   lo_enable[RK_BAYERNR_V2_MAX_ISO_NUM];
    int   hi_enable[RK_BAYERNR_V2_MAX_ISO_NUM];
    int   lo_med_en[RK_BAYERNR_V2_MAX_ISO_NUM];
    int   lo_gsbay_en[RK_BAYERNR_V2_MAX_ISO_NUM];
    int   lo_gslum_en[RK_BAYERNR_V2_MAX_ISO_NUM];
    int   hi_med_en[RK_BAYERNR_V2_MAX_ISO_NUM];
    int   hi_gslum_en[RK_BAYERNR_V2_MAX_ISO_NUM];
    int   global_pk_en[RK_BAYERNR_V2_MAX_ISO_NUM];
    int   global_pksq[RK_BAYERNR_V2_MAX_ISO_NUM];

    float lo_filter_strength[RK_BAYERNR_V2_MAX_ISO_NUM];
    float hi_filter_strength[RK_BAYERNR_V2_MAX_ISO_NUM];
    float soft_threshold_ratio[RK_BAYERNR_V2_MAX_ISO_NUM];
    float hi_wgt_comp[RK_BAYERNR_V2_MAX_ISO_NUM];
```

```
    float clipwgt[RK_BAYERNR_V2_MAX_ISO_NUM];
    float hidif_th[RK_BAYERNR_V2_MAX_ISO_NUM];
} RK_Bayertnr_Params_V2_t;
```

【成员】

成员名称	参数类型	描述
Enable	调试参数	模块使能开关
iso	调试参数	不同iso档位，对应不同调试参数。目前仅支持13档
lumapoint / sigma	标定参数	不同亮度对应的噪声曲线值。共16个点。 lumapoint对应pixel亮度,sigma对应噪声曲线值
lumapoint2 / lo_sigma	标定参数	不同亮度对应的噪声曲线值。共16个点。 lumapoint2对应像素亮度,lo_sigma对应噪声曲线值
lumapoint2 / hi_sigma	标定参数	不同亮度对应的噪声曲线值。共16个点。 lumapoint2对应像素亮度,hi_sigma对应噪声曲线值
thumbds	调试参数	下采样比例
lo_enable	调试参数	低频运动判断是否打开, 1打开，0关闭。默认打开。
hi_enable	调试参数	高频运动判断是否打开, 1打开，0关闭。默认打开。
lo_med_en	调试参数	内部低频子模块开关, 1打开，0关闭。默认打开。
lo_gsbay_en	调试参数	内部低频子模块开关, 1打开，0关闭。默认打开。
lo_gslum_en	调试参数	内部低频子模块开关, 1打开，0关闭。默认打开。
hi_med_en	调试参数	内部高频子模块开关, 1打开，0关闭。默认打开。
hi_gslum_en	调试参数	内部高频子模块开关, 1打开，0关闭。默认打开。
global_pk_en	调试参数	时域降噪是否使用全局 pk, 1使用，0不使用。 目前暂只能用0.
global_pksq	调试参数	全局 pk 的平方值，当global_pk_en为 1 的时候才用它. 默认值1024,取值范围[0, 268435455]
lo_filter_strength	调试参数	高频运动判断力度。最终影响hi sigma进而影响时域去噪力度。 默认值1， 取值范围[0.0, 16.0]
hi_filter_strength	调试参数	高频运动判断力度。最终影响hi sigma进而影响时域去噪力度。 默认值1， 取值范围[0.0, 16.0]

成员名称	参数类型	描述
soft_threshold_ratio	调试参数	软阈值权重。值越大，保留噪声越多 取值范围[0.0 1.0]，默认值0。
hi_wgt_comp	调试参数	叠加权重回补的比例系数值,只在高频打开的时候才有用; 默认值0.16， 取值范围[0.0, 1.0]。
clipwgt	调试参数	图像叠加的权重限制值。 默认值0.03215， 取值范围[0.0, 1.0]。
hidif_th	调试参数	高频差异阈值。 默认值32767， 取值范围[0, 65535]。

RK_Bayertnr_Params_V2_Select_t

【说明】
定义去噪模块的手动模式下算法属性结构体

【定义】

```
typedef struct RK_Bayertnr_Params_V2_Select_s
{
    int enable;

    //calib
    int lumapoint[16];
    int sigma[16];
    int lumapoint2[16];
    int lo_sigma[16];
    int hi_sigma[16];

    //tuning
    int thumbds;
    int lo_enable;
    int hi_enable;
    int lo_med_en;
    int lo_gsbay_en;
    int lo_gslum_en;
    int hi_med_en;
    int hi_gslum_en;
    int global_pk_en;
    int global_pksq;

    float lo_filter_strength;
    float hi_filter_strength;
    float soft_threshold_ratio;

    float clipwgt;
    float hi_wgt_comp;
    float hidif_th;
} RK_Bayertnr_Params_V2_Select_t;
```

【成员】

成员名称	参数类型	描述
Enable	调试参数	模块使能开关
lumapoint / sigma	标定参数	不同亮度对应的噪声曲线值。共16个点。 lumapoint对应pixel亮度,sigma对应噪声曲线值
lumapoint2 / lo_sigma	标定参数	不同亮度对应的噪声曲线值。共16个点。 lumapoint2对应像素亮度,lo_sigma对应噪声曲线值
lumapoint2 / hi_sigma	标定参数	不同亮度对应的噪声曲线值。共16个点。 lumapoint2对应像素亮度,hi_sigma对应噪声曲线值
thumbds	调试参数	下采样比例
lo_enable	调试参数	低频运动判断是否打开, 1打开，0关闭。默认打开。
hi_enable	调试参数	高频运动判断是否打开, 1打开，0关闭。默认打开。
lo_med_en	调试参数	内部低频子模块开关, 1打开，0关闭。默认打开。
lo_gsbay_en	调试参数	内部低频子模块开关, 1打开，0关闭。默认打开。
lo_gslum_en	调试参数	内部低频子模块开关, 1打开，0关闭。默认打开。
hi_med_en	调试参数	内部高频子模块开关, 1打开，0关闭。默认打开。
hi_gslum_en	调试参数	内部高频子模块开关, 1打开，0关闭。默认打开。
global_pk_en	调试参数	时域降噪是否使用全局 pk, 1使用，0不使用。 目前暂只能用0.
global_pksq	调试参数	全局 pk 的平方值，当global_pk_en为 1 的时候才用它. 默认值1024,取值范围[0, 268435455]
lo_filter_strength	调试参数	高频运动判断力度。最终影响hi sigma进而影响时域去噪力度。 默认值1， 取值范围[0.0, 16.0]
hi_filter_strength	调试参数	高频运动判断力度。最终影响hi sigma进而影响时域去噪力度。 默认值1， 取值范围[0.0, 16.0]
soft_threshold_ratio	调试参数	软阈值权重。值越大，保留噪声越多 取值范围[0.0 1.0]，默认值0。

成员名称	参数类型	描述
hi_wgt_comp	调试参数	叠加权重回补的比例系数值,只在高频打开的时候才有用; 默认值0.16， 取值范围[0.0, 1.0]。
clipwgt	调试参数	图像叠加的权重限制值。 默认值0.03215， 取值范围[0.0, 1.0]。
hidif_th	调试参数	高频差异阈值。 默认值32767， 取值范围[0, 65535]。

RK_Bayertnr_Fix_V2_t

【说明】
定义去噪模块的手动模式下寄存器配置

【定义】

```
typedef struct RK_Bayertnr_Fix_V2_s {

    // BAY3D_BAY3D_CTRL 0x2c00
    uint8_t bay3d_exp_sel;
    uint8_t bay3d_soft_st;
    uint8_t bay3d_soft_mode;
    uint8_t bay3d_bwsaving_en;
    uint8_t bay3d_loswitch_protect;
    uint8_t bay3d_glbpk_en;
    uint8_t bay3d_logaus3_bypass_en;
    uint8_t bay3d_logaus5_bypass_en;
    uint8_t bay3d_lomed_bypass_en;
    uint8_t bay3d_hichnsplit_en;
    uint8_t bay3d_hiabs_pssel;
    uint8_t bay3d_higaus_bypass_en;
    uint8_t bay3d_himed_bypass_en;
    uint8_t bay3d_lobypass_en;
    uint8_t bay3d_hibypass_en;
    uint8_t bay3d_bypass_en;
    uint8_t bay3d_en_i;

    // BAY3D_BAY3D_KALRATIO 0x2c04
    uint16_t bay3d_softwgt;
    uint16_t bay3d_hidif_th;

    // BAY3D_BAY3D_GLBPK2 0x2c08
    uint32_t bay3d_glbpk2;

    // BAY3D_BAY3D_WGTLMT 0x2c10
    uint16_t bay3d_wgtlmt;
    uint16_t bay3d_wgtratio;

    // BAY3D_BAY3D_SIG_X0  0x2c14 - 0x2c30
    uint16_t bay3d_sig0_x[16];

    // BAY3D_BAY3D_SIG0_Y0 0x2c34 - 0x2c50
```

```
    uint16_t bay3d_sig0_y[16];

    // BAY3D_BAY3D_SIG_X0  0x2c54 - 0x2c70
    uint16_t bay3d_sig1_x[16];

    // BAY3D_BAY3D_SIG1_Y0 0x2c74 - 0x2c90
    uint16_t bay3d_sig1_y[16];

    // BAY3D_BAY3D_SIG2_Y0 0x2c94 - 0x2cb0
    uint16_t bay3d_sig2_y[16];

    //BAY3D_BAY3D_LODIF_STAT0 0x2cb4 -0x2cb8
    uint64_t ro_sum_lodif;

    //BAY3D_BAY3D_LODIF_STAT0 0x2cbc -0x2cc0
    uint64_t ro_sum_hidif0;

    //BAY3D_BAY3D_MI_ST 0x2CC8
    uint8_t sw_bay3dmi_st_linemode;
    uint8_t sw_bay3d_mi2cur_linecnt;
} RK_Bayertnr_Fix_V2_t;
```

【成员】

参数	位宽	参数说明
bay3d_soft_st	1	软件模式的启动脉冲位,开启的时候写 1.
bay3d_soft_mode	1	软件模式的使能位. 1: 打开软件模式;> 0: 关闭软件模式; 默认配置值 0x0;
bay3d_bwsaving_en	1	带宽优化使能位,非 hdr 才有效. 1: 打开优化; 0: 关闭优化; 默认配置值 0x0;
bay3d_loswitch_protect	1	
bay3d_glbpk_en	1	Bayer 时域降噪是否使用全局 pk; 1: 使用全局的 pk 值; 0: 使用局部的 pk 值; 默认配置值为 1;
bay3d_logaus3_bypass_en	1	低频在后级亮度域上是否做高斯滤波; 1: 不做; 0: 做; 默认配置值 0x0;
bay3d_logaus5_bypass_en	1	低频在前级 bayer 域上是否做高斯滤波; 1: 不做; 0: 做; 默认配置值 0x0;
bay3d_lomed_bypass_en	1	低频是否做中值滤波; 1: 不做; 0: 做; 默认配置值 0x0;
bay3d_hichnsplit_en	1	高频判断时选择它的高斯滤波是在 bayer 域或亮度域上做,只开高频的运动判断时，打开这个使能位; 1: 在 bayer 域高斯滤波; 0: 在亮度域做高斯滤波; 默认配置值 0x0;
bay3d_hiabs_pssel	1	高频判断时在高斯滤波前是否做绝对值,只开高频的运动判断时，打开这个使能位; 1: 做; 0: 不做; 默认配置值 0x0;
bay3d_higaus_bypass_en	1	高频是否做高斯滤波; 1: 不做; 0: 做; 默认配置值 0x0;

参数	位宽	参数说明
bay3d_himed_bypass_en	1	高频是否做中值滤波; 1: 不做; 0: 做; 默认配置值 0x0;
bay3d_lobypass_en	1	高频运动判断是否打开; 1: 关闭; 0: 打开; 默认配置值 0x0;
bay3d_hibypass_en	1	高频运动判断是否打开; 1: 关闭; 0: 打开; 默认配置值 0x0;
bay3d_bypass_en	1	Bayer 时域降噪的 bypass 使能位; 1: 打开 bypass,相当于 bayer3d 不做. 0: 关闭 bypass; 默认配置值为 0;
bay3d_en_i	1	Bayer 时域降噪的开关使能位; 1: 打开时域降噪; 0: 关闭时域降噪; 默认配置值为 0;
bay3d_softwgt	10	软阈值的权重值,值越大阈值越大,保留的噪声越多. 小数位 10bit,默认配置值 0x100;
bay3d_hidif_th	16	统计高频差异值个数的阈值.默认配置值 0xffff
bay3d_glbpk2	28	全局 pk 的平方值,当 pk_en 为 0 的时候才用它.默认配置值 0x800;
bay3d_wgtlmt	10	图像叠加的权重限制值. 小数位为 10 位, 默认配置值 0x380;
bay3d_wgtratio	10	叠加权重回补的比例系数值,只在高频打开的时候才有用; 小数位为 10 位, 默认配置值 0x0;
bay3d_sig0_x0-15	16	噪声曲线的 16 个亮度水平等级，配置时要保证两个值的差值为 2 的幂次方;
bay3d_sig0_y0-15	14	噪声曲线的 16 个噪声 sigma 等级;配置参数由标定得到.
bay3d_sig1_x0-15	16	高低频的噪声曲线的 16 个亮度水平等级，配置时要保证两个值的差值为 2 的幂次方;
bay3d_sig1_y0-15	14	高频的噪声曲线的 16 个噪声 sigma 等级;配置参数由标定得到.
bay3d_sig2_y0-15	14	低频的噪声曲线的 16 个噪声 sigma 等级;配置参数由标定得到.

rk_aiq_bayertnr_strength_v2_t

【说明】
定义去噪模块的去噪强度配置结构体

【定义】

```
typedef struct rk_aiq_bayertnr_strength_v2_s {
    rk_aiq_uapi_sync_t sync;
    float percent;
} rk_aiq_bayertnr_strength_v2_t;
```

【成员】

成员名称	描述
sync	同步异步模式选择，参见 **“概述/API说明”** 章节
percent	去噪强度，取值范围0.0-1.0。

rk_aiq_ynr_attrib_v3_t

【说明】
定义去噪模块的参数

【定义】

```
typedef struct rk_aiq_ynr_attrib_v3_s {
    rk_aiq_uapi_sync_t sync;
    Aynr_OPMode_V3_t eMode;
    Aynr_Auto_Attr_V3_t stAuto;
    Aynr_Manual_Attr_V3_t stManual;
} rk_aiq_ynr_attrib_v3_t;
```

【成员】

成员名称	描述
sync	同步异步模式选择，参见 **“概述/API说明”** 章节
eMode	ynr去噪模块模式
stAuto	ynr去噪模块自动模式参数
stManual	ynr去噪模块手动模式参数

Aynr_OPMode_V3_t

【说明】
定义去噪模块的模式

【定义】

```
typedef enum Aynr_OPMode_V3_e {
    AYNRV3_OP_MODE_INVALID          = 0,
    AYNRV3_OP_MODE_AUTO             = 1,
    AYNRV3_OP_MODE_MANUAL           = 2,
    AYNRV3_OP_MODE_REG_MANUAL       = 3,
    AYNRV3_OP_MODE_MAX
} Aynr_OPMode_V3_t;
```

【成员】

成员名称	描述
AYNRV3_OP_MODE_INVALID	y域 ynr去噪模块无效模式
AYNRV3_OP_MODE_AUTO	y域 ynr去噪模块自动模式
AYNRV3_OP_MODE_MANUAL	y域 ynr去噪模块手动模式的算法设置
AYNRV3_OP_MODE_REG_MANUAL	y域 ynr去噪模块手动模式的寄存器设置
AYNRV3_OP_MODE_MAX	y域 ynr去噪模块模式最大值，是一个无效模式

Aynr_Auto_Attr_V3_t

【说明】
定义去噪模块的自动属性

【定义】

```
typedef struct Aynr_Auto_Attr_V3_s
{
    RK_YNR_Params_V3_t stParams;
    RK_YNR_Params_V3_Select_t stSelect;

} Aynr_Auto_Attr_V3_t;
```

【成员】

成员名称	描述
stParams	ynr模块各个iso对应算法属性参数
stSelect	ynr模块根据当前iso计算出来属性参数

Aynr_Manual_Attr_V3_t

【说明】
定义去噪模块的手动属性

【定义】

```
typedef struct Aynr_Manual_Attr_V3_s
{
    RK_YNR_Params_V3_Select_t stSelect;
    RK_YNR_Fix_V3_t stFix;

} Aynr_Manual_Attr_V3_t;
```

【成员】

成员名称	描述
stSelect	ynr手动模式下算法参数值
stFix	ynr手动模式下寄存器值

RK_YNR_Params_V3_t

【说明】
定义YNR去噪模块自动模式各个iso对应算法属性参数

【定义】

```
typedef struct RK_YNR_Params_V3_s
{
    int enable;
    char version[64];
    float iso[RK_YNR_V3_MAX_ISO_NUM];
    RK_YNR_Params_V3_Select_t arYnrParamsISO[RK_YNR_V3_MAX_ISO_NUM];

} RK_YNR_Params_V3_t;
```

【成员】

成员名称	描述
enable	ynr模块使能开关
version	ynr模块版本号
iso	不同iso档位，对应不同调试参数。目前仅支持13档
RK_YNR_Params_V3_Select_t	ynr模块算法参数

RK_YNR_Params_V3_Select_t

【说明】
定义去噪模块的手动模式下算法属性

【定义】

```
typedef struct RK_YNR_Params_V3_Select_s
{
    int enable;
```

```
    float lci;
    float hci;
    float sigma[YNR_V3_ISO_CURVE_POINT_NUM];
    short lumaPoint[YNR_V3_ISO_CURVE_POINT_NUM];

    float lo_lumaPoint[6];
    float lo_ratio[6];

    float hi_lumaPoint[6];
    float hi_ratio[6];

    // low frequency
    float rnr_strength[17];
    int ynr_bft3x3_bypass;
    int ynr_lbft5x5_bypass;
    int ynr_lgft3x3_bypass;
    int ynr_flt1x1_bypass;
    int ynr_sft5x5_bypass;
    float low_bf1;
    float low_bf2;
    float low_thred_adj;
    float low_peak_supress;
    float low_edge_adj_thresh;
    float low_lbf_weight_thresh;
    float low_center_weight;
    float low_dist_adj;
    float low_weight;
    float low_filt1_strength;
    float low_filt2_strength;
    float low_bi_weight;

    // high frequency
    float base_filter_weight1;
    float base_filter_weight2;
    float base_filter_weight3;
    float high_thred_adj;
    float high_weight;
    float high_direction_weight[8];
    float hi_min_adj;
    float hi_edge_thed;

    //local gain control
    float ynr_global_gain_alpha;
    float ynr_global_gain;
    float ynr_adjust_thresh;
    float ynr_adjust_scale;
} RK_YNR_Params_V3_Select_t;
```

【成员】

成员名称	参数类型	描述
enable	调试参数	ynr模块使能开关，1：模块打开，0：模块关闭。
lci	标定数据	影响低频噪声sigma影响因子。值越大，噪声sigma越大，去噪力度越强。
hci	标定数据	影响高频噪声sigma影响因子。值越大，噪声sigma越大，去噪力度越强。
sigma	标定数据	噪声sigma曲线。
lumaPoint	标定数据	不同pixel亮度，对应不同噪声sigma曲线点。共16个点。
rnr_strength	调试参数	图像中心，按照圆的半径r方向，设置不同去噪力度。 主要是为lsc这种噪声进行配置的。 取值范围[0, 16.0]，默认值1。
ynr_bft3x3_bypass	调试参数	模块内部子模块bypass功能 0：功能使能 1：功能bypass 一般情况，全部子模块都打开使能，这几个值设置为0。
ynr_lbft5x5_bypass	调试参数	模块内部子模块bypass功能 0：功能使能 1：功能bypass 一般情况，全部子模块都打开使能，这几个值设置为0。
ynr_lgft3x3_bypass	调试参数	模块内部子模块bypass功能 0：功能使能 1：功能bypass 一般情况，全部子模块都打开使能，这几个值设置为0。
ynr_flt1x1_bypass	调试参数	模块内部子模块bypass功能 0：功能使能 1：功能bypass 一般情况，全部子模块都打开使能，这几个值设置为0。
ynr_sft5x5_bypass	调试参数	模块内部子模块bypass功能 0：功能使能 1：功能bypass 一般情况，全部子模块都打开使能，这几个值设置为0。
low_bf1	调试参数	原图3x3双边滤波力度，值越大，去噪越强。 取值范围[0.01, 32]，默认值1。
low_bf2	调试参数	上一帧小图5x5双边滤波力度，值越大，去噪越强。 取值范围[0.01, 32]，默认值1。
low_thred_adj	调试参数	低频软阈值的调整力度，值越大，低频降噪力度越大。 取值范围[0, 31]，默认值0.5。

成员名称	参数类型	描述
low_peak_supress	调试参数	控制去除孤立噪声的力度，值越小，去噪力度越大。 取值范围[0, 1]，默认值0.5。
low_edge_adj_thresh	调试参数	小图边缘检测的调整系数的门限，用于限制调整系数所能取到的最大值。 值越小，去噪力度越大，图像越模糊。取值范围[0, 1023]整数，默认值7。
low_lbf_weight_thresh	调试参数	用于对 5x5 的双边滤波的权重进行限制， 该值越大，则低频降噪力度越弱。 取值范围[0.0,1.0]。默认值0.25。
low_center_weight	调试参数	5x5 双边滤波时中心点的权重，该值越小，则降噪力度越强。 取值范围[0,1]，默认值0.5。
low_dist_adj	调试参数	双边滤波距离权重调整因子。值越小，去噪越强。 取值范围[0, 127.0]，默认值8.0。
low_weight	调试参数	低频去噪结果的权重,值越大，低频降噪力度越大。 取值范围[0, 1]，默认值0.5。
low_filt1_strength	调试参数	对原图进行高斯滤波的滤波核权重。 取值范围[0，1.0]，默认值0.7。
low_filt2_strength	调试参数	对双边滤波的结果进行高斯滤波的滤波核权重。 取值范围[0，1.0]，默认值0.85。
low_bi_weight	调试参数	软阈值处理中使用的第一步双边滤波权重，该值越大，则降噪力度也越大。 取值范围[0, 1]，默认值0.3。
base_filter_weight1	调试参数	方向滤波器的系数 。一般不太需要调整。
high_thred_adj	调试参数	软阈值的调整系数，该值越大，则高频降噪的力度也越大。 取值范围[0, 31.0]，默认值1.0。
high_weight	调试参数	高频去噪权重，注意该值表示的是保留的高频分量的比例，值越小，则表示降噪力度越强. 取值范围[0, 1]，默认值0.78。
high_direction_weight	调试参数	各个方向的权重设置， 某一方向上该值越大，表示沿着该方向的降噪力度越强。
hi_min_adj	调试参数	所有差异值减去的最小差异值的比例，该值越大，则边缘越锐利。 取值范围[0.0, 1.0]，默认值0.9
hi_edge_thed	调试参数	对差异值作限制的门限，该值越小，则高频降噪力度越大。 取值范围[0, 255]的整数,默认值100。

成员名称	参数类型	描述
ynr_global_gain_alpha	调试参数	ynr去噪local模式和global模式插值力度配置。 一般使用默认值，不用配置，全部使用local gain的方式。 取值范围[0.0 1.0]，默认值0
ynr_global_gain	调试参数	ynr去噪local模式和global模式插值力度配置。 一般使用默认值，不用配置，全部使用local gain的方式。 gain取值范围[0.0 64.0]。默认值1。
ynr_adjust_thresh	调试参数	对大于阈值ynr_adjust_thresh的噪声进行去噪力度控制。 运动区域噪声比较大，设定合适阈值，使运动区域ynr去噪力度加大。 取值范围[0.0, 1.0], 默认值1。
ynr_adjust_scale	调试参数	对大于阈值ynr_adjust_thresh的噪声进行去噪力度控制。 运动区域噪声比较大，设定合适阈值，使运动区域ynr去噪力度加大。 取值范围[0, 16.0], 默认值1
lo_lumaPoint / lo_ratio	调试参数	不同pixel亮度，对低频sigma进行微调。
hi_lumaPoint / hi_ratio	调试参数	不同pixel亮度，对高频sigma进行微调。

RK_YNR_Fix_V3_t

【说明】
定义去噪模块的手动模式下寄存器配置

【定义】

```
typedef struct RK_YNR_Fix_V3_s {

    // YNR_2700_GLOBAL_CTRL (0x0000)
    uint8_t ynr_rnr_en;
    uint8_t ynr_gate_dis;
    uint8_t ynr_thumb_mix_cur_en;
    uint8_t ynr_global_gain_alpha;
    uint16_t ynr_global_gain;
    uint8_t ynr_flt1x1_bypass_sel;
    uint8_t ynr_sft5x5_bypass;
    uint8_t ynr_flt1x1_bypass;
    uint8_t ynr_lgft3x3_bypass;
    uint8_t ynr_lbft5x5_bypass;
    uint8_t ynr_bft3x3_bypass;
    uint8_t ynr_en;


    // YNR_2700_RNR_MAX_R  (0x0004)
    uint8_t ynr_local_gainscale;
    uint16_t ynr_rnr_max_r;

    // YNR_2700_RNR_MAX_R  (0x0008)
```

```
    uint16_t ynr_rnr_center_coorv;
    uint16_t ynr_rnr_center_coorh;

    // YNR_2700_RNR_MAX_R  (0x000c)
    uint8_t ynr_localgain_adj;
    uint16_t ynr_localgain_adj_thresh;

    // YNR_2700_LOWNR_CTRL0 (0x0010)
    uint16_t ynr_low_bf_inv[2];

    // YNR_2700_LOWNR_CTRL1  (0x0014)
    uint8_t ynr_low_peak_supress;
    uint16_t ynr_low_thred_adj;

    // YNR_2700_LOWNR_CTRL2 (0x0018)
    uint16_t ynr_low_dist_adj;
    uint16_t ynr_low_edge_adj_thresh;

    // YNR_2700_LOWNR_CTRL3 (0x001c)
    uint8_t ynr_low_bi_weight;
    uint8_t ynr_low_weight;
    uint16_t ynr_low_center_weight;

    // YNR_2700_HIGHNR_CTRL0 (0x0020)
    uint8_t ynr_hi_min_adj;
    uint16_t ynr_high_thred_adj;

    // YNR_2700_HIGHNR_CTRL1  (0x0024)
    uint8_t ynr_high_retain_weight;
    uint8_t ynr_hi_edge_thed;

    // YNR_2700_HIGHNR_BASE_FILTER_WEIGHT  (0x0028)
    uint8_t ynr_base_filter_weight[3];

    // YNR_2700_HIGHNR_BASE_FILTER_WEIGHT  (0x002c)
    uint32_t ynr_frame_full_size;
    uint16_t ynr_lbf_weight_thres;

    // YNR_2700_GAUSS1_COEFF  (0x0030)
    uint16_t ynr_low_gauss1_coeff[3];

    // YNR_2700_GAUSS2_COEFF  (0x0034)
    uint16_t ynr_low_gauss2_coeff[3];

    // YNR_2700_DIRECTION_W_0_3  (0x0038 - 0x003c)
    uint8_t ynr_direction_weight[8];

    // YNR_2700_SGM_DX_0_1 (0x0040 - 0x0060)
    uint16_t ynr_luma_points_x[17];

    // YNR_2700_LSGM_Y_0_1 (0x0070- 0x0090)
    uint16_t ynr_lsgm_y[17];

    // YNR_2700_HSGM_Y_0_1 (0x00a0- 0x00c0)
    uint16_t ynr_hsgm_y[17];

    // YNR_2700_RNR_STRENGTH03 (0x00d0- 0x00e0)
    uint16_t ynr_rnr_strength[17];
```

```
} RK_YNR_Fix_V3_t;
```

【成员】

成员名称	位宽（整数.小数）	描述
ynr_low_bf_inv[0]	5.9	用于控制去噪力度，注意这个寄存器配置的值力度的倒数，所以该值越小，去 噪的力度越大。 该值为 512 表示 1 倍的去噪力度， 256 表示 2 倍的去噪力度。 取值范围 [1, 16383]
ynr_low_bf[1]	5.9	用于控制去噪力度，注意这个寄存器配置的值力度的倒数，所以该值越小，去噪的力度越大。 该值为 512 表示 1 倍的去噪力度， 256 表示 2 倍的去噪力度
ynr_low_thred_adj	5.6	低频软阈值的调整力度，越大则低频降噪力度越强 取值范围[0, 2047]
ynr_low_peak_supress	1.7	控制去除较大的孤立噪声的力度，该值越 大则力度越强。 取值范围[0, 128]
ynr_low_edge_adj_thresh	11.0	小图边缘检测的调整系数的门限，用于限制调整系数所能取到的最大值。 取值范围[1, 1024]
ynr_low_center_weight	1.10	5x5 双边滤波时中心点的权重，该值越小，则降噪力度越强。 取值范围[1, 1024]
sw_ynr_lbf_weight_thres	10.0	用于对 5x5 的双边滤波的权重进行限制，该值越大，则低频降噪力度越弱。 取值范围[0, 1023]
ynr_low_dist_adj	7.2	双边滤波距离权重调整因子 取值范围[1, 511]
ynr_low_weight	1.7	低频去噪结果的权重，该值越大则低频降噪力度越大。 取值范围[0, 128]
ynr_low_gauss1_coeff[3]	1.8	对原图进行高斯滤波的滤波核权重 取值范围[0, 256]
ynr_low_gauss2_coeff[3]	1.8	对双边滤波的结果进行高斯滤波的高斯滤 波核权重 取值范围[0, 256]
ynr_low_bi_weight	1.7	软阈值处理中使用的第一步双边滤波权重，该值越大，则降噪力度也越大。 取值范围[0, 128]
ynr_base_filter_weight[3]	1.6	方向滤波器的系数 取值范围[0, 64]

成员名称	位宽（整数.小数）	描述
ynr_high_thred_adj	5.6	软阈值的调整系数，该值越大，则高频降噪的力度也越大。 取值范围 [0, 2047]
ynr_high_retain_weight	1.7	高频去噪权重，注意该值表示的是保留的高频分量的比例，因此，该值越大，则表示降噪力度越弱。 取值范围 [0, 128]
ynr_direction_weight[8]	1.4	各个方向的权重设置， 某一方向上该值越大，表示沿着该方向的降噪力度越强。 取值范围 [0,16]
ynr_hi_min_adj	0.6	所有差异值减去的最小差异值的比例，该值越大，则边缘越锐利。 取值范围 [6,60]
ynr_hi_edge_thed	10.0	对差异值作限制的门限，该值越小，则高频降噪力度越大。 取值范围 [0,1023]
ynr_rnr_max_r	14.0	用于径向降噪力度调整，用于存放 1/((rows/2)^2+(cols/2)^2)的值，其中前面 9 bit 是尾数 M，后面 5 bit 是指数 E 取值范围 [1, 16383]
ynr_rnr_strength[17]	4.4	用于控制径向的降噪力度，一共有 17个数据点，分为 16 段，表示从图像中心点到图像的四个角的去噪力度。 取值范围 [0, 15]
ynr_global_gain	6.4	外部配置的全局 gain，用于控制 YNR整体的降噪力度，值越大则降噪力度越强。 取值范围 [0, 1023]
ynr_global_gain_alpha	1.3	用于控制 gain 模块传递的 gain 值和外部配置的 gain 值的使用比例，该值越大则使用外部全局 gain 的比例越大。 取值范围 [0, 8]
ynr_bft3x3_bypass	1	为 1 时关掉 3x3 双边滤波 取值范围 0或1
ynr_lbft5x5_bypass	1	为 1 时关掉 5x5 双边滤波 取值范围 0或1
ynr_lgft3x3_bypass	1	为 1 时关掉针对 5x5 双边滤波结果再做的 3x3 高斯滤波
ynr_flt1x1_bypass	1	为 1 时关掉低频软阈值处理 取值范围 0或1

成员名称	位宽（整数.小数）	描述
ynr_sft5x5_bypass	1	为 1 时关掉 5x5 的高频降噪 取值范围 0或1

rk_aiq_ynr_strength_v3_t

【说明】
定义去噪模块的去噪强度配置结构体

【定义】

```
typedef struct rk_aiq_ynr_strength_v3_s {
    rk_aiq_uapi_sync_t sync;
    float percent;
} rk_aiq_ynr_strength_v3_t;
```

【成员】

成员名称	描述
sync	同步异步模式选择，参见 **“概述/API说明”** 章节
percent	去噪强度，取值范围0.0-1.0。

rk_aiq_cnr_attrib_v2_t

【说明】
定义去噪模块参数

【定义】

```
typedef struct rk_aiq_cnr_attrib_v2_s {
    rk_aiq_uapi_sync_t sync;
    AcnrV2_OPMode_t eMode;
    Acnr_Auto_Attr_V2_t stAuto;
    Acnr_Manual_Attr_V2_t stManual;
} rk_aiq_cnr_attrib_v2_t;
```

【成员】

成员名称	描述
sync	同步异步模式选择，参见 **“概述/API说明”** 章节
eMode	cnr 去噪模块模式
stAuto	cnr 去噪模块自动模式参数
stManual	cnr 去噪模块手动模式参数

AcnrV2_OPMode_t

【说明】
定义去噪模块的模式属性

【定义】

```
typedef enum AcnrV2_OPMode_e {
    ACNRV2_OP_MODE_INVALID          = 0,
    ACNRV2_OP_MODE_AUTO             = 1,
    ACNRV2_OP_MODE_MANUAL           = 2,
    ACNRV2_OP_MODE_REG_MANUAL       = 3,
    ACNRV2_OP_MODE_MAX
} AcnrV2_OPMode_t;
```

【成员】

成员名称	描述
ACNRV2_OP_MODE_INVALID	uv域 cnr去噪模块无效模式
ACNRV2_OP_MODE_AUTO	uv域 cnr去噪模块自动模式
ACNRV2_OP_MODE_MANUAL	uv域 cnr去噪模块手动模式的算法设置
ACNRV2_OP_MODE_REG_MANUAL	uv域 cnr去噪模块手动模式的寄存器设置
ACNRV2_OP_MODE_MAX	uv域 cnr去噪模块模式最大值，是一个无效模式

Acnr_Auto_Attr_V2_t

【说明】
定义去噪模块的自动属性

【定义】

```
typedef struct Acnr_Auto_Attr_V2_s
{
    //all ISO params and select param

    RK_CNR_Params_V2_t stParams;
    RK_CNR_Params_V2_Select_t stSelect;

} Acnr_Auto_Attr_V2_t;
```

【成员】

成员名称	描述
stParams	cnr模块各个iso对应算法属性参数
stSelect	cnr模块根据当前iso计算出来属性参数

Acnr_Manual_Attr_V2_t

【说明】
定义去噪模块的手动属性

【定义】

```
typedef struct Acnr_Manual_Attr_V2_s
{
    RK_CNR_Params_V2_Select_t stSelect;

    RK_CNR_Fix_V2_t stFix;

} Acnr_Manual_Attr_V2_t;
```

【成员】

成员名称	描述
stSelect	cnr手动模式下算法参数值
stSelect	cnr手动模式下寄存器值

RK_CNR_Params_V2_t

【说明】
定义去噪模块的自动模式各个iso对应算法属性参数

【定义】

```
typedef struct RK_CNR_Params_V2_s
{
    int enable;
    float iso[RK_CNR_V2_MAX_ISO_NUM];
```

```
    int hf_bypass[RK_CNR_V2_MAX_ISO_NUM];
    int lf_bypass[RK_CNR_V2_MAX_ISO_NUM];

    // gain
    float global_gain[RK_CNR_V2_MAX_ISO_NUM];
    float global_gain_alpha[RK_CNR_V2_MAX_ISO_NUM];
    float local_gain_scale[RK_CNR_V2_MAX_ISO_NUM];

    // strength adj by gain
    int gain_adj_strength_ratio[RK_CNR_V2_MAX_ISO_NUM]
[RKCNR_V2_SGM_ADJ_TABLE_LEN];

    //
    float color_sat_adj[RK_CNR_V2_MAX_ISO_NUM];
    float color_sat_adj_alpha[RK_CNR_V2_MAX_ISO_NUM];

    // step1
    // median filter
    float hf_spikes_reducion_strength[RK_CNR_V2_MAX_ISO_NUM];

    // bilateral filter
    float hf_denoise_strength[RK_CNR_V2_MAX_ISO_NUM];
    float hf_color_sat[RK_CNR_V2_MAX_ISO_NUM];
    float hf_denoise_alpha[RK_CNR_V2_MAX_ISO_NUM];
    int  hf_bf_wgt_clip[RK_CNR_V2_MAX_ISO_NUM];


    // step2
    // median filter
    float thumb_spikes_reducion_strength[RK_CNR_V2_MAX_ISO_NUM];

    // bilateral filter
    float thumb_denoise_strength[RK_CNR_V2_MAX_ISO_NUM];
    float thumb_color_sat[RK_CNR_V2_MAX_ISO_NUM];

    // step3
    // bilateral filter
    float lf_denoise_strength[RK_CNR_V2_MAX_ISO_NUM];
    float lf_color_sat[RK_CNR_V2_MAX_ISO_NUM];
    float lf_denoise_alpha[RK_CNR_V2_MAX_ISO_NUM];

    // bilateral filter kernels
    float kernel_5x5[5];
} RK_CNR_Params_V2_t;
```

【成员】

参数名称	参数类型	参数说明
Enable	调试参数	模块开关使能。1：模块打开，0：模块关闭。
iso	调试参数	不同iso档位，对应不同调试参数。目前仅支持13档。
hf_bypass	调试参数	高频降噪bypass. 0：不bypass, 1:bypass.
lf_bypass	调试参数	低频降噪bypass.0：不bypass, 1:bypass.
global_gain	调试参数	cnr去噪local模式和global模式插值力度配置。一般使用默认值，不用配置，全部使用local gain的方式。 取值范围[0.0 64.0]，默认值1。
global_gain_alpha	调试参数	cnr去噪local模式和global模式插值力度配置。一般使用默认值，不用配置，全部使用local gain的方式。 取值范围[0.0 1.0]，默认值0。
global_gain_scale	调试参数	放大cnr去噪力度，一般用默认值，不太去调整。 取值范围[0, 128]，默认值1。
gain_adj_strength_ratio	调试参数	根据 local gain 值调整滤波强度。值越小，去噪力度越强。 取值范围[1, 255]，默认值255.
color_sat_adj	调试参数	基于梯度调整双边滤波的 uv 比例，1~255。值越小，去彩噪越好。 取值范围[1, 255]。默认值40。
color_sat_adj_alpha	调试参数	color_sat_adj 调整的比例。值越大，去彩噪越好。 取值范围[0, 1.0]。默认值0.8。

参数名称	参数类型	参数说明
hf_spikes_reducion_strength	调试参数	高频中值滤波强度。值越大，中值滤波越强。 取值范围[0, 1.0]。默认值0.5。
hf_denoise_strength	调试参数	高频双边滤波强度。值越大，去彩噪越好。 取值范围[1, 1023]。默认值10。
hf_color_sat	调试参数	高频双边滤波的 uv 比例因子。值越小，色彩饱和度下降越多。 取值范围[0.0, 7.9]。默认值1.5。
hf_denoise_alpha	调试参数	高频双边滤波中心点的权重。 取值范围[0.0, 1.0]。默认值0。
hf_bf_wgt_clip	调试参数	高频最小去噪力度。值越大，去噪越强。 取值范围[0, 255]。默认值0。
thumb_spikes_reducion_strength	调试参数	缩略图中值滤波强度。值越大，中值滤波越强。 取值范围[0.0, 1.0]。默认值0.5。
thumb_denoise_strength	调试参数	缩略图双边滤波强度。值越大，去彩噪越好。 取值范围[1, 1023]。默认值8。
thumb_color_sat	调试参数	缩略图双边滤波的 uv 比例因子，色彩饱和度下降越多。 取值范围[0.0, 7.9]。默认值4。
lf_denoise_strength	调试参数	低频双边滤波强度。值越大，去彩噪越好。 取值范围[1, 1023]。默认值8。
lf_color_sat	调试参数	低频双边滤波的 uv 比例因。值越小，色彩饱和度下降越多。 取值范围[0.0, 7.9]。默认值4。

参数名称	参数类型	参数说明
lf_denoise_alpha	调试参数	低频双边滤波中心点的权重。 取值范围[0.0, 1.0]。默认值0.5。
kernel_5x5	调试参数	5x5双边滤波核。

RK_CNR_Params_V2_Select_t

【说明】
定义去噪模块的手动模式下算法属性

【定义】

```
typedef struct RK_CNR_Params_V2_Select_s
{
    int enable;

    // bypass
    int hf_bypass;
    int lf_bypass;

    // gain

    // gain
    float global_gain;
    float global_gain_alpha;
    float local_gain_scale;

    // strength adj by gain
    int gain_adj_strength_ratio[RKCNR_V2_SGM_ADJ_TABLE_LEN];

    //
    float color_sat_adj;
    float color_sat_adj_alpha;

    // step1
    // median filter
    float hf_spikes_reducion_strength;

    // bilateral filter
    float hf_denoise_strength;
    float hf_color_sat;
    float hf_denoise_alpha;
    int hf_bf_wgt_clip;
```

```
    // step2

    // median filter
    float thumb_spikes_reducion_strength;

    // bilateral filter
    float thumb_denoise_strength;
    float thumb_color_sat;

    // step3
    // bilateral filter
    float lf_denoise_strength;
    float lf_color_sat;
    float lf_denoise_alpha;

    // bilateral filter kernels
    float kernel_5x5[5];

} RK_CNR_Params_V2_Select_t;
```

【成员】

参数名称	参数类型	参数说明
Enable	调试参数	模块开关使能。1：模块打开，0：模块关闭。
iso	调试参数	不同iso档位，对应不同调试参数。目前仅支持13档。
hf_bypass	调试参数	高频降噪bypass. 0：不bypass, 1:bypass.
lf_bypass	调试参数	低频降噪bypass.0：不bypass, 1:bypass.
global_gain	调试参数	cnr去噪local模式和global模式插值力度配置。一般使用默认值，不用配置，全部使用local gain的方式。 取值范围[0.0 64.0]，默认值1。
global_gain_alpha	调试参数	cnr去噪local模式和global模式插值力度配置。一般使用默认值，不用配置，全部使用local gain的方式。 取值范围[0.0 1.0]，默认值0。
global_gain_scale	调试参数	放大cnr去噪力度，一般用默认值，不太去调整。 取值范围[0, 128]，默认值1。
gain_adj_strength_ratio	调试参数	根据 local gain 值调整滤波强度。值越小，去噪力度越强。 取值范围[1, 255]，默认值255.
color_sat_adj	调试参数	基于梯度调整双边滤波的 uv 比例，1~255。值越小，去彩噪越好。 取值范围[1, 255]。默认值40。
color_sat_adj_alpha	调试参数	color_sat_adj 调整的比例。值越大，去彩噪越好。 取值范围[0, 1.0]。默认值0.8。

参数名称	参数类型	参数说明
hf_spikes_reducion_strength	调试参数	高频中值滤波强度。值越大，中值滤波越强。 取值范围[0, 1.0]。默认值0.5。
hf_denoise_strength	调试参数	高频双边滤波强度。值越大，去彩噪越好。 取值范围[1, 1023]。默认值10。
hf_color_sat	调试参数	高频双边滤波的 uv 比例因子。值越小，色彩饱和度下降越多。 取值范围[0.0, 7.9]。默认值1.5。
hf_denoise_alpha	调试参数	高频双边滤波中心点的权重。 取值范围[0.0, 1.0]。默认值0。
hf_bf_wgt_clip	调试参数	高频最小去噪力度。值越大，去噪越强。 取值范围[0, 255]。默认值0。
thumb_spikes_reducion_strength	调试参数	缩略图中值滤波强度。值越大，中值滤波越强。 取值范围[0.0, 1.0]。默认值0.5。
thumb_denoise_strength	调试参数	缩略图双边滤波强度。值越大，去彩噪越好。 取值范围[1, 1023]。默认值8。
thumb_color_sat	调试参数	缩略图双边滤波的 uv 比例因子，色彩饱和度下降越多。 取值范围[0.0, 7.9]。默认值4。
lf_denoise_strength	调试参数	低频双边滤波强度。值越大，去彩噪越好。 取值范围[1, 1023]。默认值8。
lf_color_sat	调试参数	低频双边滤波的 uv 比例因。值越小，色彩饱和度下降越多。 取值范围[0.0, 7.9]。默认值4。

参数名称	参数类型	参数说明
lf_denoise_alpha	调试参数	低频双边滤波中心点的权重。 取值范围[0.0, 1.0]。默认值0.5。
kernel_5x5	调试参数	5x5双边滤波核。

RK_CNR_Fix_V2_t

【说明】
定义去噪模块的手动模式下寄存器配置

【定义】

```
typedef struct RK_CNR_Fix_V2_s {
    //ISP_CNR_2800_CTRL
    uint8_t cnr_thumb_mix_cur_en;
    uint8_t cnr_lq_bila_bypass;
    uint8_t cnr_hq_bila_bypass;
    uint8_t cnr_exgain_bypass;
    uint8_t cnr_en_i;

    // ISP_CNR_2800_EXGAIN
    uint8_t cnr_global_gain_alpha;
    uint16_t cnr_global_gain;

    // ISP_CNR_2800_GAIN_PARA
    uint8_t cnr_gain_iso;
    uint8_t cnr_gain_offset;
    uint8_t cnr_gain_1sigma;

    // ISP_CNR_2800_GAIN_UV_PARA
    uint8_t cnr_gain_uvgain1;
    uint8_t cnr_gain_uvgain0;

    // ISP_CNR_2800_LMED3
    uint8_t cnr_lmed3_alpha;

    // ISP_CNR_2800_LBF5_GAIN
    uint8_t cnr_lbf5_gain_y;
    uint8_t cnr_lbf5_gain_c;

    // ISP_CNR_2800_LBF5_WEITD0_4
    uint8_t cnr_lbf5_weit_d[5];

    // ISP_CNR_2800_HMED3
```

```
    uint8_t cnr_hmed3_alpha;

    // ISP_CNR_2800_HBF5
    uint8_t cnr_hbf5_weit_src;
    uint8_t cnr_hbf5_min_wgt;
    uint16_t cnr_hbf5_sigma;

    // ISP_CNR_2800_LBF3
    uint8_t cnr_lbf5_weit_src;
    uint16_t cnr_lbf3_sigma;

    //ISP_CNR_2800_SIGMA0-SIGMA3
    uint8_t cnr_sigma_y[13];

} RK_CNR_Fix_V2_t;
```

【成员】

参数名称	参数说明
cnr_thumb_mix_cur_en	当前帧缩略图混合上一帧图像使能信号: 1'b1:mix 启用 1'b0:mix 是禁用的 回读的是显示寄存器。
cnr_lq_bila_bypass	低频双边 3x3 滤波器旁路使能信号 1'b1:bypass is enable, out_data 等于 in_data 在这个 3x3 过滤器中。 1'b0:bypass 被禁用。 回读的是显示寄存器。
cnr_hq_bila_bypass	高频5x5双边滤波器bypass信号 1:bypass is enable, out_data 等于 in_data 在这个 5x5 过滤器中。 0:bypass 被禁用。 回读的是显示寄存器。
cnr_exgain_bypass	sw_cnr_exgain_bypass 外部增益bypass使能信号 1:bypass 使能，外部增益值固定为 sw_cnr_egain_mux 0:bypass 禁用，使用外部增益。 当前版本不支持绕过禁用。 回读的是显示寄存器。
cnr_en_i	CNR 模块使能信号 1: 使能； 0： 失能
cnr_global_gain	cnr去噪local模式和global模式插值力度配置。一般使用默认值，不用配置，全部使用local gain的方式。 取值范围[0.0 64.0]，默认值1。
cnr_global_gain_alpha	cnr去噪local模式和global模式插值力度配置。一般使用默认值，不用配置，全部使用local gain的方式。 取值范围[0.0 1.0]，默认值0。
cnr_gain_iso	放大外部增益，范围0~128
cnr_gain_offset	增益校准中低通滤波器的增益偏移，范围0~16
cnr_gain_1sigma	在增益校准中放大高通滤波器的输出数据
cnr_gain_uvgain1	uvgain参数1
cnr_gain_uvgain0	uvgain参数0
cnr_lmed3_alpha	in_data在低频中值滤波器中的权重，范围从0~16
cnr_lbf5_gain_y;	lbf5x5 双边滤波器的 y 增益
cnr_lbf5_gain_c;	lbf5x5 双边滤波器的 uv 增益
cnr_lbf5_weit_d[0]	pix12的空间权重，范围1~128
cnr_lbf5_weit_d[1]	pix7,11,13,17的空间权重，范围0~128
cnr_lbf5_weit_d[2]	pix6,8,16,18的空间权重，范围0~128
cnr_lbf5_weit_d[3]	距离4或5的空间权重，范围0~128

参数名称	参数说明
cnr_lbf5_weit_d[4]	pix0,4,20,24的空间权重，范围0~128
cnr_hmed3_alpha	in_data在高频中值滤波器中的权重，范围从0~16
cnr_hbf5_weit_src	hbf5x5的in_data权重，范围0~128
cnr_hbf5_min_wgt	hbf5x5 weit_r 下限
cnr_hbf5_sigma	hbf5x5 的sigma值
cnr_lbf5_weit_src	lbf3x3的in_data权重，范围0~128
cnr_lbf3_sigma	lbf3x3 的sigma值

rk_aiq_cnr_strength_v2_t

【说明】
定义去噪模块的去噪强度配置结构体

【定义】

```
typedef struct rk_aiq_cnr_strength_v2_s {
    rk_aiq_uapi_sync_t sync;
    float percent;
} rk_aiq_cnr_strength_v2_t;
```

【成员】

成员名称	描述
sync	同步异步模式选择，参见 **“概述/API说明”** 章节
percent	去噪强度，取值范围0.0-1.0。

BLC

功能描述

功能级API参考

rk_aiq_user_api2_ablc_SetAttrib

【描述】
设置blc参数属性

【语法】

```
XCamReturn
rk_aiq_user_api2_ablc_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_blc_attrib_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	blc参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ablc.h、RkAiqHandleInt.h、rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

rk_aiq_user_api2_ablc_GetAttrib

【描述】
获取BLC参数属性

【语法】

```
XCamReturn
rk_aiq_user_api2_ablc_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_blc_attrib_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	blc参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_ablc.h、RkAiqHandleInt.h、rk_aiq_user_api2_sysctl.h
- 库文件：librkaiq.so

模块级API数据类型

rk_aiq_blc_attrib_t

【说明】
blc模块参数

【定义】

```
typedef struct rk_aiq_blc_attrib_s {
    rk_aiq_uapi_sync_t sync;
    AblcOPMode_t eMode;
    AblcParams_t stBlc0Auto;
    AblcParams_t stBlc1Auto;
    AblcManualAttr_t stBlc0Manual;
    AblcManualAttr_t stBlc1Manual;
} rk_aiq_blc_attrib_t;
```

【成员】

成员名称	描述
sync	同步异步模式选择，参见 **“概述/API说明”** 章节
eMode	blc模块模式
stBlc0Auto	blc0 自动模式参数
stBlc0Manual	blc0 手动模式参数
stBlc1Auto	blc1 自动模式参数
stBlc1Manual	blc1 手动模式参数

AblcOPMode_t

【说明】
定义去噪模块的模式属性

【定义】

```
typedef enum AblcOPMode_e {
    ABLC_OP_MODE_OFF          = 0,
    ABLC_OP_MODE_AUTO             = 1,
    ABLC_OP_MODE_MANUAL           = 2,
    ABLC_OP_MODE_MAX
} AblcOPMode_t;
```

【成员】

成员名称	描述
ABLC_OP_MODE_OFF	BLC 模块无效模式
ABLC_OP_MODE_AUTO	BLC 模块自动模式
ABLC_OP_MODE_MANUAL	BLC 模块手动模式的算法设置
ABLC_OP_MODE_MAX	BLC 模块模式最大值，是一个无效模式

AblcParams_t

【说明】
blc模块自动模式参数

【定义】

```
typedef struct AblcParams_s {
    bool enable;
    int len;
    float* iso;
    float* blc_r;
    float* blc_gr;
    float* blc_gb;
    float* blc_b;
} AblcParams_t;
```

【成员】

成员名称	描述
enable	模块使能开关 1：使能； 0：失能；
len	iso对应数组长度
iso	iso等级
blc_r	不同iso对应，blc值。blc的数组指针
blc_gr	不同iso对应，blc值。blc的数组指针
blc_gb	不同iso对应，blc值。blc的数组指针
blc_b	不同iso对应，blc值。blc的数组指针

AblcManualAttr_t

【说明】
blc模块手动模式参数

【定义】

```
typedef struct AblcSelect_s {
    bool enable;
    short int blc_r;
    short int blc_gr;
    short int blc_gb;
    short int blc_b;
} AblcSelect_t;

typedef AblcSelect_t AblcManualAttr_t;
```

【成员】

成员名称	描述
enable	模块使能开关 1：使能； 0：失能；
blc_r	blc模块r通道属性
blc_gr	blc模块gr通道属性
blc_gb	blc模块gb通道属性
blc_b	blc模块b通道属性

Dehaze&Enhance

功能描述

Dehaze&Enhance包含2种模式：

- Dehaze 是通过动态的改变图象的对比度和亮度来实现的去雾增强。
- Enhance提升图像局部区域的对比度。

  其中 Dehaze 与 Enhance功能互斥

功能级API参考

rk_aiq_uapi2_setMDehazeStrth

【描述】

设置手动去雾力度。

设置后API模式为DEHAZE_API_DEHAZE_MANUAL。

【语法】

```
XCamReturn rk_aiq_uapi2_setMDehazeStrth(const rk_aiq_sys_ctx_t* ctx, unsigned
int level);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
level	等级，，取值范围[1,100]，默认值为50，精度1。	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getMDehazeStrth

【描述】

获取手动去雾力度。

若当前API模式不是DEHAZE_API_DEHAZE_MANUAL，则无非获取到等级。

【语法】

```
XCamReturn rk_aiq_uapi2_getMDehazeStrth(const rk_aiq_sys_ctx_t* ctx, unsigned
int* level);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
level	等级，，取值范围[1,100]，默认值为50，精度1。	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setMEnhanceStrth

【描述】

设置手动Enhance等级。

设置后API模式为DEHAZE_API_ENHANCE_MANUAL。

【语法】

```
XCamReturn rk_aiq_uapi2_setMEnhanceStrth(const rk_aiq_sys_ctx_t* ctx, unsigned int level);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
level	等级，，取值范围[1,100]，默认值为50，精度1。	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getMEnhanceStrth

【描述】

获取手动Enhance等级。

若当前API模式不是DEHAZE_API_ENHANCE_MANUAL，则无非获取到等级。

【语法】

```
XCamReturn rk_aiq_uapi2_getMEnhanceStrth(const rk_aiq_sys_ctx_t* ctx, unsigned int *level);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
level	等级，，取值范围[1,100]，默认值为50，精度1。	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

模块级API参考

rk_aiq_user_api2_adehaze_setSwAttrib

【描述】

设置去雾参数。

【语法】

```
XCamReturn  rk_aiq_user_api2_adehaze_setSwAttrib(const rk_aiq_sys_ctx_t*
sys_ctx, adehaze_sw_V2_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去雾参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

rk_aiq_user_api2_adehaze_getSwAttrib

【描述】

获取当前去雾参数。

【语法】

```
XCamReturn  rk_aiq_user_api2_adehaze_getSwAttrib(const rk_aiq_sys_ctx_t*
sys_ctx, adehaze_sw_V2_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去雾参数	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

模块级API数据类型

dehaze_api_mode_t

【说明】
定义dehaze模块工作模式

【定义】

```
typedef enum dehaze_api_mode_e {
    DEHAZE_API_BYPASS                   = 0,
    DEHAZE_API_MANUAL                   = 1,
    DEHAZE_API_DEHAZE_AUTO              = 2,
    DEHAZE_API_DEHAZE_MANUAL            = 3,
    DEHAZE_API_DEHAZE_OFF               = 4,
    DEHAZE_API_ENHANCE_MANUAL           = 5,
    DEHAZE_API_ENHANCE_AUTO             = 6,
    DEHAZE_API_ENHANCE_OFF              = 7,
} dehaze_api_mode_t;
```

【成员】

成员名称	描述
DEHAZE_API_BYPASS	api关闭模式
DEHAZE_API_MANUAL	手动dehaze模块模式
DEHAZE_API_DEHAZE_AUTO	自动Dehaze功能模式，此时Dehaze功能开启，Enhance功能关闭，自动Dehaze功能参数由Json决定
DEHAZE_API_DEHAZE_MANUAL	手动Dehaze功能模式，此时Dehaze功能开启，Enhance功能关闭，手动Dehaze功能参数由stDehazeManu决定
DEHAZE_API_DEHAZE_OFF	Dehaze功能关闭模式，此时Dehaze功能关闭，自动Enhance功能开关以及功能参数由Json决定
DEHAZE_API_ENHANCE_MANUAL	手动Enahnce功能模式，此时Dehaze功能关闭，Enhance功能开启，手动Enhance功能参数由stEnhanceManu决定
DEHAZE_API_ENHANCE_AUTO	自动Enahnce功能模式，此时Dehaze功能关闭，Enhance功能开启，自动Enahnce功能参数由Json决定
DEHAZE_API_ENHANCE_OFF	Enhance功能关闭模式，此时Enhance功能关闭，自动Dehaze功能开关以及功能参数由Json决定

mDehazeDataV21_t

【说明】
定义手动DehazeData属性

【定义】

```
typedef struct mDehazeDataV21_s {
    float dc_min_th;
    float dc_max_th;
    float yhist_th;
    float yblk_th;
    float dark_th;
    float bright_min;
    float bright_max;
    float wt_max;
    float air_min;
    float air_max;
    float tmax_base;
    float tmax_off;
    float tmax_max;
    float cfg_wt;
    float cfg_air;
    float cfg_tmax;
    float dc_weitcur;
    float bf_weight;
    float range_sigma;
    float space_sigma_pre;
    float space_sigma_cur;
} mDehazeDataV21_t;
```

【成员】

成员名称	描述
dc_min_th	wt自适应的统计范围，取值范围[16, 120]，默认值64，精度0.1。
dc_max_th	wt自适应高曝区统计范围，取值范围[170, 255]，默认值192，精度0.1。
yhist_th	y分量高曝区统计范围，取值范围[170, 255]，默认值249，精度0.1。
yblk_th	y分量块数目比例阈值，取值范围[0.002, 0.01]，默认值0.002，精度0.1。
dark_th	wt自适应y分量块最小值阈值，取值范围[230, 250]，默认值250，精度0.1。
bright_min	air自适应阈值的最小值，取值范围[160, 200]，默认值180，精度0.1。
bright_max	air自适应阈值的最大值，取值范围[210, 250]，默认值240，精度0.1。
wt_max	wt自适应的最大值限制，取值范围[0.75, 0.9]，默认值0.9，精度0.001。
air_min	air自适应的最小值限制，取值范围[200, 220]，默认值200，精度0.001。
air_max	air自适应的最大值限制，取值范围[230, 250]，默认值250，精度0.001。
tmax_base	tmax自适应基础值，默认125，对应配置如下，200(131)，210(125)，220(119)，230(114)，240(109)，250(105)，推荐131-105
tmax_off	tmax自适应的固定值，取值范围[0.1, 0.5]，默认值0.1，精度0.1。
tmax_max	tmax自适应的最大值，取值范围[0.1, 0.5]，默认值0.5，精度0.1。
cfg_wt	软件配置wt，图像去雾力度，取值范围[0, 1]，默认值0.8，精度0.001。
cfg_air	软件配置air，大气光系数，取值范围[0, 255]，默认值210，精度0.001。
cfg_tmax	软件配置tmax，去雾的最大值，取值范围[0, 1]，默认值0.2，精度0.001。
bf_weight	两个双边滤波的合成权重，取值范围[0, 1]，默认值0.5，精度0.1。
dc_weitcur	dark channel部分的双边权重，取值范围[0, 1]，默认值1，精度0.1。
range_sigma	双边滤波值域 sigma 值，取值范围[0, 1]，默认值0.4，精度0.1。
space_sigma_pre	以 IIR 数据为参考时，双边滤波空域 sigma 值，取值范围[0, 1]，默认值0.4，精度0.1。
space_sigma_cur	以当前数据为参考时，双边滤波空域 sigma 值，取值范围[0, 1]，默认值0.8，精度0.1。

mDehaze_Setting_V21_t

【说明】
定义手动dehaze功能属性

【定义】

```
typedef struct mDehaze_Setting_V21_s {
    bool             en;
    bool             air_lc_en;
    float            stab_fnum;
    float            sigma;
    float            wt_sigma;
    float            air_sigma;
    float            tmax_sigma;
    float            pre_wet;
    mDehazeDataV21_t DehazeData;
} mDehaze_Setting_V21_t;
```

【成员】

成员名称	描述
en	开关功能
air_lc_en	是否使用 airlight base 对 airlight 进行最小值截止开关
stab_fnum	帧稳定的最大值，取值范围[0,31]，默认值8，精度0.01。
sigma	iir控制的sigma，取值范围[0,255]，默认值6，精度1。
wt_sigma	帧间wt滤波系数，取值范围[0,255]，默认值8，精度1。
air_sigma	帧间air滤波系数，取值范围[0,255]，默认值12，精度1。
tmax_sigma	帧间tmax滤波系数，取值范围[0,1]，默认值0.01，精度0.0001。
pre_wet	参考数据 IIR 滤波系数，取值范围[0,1]，默认值0.01，精度0.0001。
DehazeData	dehaze调试参数

mEnhanceDataV21_t

【说明】
定义手动EnhanceData属性

【定义】

```
typedef struct mEnhanceDataV21_s {
    float enhance_value;
    float enhance_chroma;
} mEnhanceDataV21_t;
```

【成员】

成员名称	描述
enhance_value	通用对比度力度，取值范围[0,16]，默认值1.0，精度0.001。
enhance_chroma	色度的增强调节参数，取值范围[0,16]，默认值1.0，精度0.001。

mEnhance_Setting_V21_t

【说明】
定义手动enhance功能属性

【定义】

```
typedef struct mEnhance_Setting_V21_s {
    bool              en;
    float             enhance_curve[CALIBDB_ADEHAZE_ENHANCE_CURVE_KNOTS_NUM];
    mEnhanceDataV21_t EnhanceData;
} mEnhance_Setting_V21_t;
```

【成员】

成员名称	描述
en	enhance功能开关
enhance_curve	低频曲线
EnhanceData	enhance调试参数

mHistDataV21_t

【说明】
定义手动HistData属性

【定义】

```
typedef struct mHistDataV21_s {
    float hist_gratio;
    float hist_th_off;
    float hist_k;
    float hist_min;
    float hist_scale;
    float cfg_gratio;
} mHistDataV21_t;
```

【成员】

成员名称	描述
hist_gratio	直方图拉伸倍数，直方图均衡控制系数，取值范围[0, 32]，默认值，精度0.01。
hist_th_off	直方图统计阈值，取值范围[0, 255]，默认值64，精度0.01。
hist_k	直方图自适应阈值放大倍数，取值范围[0, 7)，默认值2，精度0.01。
hist_min	直方图统计阈值的最小值，取值范围[0,2)，默认值0.016，精度0.01。
hist_scale	直方图均衡控制系数，取值范围[0, 32]，默认值0.9，精度0.01。
cfg_gratio	软件配置直方图拉伸倍数，直方图均衡控制系数，取值范围[0, 32)，默认值1，精度0.01。

mHist_setting_V21_t

【说明】
定义手动hist功能属性

【定义】

```
typedef struct mHist_setting_V21_s {
    bool           en;
    bool           hist_para_en;
    mHistDataV21_t HistData;
} mHist_setting_V21_t;
```

【成员】

成员名称	描述
en	hist功能开关
hist_para_en	直方图拉伸控制开关
HistData	hist调试参数

mDehazeAttr_t

【说明】
定义手动dehaze模块属性

【定义】

```
typedef struct mDehazeAttr_s {
    bool                   Enable;
    float                  cfg_alpha;
    mDehaze_Setting_V21_t  dehaze_setting;
    mEnhance_Setting_V21_t enhance_setting;
    mHist_setting_V21_t    hist_setting;
} mDehazeAttr_t;
```

【成员】

成员名称	描述
Enable	开关功能
cfg_alpha	手动软件配置占比，取值范围[0,1]，默认值1，精度0.01。
dehaze_setting	手动dehaze功能参数
enhance_setting	手动enhance功能参数
hist_setting	手动hist功能参数

adehaze_sw_V2_t

【说明】
定义dehaze属性

【定义】

```
typedef struct adehaze_sw_V2_s {
    rk_aiq_uapi_sync_t sync;
    dehaze_api_mode_t  mode;
    mDehazeAttr_t      stManual;
    DehazeManuAttr_t   stDehazeManu;
    EnhanceManuAttr_t  stEnhanceManu;
} adehaze_sw_V2_t;
```

【成员】

成员名称	描述
sync	参考rk_aiq_uapi_sync_t定义，参见 **“概述/API说明”** 章节
mode	API模式
stManual	手动dehaze模块参数
stDehazeManu	手动dehaze功能参数
stEnhanceManu	手动enhance功能参数

CPROC

功能描述

CPROC(Color Processing) 提供基本的喜好色调节功能，通过对一定区间内的亮度、对比度、饱和度、色度的调节，达到对喜好色的调节，该模块作用于YUV域图像。

功能级API参考

模块级API参考

rk_aiq_user_api2_acp_SetAttrib

【描述】

设置cproc模块属性

【语法】

```
XCamReturn rk_aiq_user_api2_acp_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,

acp_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	cproc模块属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_acp.h
- 库文件：librkaiq.so

rk_aiq_user_api2_acp_GetAttrib

【描述】

获取cproc模块当前属性

【语法】

```
XCamReturn rk_aiq_user_api2_acp_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,

acp_attrib_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	cproc模块属性	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_acp.h
- 库文件：librkaiq.so

模块级API数据类型

acp_attrib_t

【说明】
定义cproc模块的参数

【定义】

```
typedef struct acp_attrib_s {
    uint8_t brightness; /* 0 ~ 255 */
    uint8_t contrast; /* 0 ~ 255 */
    uint8_t saturation; /* 0 ~ 255 */
    uint8_t hue; /* 0 ~ 255 */
} acp_attrib_t;
```

【成员】

成员名称	描述
brightness	亮度，范围：[0,255]，对应范围：[-128,127]，默认值 128
contrast	对比度，范围：[0,255] , 对应倍数：[0, 1.992]，默认值为 128
saturation	饱和度，范围：[0,255], 对应倍数：[0, 1.992]，默认值为 128
hue	色度，范围：[0,255] ，对应度数[-90,87.188] ，默认值 128

IE

功能描述

IE(Image Effect) 提供图像的特殊效果设置，如黑白效果等。该模块作用于YUV域图像。

功能级API参考

模块级API参考

rk_aiq_user_api2_aie_SetAttrib

【描述】

设置IE模块属性

【语法】

```
XCamReturn rk_aiq_user_api2_aie_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,

aie_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	IE模块属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_aie.h
- 库文件：librkaiq.so

rk_aiq_user_api2_aie_GetAttrib

【描述】

获取IE模块当前属性

【语法】

```
XCamReturn rk_aiq_user_api2_aie_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,

aie_attrib_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	IE模块属性	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_aie.h
- 库文件：librkaiq.so

模块级API数据类型

aie_attrib_t

【说明】
定义IE模块的参数

【定义】

```
typedef struct aie_attrib_s {
     rk_aiq_ie_effect_t mode;
} aie_attrib_t;
```

【成员】

成员名称	描述
mode	效果类型

rk_aiq_ie_effect_t

【说明】
定义IE效果类型

【定义】

```
typedef enum rk_aiq_ie_effect_e {
    RK_AIQ_IE_EFFECT_NONE,
    RK_AIQ_IE_EFFECT_BW,
    RK_AIQ_IE_EFFECT_NEGATIVE,
    RK_AIQ_IE_EFFECT_SEPIA,
    RK_AIQ_IE_EFFECT_EMBOSS,
   RK_AIQ_IE_EFFECT_SKETCH,
    RK_AIQ_IE_EFFECT_SHARPEN, /*!< deprecated */
} rk_aiq_ie_effect_t;
```

【成员】

成员名称	描述
RK_AIQ_IE_EFFECT_NONE	无特殊效果
RK_AIQ_IE_EFFECT_BW	黑白效果
RK_AIQ_IE_EFFECT_NEGATIVE	负片效果
RK_AIQ_IE_EFFECT_SEPIA	深褐色效果
RK_AIQ_IE_EFFECT_EMBOSS	浮雕效果
RK_AIQ_IE_EFFECT_SKETCH	素描效果
RK_AIQ_IE_EFFECT_SHARPEN	不支持

CSM

功能描述

CSM(Color Space Matrix) 可设置RGB到YUV转换时的参数。

功能级API参考

模块级API参考

rk_aiq_user_api2_acsm_SetAttrib

【描述】

设置CSM模块属性

【语法】

```
XCamReturn rk_aiq_user_api2_acsm_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,

rk_aiq_uapi_acsm_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	CSM模块属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_acsm.h
- 库文件：librkaiq.so

rk_aiq_user_api2_aie_GetAttrib

【描述】

获取IE模块当前属性

【语法】

```
XCamReturn rk_aiq_user_api2_acsm_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,

rk_aiq_uapi_acsm_attrib_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	CSM模块属性	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_acsm.h
- 库文件：librkaiq.so

模块级API数据类型

rk_aiq_uapi_acsm_attrib_t

【说明】
定义CSM模块的参数

【定义】

```
typedef struct rk_aiq_uapi_acsm_attrib_s {
    rk_aiq_uapi_sync_t sync;
    rk_aiq_acsm_params_t param;
} rk_aiq_uapi_acsm_attrib_t;
```

【成员】

成员名称	描述
sync	参考rk_aiq_uapi_sync_t定义，参见 **“概述/API说明”** 章节
rk_aiq_acsm_params_t	csm参数

rk_aiq_acsm_params_t

【说明】
定义CSM模块参数

【定义】

```
typedef struct __csm_param {^M
    // M4_BOOL_DESC("op_mode", "RKAiqOPMode_t", "RK_AIQ_OP_MODE_AUTO")
    RKAiqOPMode_t op_mode;
    // M4_NUMBER_DESC("full_range", "u8", M4_RANGE(0,1), "1", M4_DIGIT(0),
0)^M
    bool full_range;
    // M4_NUMBER_DESC("y_offset", "u16", M4_RANGE(0,63), "0", M4_DIGIT(0),
0)^M
     unsigned short y_offset;^M
     // M4_NUMBER_DESC("y_offset", "u16", M4_RANGE(0,255), "0", M4_DIGIT(0),
0)^M
     unsigned short c_offset;^M
     // M4_ARRAY_DESC("coeff", "u32", M4_SIZE(3,3), M4_RANGE(0,511), "0",
M4_DIGIT(4), M4_DYNAMIC(0))
     unsigned int coeff[RK_AIQ_CSM_COEFF_NUM];
 } Csm_Param_t;
```

【成员】

成员名称	描述
op_mode	模式，RK_AIQ_OP_MODE_AUTO 或者 RK_AIQ_OP_MODE_MANUAL，AUTO时由AIQ决定，目前为芯片default值
full_range	是否输出 full range，默认值：TRUE
y_offset	亮度偏移值，范围：[0,63]，默认值：0
c_offset	色度偏移值，范围：[0,255]，默认值：0
coeff	RGB到YUV的3x3转换矩阵，范围：[0,511]，默认值： [0x021, 0x040, 0x00d, 0x1ed, 0x1db, 0x038, 0x038, 0x1d1, 0x1f7]

Sharpen

功能描述

Sharpen 模块用于增强图像的清晰度，包括调节图像边缘的锐化属性和增强图像的细节和纹理。

功能级API参考

rk_aiq_uapi2_setSharpness

【描述】

设置锐化等级。

【语法】

```
XCamReturn rk_aiq_uapi2_setSharpness(const rk_aiq_sys_ctx_t* ctx, unsigned int level);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
level	锐化等级取值范围：[0,100]默认值为50	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_getSharpness

【描述】

获取锐化等级。

【语法】

```
XCamReturn rk_aiq_uapi2_getSharpness(const rk_aiq_sys_ctx_t* ctx, unsigned int*
level);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
level	锐化等级	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

模块级API参考

rk_aiq_user_api2_asharpV4_SetAttrib

【描述】
设置锐化算法属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_asharpV4_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_sharp_attrib_v4_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去噪的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_asharp_v4.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_asharpV4_GetAttrib

【描述】
获取锐化算法属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_asharpV4_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_sharp_attrib_v4_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去噪的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_asharp_v4.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_asharpV4_SetStrength

【描述】
设置锐化力度。

【语法】

```
XCamReturn
rk_aiq_user_api2_asharpV4_SetStrength(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_sharp_strength_v4_t *pStrength);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
pStrength	锐化强度结构体指针，结构体中percent取值范围0.0-1.0	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_asharp_v4.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

rk_aiq_user_api2_asharpV4_GetStrength

【描述】
获取锐化力度。

【语法】

```
XCamReturn
rk_aiq_user_api2_asharpV4_GetStrength(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_sharp_strength_v4_t *pStrength);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
pStrength	锐化强度结构体指针，结构体中percent取值范围0.0-1.0	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_asharp_v4.h、RkAiqHandleIntV3x.h
- 库文件：librkaiq.so

模块级API数据类型

rk_aiq_sharp_attrib_v4_t

【说明】
定义锐化模块的参数

【定义】

```
typedef struct rk_aiq_sharp_attrib_v4_s {
    rk_aiq_uapi_sync_t sync;
    Asharp4_OPMode_t eMode;
    Asharp_Auto_Attr_V4_t stAuto;
    Asharp_Manual_Attr_V4_t stManual;
} rk_aiq_sharp_attrib_v4_t;
```

【成员】

成员名称	描述
sync	同步异步模式选择，参见 **“概述/API说明”** 章节
eMode	锐化模块属性
stAuto	锐化模块自动模式参数
stManual	锐化模块手动模式参数

Asharp4_OPMode_t

【说明】

【定义】

```
typedef enum Asharp4_OPMode_e {
    ASHARP4_OP_MODE_INVALID           = 0,
    ASHARP4_OP_MODE_AUTO              = 1,
    ASHARP4_OP_MODE_MANUAL            = 2,
    ASHARP4_OP_MODE_REG_MANUAL        = 3,
    ASHARP4_OP_MODE_MAX
} Asharp4_OPMode_t;
```

【成员】

成员名称	描述
ASHARP4_OP_MODE_INVALID	锐化模块无效模式
ASHARP4_OP_MODE_AUTO	锐化模块自动模式
ASHARP4_OP_MODE_MANUAL	锐化模块手动模式的算法设置
ASHARP4_OP_MODE_REG_MANUAL	锐化模块手动模式的寄存器设置
ASHARP4_OP_MODE_MAX	锐化模块模式最大值，是一个无效模式

Asharp_Auto_Attr_V4_t

【说明】
定义锐化模块的自动模式各个iso对应算法属性参数

【定义】

```
typedef struct Asharp_Auto_Attr_V4_s
{
    RK_SHARP_Params_V4_t stParams;
    RK_SHARP_Params_V4_Select_t stSelect;

} Asharp_Auto_Attr_V4_t;
```

【成员】

成员名称	描述
stParams	sharp模块各个iso对应算法属性参数
stSelect	sharp模块根据当前iso计算出来属性参数

Asharp_Manual_Attr_V4_t

【说明】
定义锐化模块的手动属性

【定义】

```
typedef struct Asharp_Manual_Attr_V4_s
{
    RK_SHARP_Params_V4_Select_t stSelect;

    RK_SHARP_Fix_V4_t stFix;

} Asharp_Manual_Attr_V4_t;
```

【成员】

成员名称	描述
stSelect	sharp手动设置算法参数
stFix	sharp手动模式下寄存器值

RK_SHARP_Params_V4_t

【说明】
定义锐化模块的手动属性

【定义】

```
typedef struct RK_SHARP_Params_V4_s
{
    int enable;

    int iso[RK_SHARP_V4_MAX_ISO_NUM];
    short luma_point        [RK_SHARP_V4_LUMA_POINT_NUM];
    short luma_sigma        [RK_SHARP_V4_MAX_ISO_NUM]
[RK_SHARP_V4_LUMA_POINT_NUM];
    float pbf_gain          [RK_SHARP_V4_MAX_ISO_NUM];
    float pbf_add           [RK_SHARP_V4_MAX_ISO_NUM];
    float pbf_ratio         [RK_SHARP_V4_MAX_ISO_NUM];
    float gaus_ratio        [RK_SHARP_V4_MAX_ISO_NUM];
    float sharp_ratio       [RK_SHARP_V4_MAX_ISO_NUM];
    short hf_clip        [RK_SHARP_V4_MAX_ISO_NUM][RK_SHARP_V4_LUMA_POINT_NUM];
    float bf_gain           [RK_SHARP_V4_MAX_ISO_NUM];
    float bf_add            [RK_SHARP_V4_MAX_ISO_NUM];
    float bf_ratio          [RK_SHARP_V4_MAX_ISO_NUM];
    short local_sharp_strength  [RK_SHARP_V4_MAX_ISO_NUM]
[RK_SHARP_V4_LUMA_POINT_NUM];

    float prefilter_coeff[RK_SHARP_V4_MAX_ISO_NUM][RK_SHARP_V4_PBF_DIAM *
RK_SHARP_V4_PBF_DIAM];
    float GaussianFilter_coeff   [RK_SHARP_V4_MAX_ISO_NUM][RK_SHARP_V4_RF_DIAM *
RK_SHARP_V4_RF_DIAM];
    float hfBilateralFilter_coeff    [RK_SHARP_V4_MAX_ISO_NUM]
[RK_SHARP_V4_BF_DIAM * RK_SHARP_V4_BF_DIAM];


} RK_SHARP_Params_V4_t;
```

【成员】

参数名称	参数类型	简要说明
Enable	调试参数	Sharp模块使能开关。 1：模块打开，0：模块关闭。
iso	调试参数	不同iso档位，对应不同调试参数。目前仅支持13档。
pbf_gain	调试参数	预滤波 sigma 乘以的比例，值越大，滤波越强，噪声越小，细节更少。 取值范围[0.0, 2.0], 默认值1.0 。
pbf_add	调试参数	预滤波 sigma 叠加的偏移，值越大，滤波越强，噪声越小，细节更少。 取值范围[0, 1023], 默认值0。
pbf_ratio	调试参数	预滤波融合权重，值越大，滤波越强，噪声越小，细节更少。 取值范围[0.0, 1.0], 默认值0.5。
gaus_ratio	调试参数	高频双边滤波的导向图像为高斯滤波与原图融合的结果。 值越大，高斯双边滤波的导向权重更大。 取值范围[0.0, 1.0], 默认值0。
sharp_ratio	调试参数	锐化强度，值越大，锐化越强。 取值范围[0.0, 16], 默认值6。
bf_gain	调试参数	高频双边滤波 sigma 乘以的比例， 值越大，滤波越强，噪声越小，细节更少。 取值范围[0.0, 2.0], 默认值1.0 。
bf_add	调试参数	高频双边滤波 sigma 叠加的偏移。值越大，滤波越强，噪声越小，细节更少。 取值范围[0, 1023], 默认值0。
bf_add	调试参数	高频双边滤波融合权重。值越大，滤波越强，噪声越小，细节更少。 取值范围[0.0, 1.0], 默认值0.5。
luma_point / luma_sigma	调试参数	不同pixel亮度，对应不同噪声sigma曲线。 luma_point为曲线亮度值，取值范围[0, 1023] 。 luma_sigma为噪声强度值，取值范围[0, 1023] 。
luma_point / hf_clip	调试参数	不同pixel亮度高频值 clip 的范围。 值越大，允许的最大锐化强度越强。 取值范围[0, 1023]。默认值256。
luma_point / local_sharp_strength	调试参数	计算不同pixel亮度，高频叠加权重的比例。 值越大，允许叠加的高频越大，图像越锐化。 取值范围[0, 1023]。默认值512。
prefilter_coeff	调试参数	预滤波算子。

参数名称	参数类型	简要说明
GaussianFilter_coeff	调试参数	高斯滤波算子。
hfBilateralFilter_coeff	调试参数	高频双边滤波算子。

RK_SHARP_Params_V4_Select_t

【说明】
定义锐化模块的手动模式下算法属性

【定义】

```
typedef struct RK_SHARP_Params_V4_Select_s
{
    int enable;

    short luma_point    [RK_SHARP_V4_LUMA_POINT_NUM];
    short luma_sigma    [RK_SHARP_V4_LUMA_POINT_NUM];
    float pbf_gain      ;
    float pbf_add       ;
    float pbf_ratio     ;
    float gaus_ratio    ;
    float sharp_ratio   ;
    short hf_clip       [RK_SHARP_V4_LUMA_POINT_NUM];
    float bf_gain       ;
    float bf_add        ;
    float bf_ratio      ;
    short local_sharp_strength        [RK_SHARP_V4_LUMA_POINT_NUM];

    float prefilter_coeff[RK_SHARP_V4_PBF_DIAM * RK_SHARP_V4_PBF_DIAM];
    float GaussianFilter_coeff   [RK_SHARP_V4_RF_DIAM * RK_SHARP_V4_RF_DIAM];
    float hfBilateralFilter_coeff    [RK_SHARP_V4_BF_DIAM *
RK_SHARP_V4_BF_DIAM];


} RK_SHARP_Params_V4_Select_t;
```

【成员】

参数名称	参数类型	简要说明
Enable	调试参数	Sharp模块使能开关。 1：模块打开，0：模块关闭。
iso	调试参数	不同iso档位，对应不同调试参数。目前仅支持13档。
pbf_gain	调试参数	预滤波 sigma 乘以的比例，值越大，滤波越强，噪声越小，细节更少。 取值范围[0.0, 2.0], 默认值1.0 。
pbf_add	调试参数	预滤波 sigma 叠加的偏移，值越大，滤波越强，噪声越小，细节更少。 取值范围[0, 1023], 默认值0。
pbf_ratio	调试参数	预滤波融合权重，值越大，滤波越强，噪声越小，细节更少。 取值范围[0.0, 1.0], 默认值0.5。
gaus_ratio	调试参数	高频双边滤波的导向图像为高斯滤波与原图融合的结果。 值越大，高斯双边滤波的导向权重更大。 取值范围[0.0, 1.0], 默认值0。
sharp_ratio	调试参数	锐化强度，值越大，锐化越强。 取值范围[0.0, 16], 默认值6。
bf_gain	调试参数	高频双边滤波 sigma 乘以的比例， 值越大，滤波越强，噪声越小，细节更少。 取值范围[0.0, 2.0], 默认值1.0 。
bf_add	调试参数	高频双边滤波 sigma 叠加的偏移。值越大，滤波越强，噪声越小，细节更少。 取值范围[0, 1023], 默认值0。
bf_add	调试参数	高频双边滤波融合权重。值越大，滤波越强，噪声越小，细节更少。 取值范围[0.0, 1.0], 默认值0.5。
luma_point / luma_sigma	调试参数	不同pixel亮度，对应不同噪声sigma曲线。 luma_point为曲线亮度值，取值范围[0, 1023] 。 luma_sigma为噪声强度值，取值范围[0, 1023] 。
luma_point / hf_clip	调试参数	不同pixel亮度高频值 clip 的范围。 值越大，允许的最大锐化强度越强。 取值范围[0, 1023]。默认值256。
luma_point / local_sharp_strength	调试参数	计算不同pixel亮度，高频叠加权重的比例。 值越大，允许叠加的高频越大，图像越锐化。 取值范围[0, 1023]。默认值512。
prefilter_coeff	调试参数	预滤波算子。

参数名称	参数类型	简要说明
GaussianFilter_coeff	调试参数	高斯滤波算子。
hfBilateralFilter_coeff	调试参数	高频双边滤波算子。

RK_SHARP_Fix_V4_t

【说明】
定义锐化模块的手动模式下寄存器配置

【定义】

```
typedef struct RK_SHARP_Fix_V4_s
{
    // SHARP_SHARP_EN (0x0000)
    uint8_t sharp_clk_dis;
    uint8_t sharp_exgain_bypass;
    uint8_t sharp_center_mode;
    uint8_t sharp_bypass;
    uint8_t sharp_en;

    // SHARP_SHARP_RATIO  (0x0004)
    uint8_t sharp_sharp_ratio;
    uint8_t sharp_bf_ratio;
    uint8_t sharp_gaus_ratio;
    uint8_t sharp_pbf_ratio;

    // SHARP_SHARP_LUMA_DX (0x0008)
    uint8_t sharp_luma_dx[7];

    // SHARP_SHARP_PBF_SIGMA_INV_0 (0x000c - 0x0014)
    uint16_t sharp_pbf_sigma_inv[8];

    // SHARP_SHARP_BF_SIGMA_INV_0 (0x0018 -  0x0020)
    uint16_t sharp_bf_sigma_inv[8];

    // SHARP_SHARP_SIGMA_SHIFT (0x00024)
    uint8_t sharp_bf_sigma_shift;
    uint8_t sharp_pbf_sigma_shift;

    // SHARP_SHARP_EHF_TH_0 (0x0028 -  0x0030)
    uint16_t sharp_ehf_th[8];

    // SHARP_SHARP_CLIP_HF_0 (0x0034 -  0x003c)
    uint16_t sharp_clip_hf[8];

    // SHARP_SHARP_PBF_COEF (0x00040)
    uint8_t sharp_pbf_coef[3];

    // SHARP_SHARP_BF_COEF (0x00044)
    uint8_t sharp_bf_coef[3];
```

```
    // SHARP_SHARP_GAUS_COEF (0x00048 - 0x0004c)
    uint8_t sharp_gaus_coef[6];

} RK_SHARP_Fix_V4_t;
```

【成员】

参数名称	参数说明
sharp_clk_dis	sharp模块clk gate设置，暂不支持此设置, 默认为0。
sharp_exgain_bypass	外部local gain模块是否bypass，暂不支持此设置，默认为0.
sharp_center_mode	是否以图像中心点为坐标进行锐化。暂不支持配置，默认是为0.
sharp_bypass	锐化模块bypass 1 :bypass ; 0 :not bypass
sharp_en	锐化模块使能 1:enable ; 0 :disable
sharp_sharp_ratio	sharp_ratio：锐化强度， 0~15.9
sharp_bf_ratio	双边滤波融合权重， 0~1.0
sharp_gaus_ratio	gaus_ratio：高斯滤波融合权重，取值范围[0.0, 1.0]
sharp_pbf_ratio	双边预滤波融合权重 取值范围[0.0, 1.0], 默认值0.5
sharp_luma_dx7	点6和点7之间的距离。实际距离等于2^sw_sharp_luma_dx7。
sharp_luma_dx6	点5和点6之间的距离。实际距离等于2^sw_sharp_luma_dx6。
sharp_luma_dx5	点4和点5之间的距离。实际距离等于2^sw_sharp_luma_dx5。
sharp_luma_dx4	点3和点4之间的距离。实际距离等于2^sw_sharp_luma_dx4。
sharp_luma_dx3	点2和点3之间的距离。实际距离等于2^sw_sharp_luma_dx3。
sharp_luma_dx2	点1和点2之间的距离。实际距离等于2^sw_sharp_luma_dx2。
sharp_luma_dx1	0点和1点之间的距离。实际距离等于2^sw_sharp_luma_dx1。 注意：总距离必须等于 1024。
sharp_pbf_sigma_inv	Sharp pbf sigma inverse
sharp_bf_sigma_inv	Sharp bf sigma inverse
sharp_pbf_coef	锐化模块双边与滤波滤波核
sharp_bf_coef	双边滤波核
sharp_gaus_coef	高斯滤波核

rk_aiq_sharp_strength_v4_t

【说明】
定义锐化模块的去噪强度配置结构体

【定义】

```
typedef struct rk_aiq_sharp_strength_v4_s {
    rk_aiq_uapi_sync_t sync;
    float percent;
} rk_aiq_sharp_strength_v4_t;
```

【成员】

成员名称	描述
sync	同步异步模式选择，参见 **“概述/API说明”** 章节
percent	锐化强度，取值范围0.0-1.0。

Gamma

功能描述

Gamma 模块对图像进行亮度空间非线性转换以适配输出设备。RK3588支持49点log域gamma曲线，其横坐标为：

```
int X_isp30[49] = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 16, 20, 24, 28, 32,
40, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256, 320, 384, 448, 512, 640,
768, 896, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584,
3840, 4095};
```

功能级API参考

rk_aiq_uapi2_setGammaCoef

【描述】

通过GammaCoef和SlopeAtZero快速设置gamma曲线。其gamma曲线生成方式如下：

```
for(int i = 0; i < 49; i++) {
    Y_isp30[i] = 4095 * pow(X_isp30[i] / 4095, 1 / GammaCoef + SlopeAtZero);
    Y_isp30[i] = LIMIT_VALUE(Y_isp30[i], 4095, 0);
 }
```

【语法】

```
XCamReturn rk_aiq_uapi2_setGammaCoef(const rk_aiq_sys_ctx_t* ctx, float
GammaCoef, float SlopeAtZero);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
GammaCoef	gamma系数，取值范围[0,100]，默认值2.2，精度0.01	输入
SlopeAtZero	暗区斜率，取值范围[-0.05,0.05]，默认值0，精度0.001	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【说明】

Api中Gamma曲线未按照场景进行切换，若场景变化，请重新通过api设置gamma曲线。

功能级API数据类型

模块级API参考

rk_aiq_user_api2_agamma_SetAttrib

【描述】
设定 Gamma软件属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_agamma_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_gamma_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	Gamma软件属性结构体	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_agamma.h
- 库文件：librkaiq.so

【说明】

Api中Gamma曲线未按照场景进行切换，若场景变化，请重新通过api设置gamma曲线。

rk_aiq_user_api2_agamma_GetAttrib

【描述】
获取 Gamma软件属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_agamma_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_gamma_attrib_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	Gamma软件属性结构体	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_agamma.h
- 库文件：librkaiq.so

【说明】

模块级API数据类型

rk_aiq_gamma_op_mode_t

【说明】
定义Gamma工作模式

【定义】

```
typedef enum rk_aiq_gamma_op_mode_s {
    RK_AIQ_GAMMA_MODE_OFF              = 0,
    RK_AIQ_GAMMA_MODE_MANUAL           = 1,
    RK_AIQ_GAMMA_MODE_FAST             = 2,
} rk_aiq_gamma_op_mode_t;
```

【成员】

成员名称	描述
RK_AIQ_GAMMA_MODE_OFF	Api关闭模式
RK_AIQ_GAMMA_MODE_MANUAL	Api手动模式
RK_AIQ_GAMMA_MODE_FAST	Api快速模式

Agamma_api_Fast_t

【说明】
定义快速模式下Gamma属性

通过GammaCoef和SlopeAtZero快速设置gamma曲线。其gamma曲线生成方式如下：

```
for(int i = 0; i < 49; i++) {
    Y_isp30[i] = 4095 * pow(X_isp30[i] / 4095, 1 / GammaCoef + SlopeAtZero);
    Y_isp30[i] = LIMIT_VALUE(Y_isp30[i], 4095, 0);
 }
```

【定义】

```
typedef struct Agamma_api_Fast_s {
    bool  en;
    float GammaCoef;
    float SlopeAtZero;
} Agamma_api_Fast_t;
```

【成员】

成员名称	描述
en	开关功能
GammaCoef	gamma系数，取值范围[0,100]，默认值2.2，精度0.01
SlopeAtZero	暗区斜率，取值范围[-0.05,0.05]，默认值0，精度0.001

GammaType_t

【说明】
定义Gamma曲线X轴间距类型属性

【定义】

```
typedef enum GammaType_e {
   GAMMATYPE_LOG = 0,
   GAMMATYPE_EQU = 1,
} GammaType_t;
```

【成员】

成员名称	描述
GAMMATYPE_LOG	非等间距
GAMMATYPE_EQU	等间距

Agamma_api_manualV21_t

【说明】
定义RK356x下手动Gamma属性

【定义】

```
typedef struct Agamma_api_manualV21_s {
   bool        Gamma_en;
   GammaType_t Gamma_out_segnum;
   uint16_t    Gamma_out_offset;
   uint16_t    Gamma_curve[CALIBDB_AGAMMA_KNOTS_NUM];
} Agamma_api_manualV21_t;
```

【成员】

成员名称	描述
Gamma_en	开关功能
Gamma_out_segnum	手动Gamma曲线X轴间距类型，默认值GAMMATYPE_LOG
Gamma_out_offset	手动Gamma曲线修正系数，取值范围[-2048,2048]，默认值0，精度1。
Gamma_curve	手动Gamma曲线，取值范围[0,4095]，精度1。

Agamma_api_manualV30_t

【说明】
定义RK3588下手动Gamma属性

【定义】

```
typedef struct Agamma_api_manualV30_s {
   bool     Gamma_en;
   uint16_t Gamma_out_offset;
   uint16_t Gamma_curve[CALIBDB_AGAMMA_KNOTS_NUM_V30];
} Agamma_api_manualV30_t;
```

【成员】

成员名称	描述
Gamma_en	开关功能
Gamma_out_offset	手动Gamma曲线修正系数，取值范围[-2048,2048]，默认值0，精度1。
Gamma_curve	手动Gamma曲线，取值范围[0,4095]，精度1。

rk_aiq_gamma_attrV21_t

【说明】
定义RK356x下Gamma属性

【定义】

```
typedef struct rk_aiq_gamma_attrV21_s {
   rk_aiq_gamma_op_mode_t mode;
   Agamma_api_manualV21_t stManual;
   Agamma_api_Fast_t      stFast;
} rk_aiq_gamma_attrV21_t;
```

【成员】

成员名称	描述
mode	API模式
stManual	手动Gamma参数
stFast	快速模式参数

rk_aiq_gamma_attrV30_t

【说明】
定义RK3588下Gamma属性

【定义】

```
typedef struct rk_aiq_gamma_attrV30_s {
   rk_aiq_gamma_op_mode_t mode;
   Agamma_api_manualV30_t stManual;
   Agamma_api_Fast_t      stFast;
} rk_aiq_gamma_attrV30_t;
```

【成员】

成员名称	描述
mode	API模式
stManual	手动Gamma参数
stFast	快速模式参数

rk_aiq_gamma_attr_t

【说明】
定义Gamma属性

【定义】

```
typedef struct rk_aiq_gamma_attr_s {
    rk_aiq_uapi_sync_t     sync;
    rk_aiq_gamma_attrV21_t atrrV21;
    rk_aiq_gamma_attrV30_t atrrV30;
} rk_aiq_gamma_attr_t;
```

【成员】

成员名称	描述
sync	参考rk_aiq_uapi_sync_t定义，参见 **“概述/API说明”** 章节
atrrV21	RK356x芯片下手动gamma属性
atrrV30	RK3588芯片下手动gamma属性

CCM

功能描述

CCM (Color Correction Matrix) 模块对图像进行颜色校正处理。

功能级API参考

rk_aiq_uapi2_setCCMMode

【描述】
设置CCM工作模式。

【语法】

```
XCamReturn rk_aiq_uapi2_setCCMMode(const rk_aiq_sys_ctx_t* ctx, opMode_t mode);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mode	工作模式； 取值范围：{OP_AUTO, OP_MANUAL}	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_accm_module.cpp 中 "sample_set_ccm_manual" / "sample_set_ccm_auto" 设置手动/自动模式。

rk_aiq_uapi2_getCCMMode

【描述】
获取CCM工作模式。

【语法】

```
XCamReturn rk_aiq_uapi2_getCCMMode(const rk_aiq_sys_ctx_t* ctx, opMode_t *mode);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mode	工作模式； 取值范围：{OP_AUTO, OP_MANUAL}	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_accm_module.cpp 中 "sample_get_ccm_mode" 获取工作模式。

rk_aiq_uapi2_setMCcCoef

【描述】
设置Manual模式下的CCM矩阵，包括色彩校正矩阵和R/G/B通道偏移。

【语法】

```
XCamReturn rk_aiq_uapi2_setMCcCoef(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_ccm_matrix_t *mccm);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mccm	Manual模式下的CCM矩阵，包括色彩校正矩阵和R/G/B通道偏移	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_accm_module.cpp 中 "sample_set_ccm_manual_matrix" 设置手动CCM矩阵。

rk_aiq_uapi2_getMCcCoef

【描述】
获取CCM矩阵，包括色彩校正矩阵和R/G/B通道偏移。

【语法】

```
XCamReturn rk_aiq_uapi2_getMCcCoef(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_ccm_matrix_t *mccm);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mccm	CCM矩阵，包括色彩校正矩阵和R/G/B通道偏移	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_accm_module.cpp 中 "sample_get_ccm_manual_matrix" 获取CCM矩阵。

rk_aiq_uapi2_getACcmSat

【描述】
获取自动模式下的CCM饱和度。

【语法】

```
XCamReturn rk_aiq_uapi2_getACcmSat(const rk_aiq_sys_ctx_t* ctx,  float
*finalsat);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
finalsat	饱和度，手动模式为0	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_accm_module.cpp 中 "sample_get_accm_sat" 获取自动模式下CCM饱和度。

rk_aiq_uapi2_getACcmMatrixName

【描述】
获取自动模式下CCM矩阵名。

【语法】

```
XCamReturn rk_aiq_uapi2_getACcmMatrixName(const rk_aiq_sys_ctx_t* ctx, char
**ccm_name);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
ccm_name	CCM矩阵名	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_accm_module.cpp 中 "sample_get_accm_matrix_name" 获取自动模式下CCM饱和度。

功能级API数据类型

opMode_t

【说明】
定义CCM工作模式。

【定义】

```
typedef enum opMode_e {
    OP_AUTO = 0,
    OP_MANUAL = 1,
    OP_INVAL
} opMode_t;
```

【成员】

成员名称	描述
OP_AUTO	自动模式
OP_MANUAL	手动模式
OP_INVAL	无效值

rk_aiq_ccm_matrix_t

【说明】
定义CCM矩阵。

【定义】

```
typedef struct rk_aiq_ccm_matrix_s {
    float  ccMatrix[9];
    float  ccOffsets[3];
} rk_aiq_ccm_matrix_t;
```

【成员】

成员名称	描述
ccMatrix	手动模式下色彩校正矩阵； 取值范围：[-8, 7.992]
ccOffsets	手动模式下R\G\B分量偏移； 取值范围：[-4096, 4095]

模块级API参考

rk_aiq_user_api2_accm_SetAttrib

【描述】
设置CCM属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_accm_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_ccm_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	CCM的属性参数	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_accm.h、rk_aiq_uapi_accm_int.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_accm_module.cpp 中 "sample_ccm_setCcmAttr" 等设置CCM属性。

rk_aiq_user_api2_accm_GetAttrib

【描述】
获取CCM属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_accm_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_ccm_attrib_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	CCM的参数属性	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_accm.h、rk_aiq_uapi_accm_int.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_accm_module.cpp 中 "sample_ccm_setCcmAttr" 等获取CCM属性。

rk_aiq_user_api2_accm_QueryCcmInfo

【描述】
查询CCM信息。

【语法】

```
XCamReturn
rk_aiq_user_api2_accm_QueryCcmInfo(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_ccm_querry_info_t *ccm_querry_info);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
ccm_querry_info	CCM的查询内容	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_accm.h、rk_aiq_uapi_accm_int.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_accm_module.cpp 中 "sample_ccm_setCcmAttr" 查询CCM信息。

模块级API数据类型

rk_aiq_ccm_op_mode_t

【说明】
定义CCM工作模式。

【定义】

```
typedef enum rk_aiq_ccm_op_mode_s {
    RK_AIQ_CCM_MODE_INVALID                     = 0,
    RK_AIQ_CCM_MODE_MANUAL                      = 1,
    RK_AIQ_CCM_MODE_AUTO                        = 2,
    RK_AIQ_CCM_MODE_TOOL                        = 3,
    RK_AIQ_CCM_MODE_MAX
} rk_aiq_ccm_op_mode_t;
```

【成员】

成员名称	描述
RK_AIQ_CCM_MODE_INVALID	CCM无效模式
RK_AIQ_CCM_MODE_MANUAL	CCM手动模式
RK_AIQ_CCM_MODE_AUTO	CCM自动模式
RK_AIQ_CCM_MODE_TOOL	CCM工具模式（暂时无用）

rk_aiq_ccm_mccm_attrib_t

【说明】
定义手动CCM属性。

【定义】

```
typedef struct rk_aiq_ccm_mccm_attrib_s {
    float  ccMatrix[9];
    float  ccOffsets[3];
    float  y_alpha_curve[17];
    float low_bound_pos_bit;
} rk_aiq_ccm_mccm_attrib_t;
```

【成员】

成员名称	描述
ccMatrix	色彩校正矩阵； 取值范围：[-8, 7.992]
ccOffsets	R\G\B分量偏移； 取值范围：[-4096, 4095]
y_alpha_curve	像素点亮度（12bit）相关颜色校正强度调节曲线的强度配置； 取值范围：[0x0, 0x400]； 0x400表示1倍强度，0x0表示不校正
low_bound_pos_bit	像素点亮度（12bit）相关颜色校正强度调节曲线拐点亮度值，即可调节亮度区间为：[0, 2^low_bound_pos_bit] 和 [4095-2^low_bound_pos_bit， 4095]； 取值范围：[0,10]

rk_aiq_ccm_color_inhibition_t

【说明】
定义CCM色彩抑制水平。

【定义】

```
typedef struct rk_aiq_ccm_color_inhibition_s {
    float sensorGain[RK_AIQ_ACCM_COLOR_GAIN_NUM];
    float level[RK_AIQ_ACCM_COLOR_GAIN_NUM];
} rk_aiq_ccm_color_inhibition_t;
```

【成员】

成员名称	描述
sensorGain	曝光增益分量
level	色彩抑制水平； 取值范围：[0,100]； 默认值为0

rk_aiq_ccm_color_saturation_t

【说明】
定义自动CCM色彩饱和度水平。

【定义】

```
typedef struct rk_aiq_ccm_color_saturation_s {
    float sensorGain[RK_AIQ_ACCM_COLOR_GAIN_NUM];
    float level[RK_AIQ_ACCM_COLOR_GAIN_NUM];
} rk_aiq_ccm_color_saturation_t;
```

【成员】

成员名称	描述
sensorGain	曝光增益分量
level	色彩饱和度水平； 取值范围：[0,100]； 默认值为100

rk_aiq_ccm_accm_attrib_t

【说明】
定义自动CCM属性。

【定义】

```
typedef struct rk_aiq_ccm_accm_attrib_s {
    rk_aiq_ccm_color_inhibition_t color_inhibition;
    rk_aiq_ccm_color_saturation_t color_saturation;
} rk_aiq_ccm_accm_attrib_t;
```

【成员】

成员名称	描述
color_inhibition	CCM色彩抑制水平
color_saturation	CCM色彩饱和度水平

rk_aiq_ccm_attrib_t

【说明】
定义CCM属性。

【定义】

```
typedef struct rk_aiq_ccm_attrib_s {
    rk_aiq_uapi_sync_t sync;
    bool byPass;
    rk_aiq_ccm_op_mode_t mode;
    rk_aiq_ccm_mccm_attrib_t stManual;
    rk_aiq_ccm_accm_attrib_t stAuto;
    CalibDbV2_Ccm_Para_V2_t stTool;
} rk_aiq_ccm_attrib_t;
```

【成员】

成员名称	描述
sync	同步/异步信号，参考rk_aiq_uapi_sync_t定义说明，参见 **“概述/API说明”** 章节
byPass	跳过模块处理； true 跳过CCM，false CCM使能
mode	工作模式选择； 取值范围：{RK_AIQ_CCM_MODE_MANUAL，RK_AIQ_CCM_MODE_AUTO}，分别表示手动和自动模式
stManual	手动模式下属性参数配置
stAuto	自动模式下属性参数配置
stTool	工具模式下参数配置，无需配置

rk_aiq_ccm_querry_info_t

【说明】
定义CCM查询信息

【定义】

```
typedef struct rk_aiq_ccm_querry_info_s {
    bool ccm_en;
    float  ccMatrix[9];
    float  ccOffsets[3];
    float  y_alpha_curve[CCM_CURVE_DOT_NUM];
    float low_bound_pos_bit;
    float color_inhibition_level;
    float color_saturation_level;
    float finalSat;
    char  ccmname1[25];
    char  ccmname2[25];
} rk_aiq_ccm_querry_info_t;
```

【成员】

成员名称	描述
ccm_en	CCM使能 true 使能，false 不使能
ccMatrix	CCM生效校正矩阵； 取值范围：[-8,7.992]
ccOffsets	生效R\G\B分量偏移； 取值范围：[-4096,4095]
y_alpha_curve	像素点亮度（12bit）相关颜色校正强度调节曲线的强度配置； 取值范围：[0x0, 0x400]； 0x400表示1倍强度，0x0表示不校正
low_bound_pos_bit	像素点亮度（12bit）相关颜色校正强度调节曲线拐点亮度值，即可调节亮度区间为：[0, 2^low_bound_pos_bit] 和 [4095-2^low_bound_pos_bit， 4095]； 取值范围：[0,10]
color_inhibition_level	CCM色彩抑制水平（仅支持自动模式）
color_saturation_level	CCM色彩饱和度水平（仅支持自动模式）
finalSat	当前矩阵对应的饱和度值（仅支持自动模式）
ccmname1	用于插值计算当前矩阵的标定矩阵名（仅支持自动模式）
ccmname2	用于插值计算当前矩阵的标定矩阵名（仅支持自动模式）

3DLUT

功能描述

3DLUT (3D look up table) 模块对图像进行RGB空间的颜色映射处理。

功能级API参考

rk_aiq_uapi2_setLut3dMode

【描述】
设置3DLUT工作模式。

【语法】

```
XCamReturn rk_aiq_uapi2_setLut3dMode(const rk_aiq_sys_ctx_t* ctx, opMode_t mode);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mode	工作模式； 取值范围：{OP_AUTO, OP_MANUAL}	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_a3dlut_module.cpp 中 "sample_set_a3dlut_manual" / "sample_set_a3dlut_auto" 设置手动/自动模式。

rk_aiq_uapi2_getLut3dMode

【描述】
获取3DLUT工作模式。

【语法】

```
XCamReturn rk_aiq_uapi2_getLut3dMode(const rk_aiq_sys_ctx_t* ctx, opMode_t *mode);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mode	工作模式	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_a3dlut_module.cpp 中 "sample_get_a3dlut_mode" 设置手动/自动模式。

rk_aiq_uapi2_setM3dLut

【描述】
设置3DLUT手动3D查找表。

【语法】

```
XCamReturn rk_aiq_uapi2_setM3dLut(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_lut3d_table_t *mlut);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mlut	3D查找表	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_a3dlut_module.cpp 中 "sample_set_a3dlut_manual_lut" 设置3DLUT手动3D查找表。

rk_aiq_uapi2_getM3dLut

【描述】
获取3DLUT 3D查找表。

【语法】

```
XCamReturn rk_aiq_uapi2_getM3dLut(const rk_aiq_sys_ctx_t* ctx,
 rk_aiq_lut3d_table_t *mlut);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mlut	3D查找表	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_a3dlut_module.cpp 中 "sample_get_a3dlut_lut" 获取3DLUT 3D查找表。

rk_aiq_uapi2_getA3dLutStrth

【描述】
获取3DLUT调节强度。

【语法】

```
XCamReturn rk_aiq_uapi2_getA3dLutStrth(const rk_aiq_sys_ctx_t* ctx, float
alpha);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
alpha	3DLUT调节强度	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_a3dlut_module.cpp 中 "sample_get_a3dlut_strength" 获取3DLUT调节强度。

rk_aiq_uapi2_getA3dLutName

【描述】
获取自动模式下3DLUT表名。

【语法】

```
XCamReturn rk_aiq_uapi2_getA3dLutName(const rk_aiq_sys_ctx_t* ctx, char *name);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
name	3DLUT表名	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_a3dlut_module.cpp 中 "sample_get_a3dlut_lutname" 获取自动模式下3DLUT表名。

功能级API数据类型

opMode_t

【说明】
定义3DLUT工作模式。

【定义】

```
typedef enum opMode_e {
    OP_AUTO = 0,
    OP_MANUALl = 1,
    OP_INVAL
} opMode_t;
```

【成员】

成员名称	描述
OP_AUTO	自动模式
OP_MANUALl	手动模式
OP_INVAL	无效值

rk_aiq_lut3d_table_t

【说明】
定义3DLUT 3D查找表。

【定义】

```
typedef struct rk_aiq_lut3d_table_s{
    unsigned short look_up_table_r[729];
    unsigned short look_up_table_g[729];
    unsigned short look_up_table_b[729];
} rk_aiq_lut3d_table_t;
```

【成员】

成员名称	描述
look_up_table_r	手动模式下R通道查找表； 取值范围：[0x0, 0x3ff]
look_up_table_g	手动模式下G通道查找表； 取值范围：[0x0, 0xfff]
look_up_table_b	手动模式下B通道查找表； 取值范围：[0x0, 0x3ff]

模块级API参考

rk_aiq_user_api2_a3dlut_SetAttrib

【描述】
设置3DLUT属性。

【语法】

```
XCamReturn rk_aiq_user_api2_a3dlut_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_lut3d_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	3DLUT的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_a3dlut.h、rk_aiq_uapi_a3dlut_int.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_a3dlut_module.cpp 中 "sample_3dlut_set3dlutAttr" 等设置3DLUT属性。

rk_aiq_user_api2_a3dlut_GetAttrib

【描述】
获取3DLUT属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_a3dlut_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_lut3d_attrib_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	3DLUT的参数属性	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_a3dlut.h、rk_aiq_uapi_a3dlut_int.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_a3dlut_module.cpp 中 "sample_3dlut_get3dlutAttr" 等获取3DLUT属性。

rk_aiq_user_api2_a3dlut_Query3dlutInfo

【描述】
查询3DLUT信息。

【语法】

```
XCamReturn
rk_aiq_user_api2_a3dlut_Query3dlutInfo(const RkAiqAlgoContext *ctx,
rk_aiq_lut3d_querry_info_t *lut3d_querry_info );
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
lut3d_querry_info	3DLUT的查询内容	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_a3dlut.h、rk_aiq_uapi_a3dlut_int.h
- 库文件：librkaiq.so

【示例】

- 参考 sample_a3dlut_module.cpp 中 "sample_query_a3dlut_info" 查询3DLUT信息。

模块级API数据类型

rk_aiq_lut3d_op_mode_t

【说明】
定义3DLUT工作模式

【定义】

```
typedef enum rk_aiq_lut3d_op_mode_s {
    RK_AIQ_LUT3D_MODE_INVALID                       = 0,
    RK_AIQ_LUT3D_MODE_MANUAL                        = 1,
    RK_AIQ_LUT3D_MODE_AUTO                          = 2,
    RK_AIQ_LUT3D_MODE_MAX
} rk_aiq_lut3d_op_mode_t;
```

【成员】

成员名称	描述
RK_AIQ_LUT3D_MODE_INVALID	3DLUT无效模式
RK_AIQ_LUT3D_MODE_MANUAL	3DLUT手动模式
RK_AIQ_LUT3D_MODE_AUTO	3DLUT自动模式

rk_aiq_lut3d_mlut3d_attrib_t

【说明】
定义手动3DLUT属性

【定义】

```
typedef struct rk_aiq_lut3d_mlut3d_attrib_s {
    unsigned short look_up_table_r[729];
    unsigned short look_up_table_g[729];
    unsigned short look_up_table_b[729];
} rk_aiq_lut3d_mlut3d_attrib_t;
```

【成员】

成员名称	描述
look_up_table_r	手动模式下R通道查找表； 取值范围：[0x0, 0x3ff]
look_up_table_g	手动模式下G通道查找表； 取值范围：[0x0, 0xfff]
look_up_table_b	手动模式下B通道查找表； 取值范围：[0x0, 0x3ff]

rk_aiq_lut3d_attrib_t

【说明】
定义3DLUT属性

【定义】

```
typedef struct rk_aiq_lut3d_attrib_s {
    rk_aiq_uapi_sync_t sync;
    bool byPass;
    rk_aiq_lut3d_op_mode_t mode;
    rk_aiq_lut3d_mlut3d_attrib_t stManual;
} rk_aiq_lut3d_attrib_t;
```

【成员】

成员名称	描述
sync	同步/异步信号，参考rk_aiq_uapi_sync_t定义说明，参见 **“概述/API说明”** 章节
byPass	跳过模块处理； true 跳过3DLUT，false 3DLUT使能
mode	工作模式选择； 取值范围：{RK_AIQ_LUT3D_MODE_MANUAL，RK_AIQ_LUT3D_MODE_AUTO}， 分别表示手动和自动模式
stManual	手动模式下属性参数配置

rk_aiq_lut3d_querry_info_t

【说明】
定义3DLUT查询信息

【定义】

```
typedef struct rk_aiq_lut3d_querry_info_s {
    bool lut3d_en;
    float alpha;
    char name[25];
    unsigned short look_up_table_r[729];
    unsigned short look_up_table_g[729];
    unsigned short look_up_table_b[729];
} rk_aiq_lut3d_querry_info_t;
```

【成员】

成员名称	描述
lut3d_en	3DLUT使能； true 使能，false 不使能
alpha	3DLUT调节强度； 取值范围：[0, 1.0]，值越大强度越大； 手动模式下为1.0
name	3DLUT表名，仅支持自动模式
look_up_table_r	R通道查找表； 取值范围：[0x0, 0x3ff]
look_up_table_g	G通道查找表； 取值范围：[0x0, 0xfff]
look_up_table_b	B通道查找表； 取值范围：[0x0, 0x3ff]

LDCH

功能描述

LDCH对存在失真的图像进行复原性处理，该模块只对水平方向的图像畸变进行校正。

功能级API参考

rk_aiq_uapi_setLdchEn

【描述】 水平畸变校正功能开关。

【语法】

```
XCamReturn rk_aiq_uapi2_setLdchEn(const rk_aiq_sys_ctx_t* ctx, bool en);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
en	校正开关	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi2_setLdchCorrectLevel

【描述】 设置水平畸变校正等级。

【语法】

```
XCamReturn rk_aiq_uapi2_setLdchCorrectLevel(const rk_aiq_sys_ctx_t* ctx, int
correctLevel);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
correctLevel	校正等级，取值范围：[0~255]	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_imgproc.h
- 库文件：librkaiq.so

模块级API参考

rk_aiq_user_api2_aldch_SetAttrib

【描述】
设置fec属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_aldch_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_ldch_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	ldch的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_aldch.h
- 库文件：librkaiq.so

rk_aiq_user_api2_aldch_GetAttrib

【描述】
获取fec属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_aldch_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_ldch_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	ldch的参数属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_aldch.h
- 库文件：librkaiq.so

模块级API数据类型

rk_aiq_ldch_attrib_t

【说明】
ldch属性配置

【定义】

```
typedef struct rk_aiq_ldch_cfg_s {
    rk_aiq_uapi_sync_t sync;

    unsigned int en;
    int correct_level;
} rk_aiq_ldch_cfg_t;
```

【成员】

成员名称	描述
sync	接口同步/异步功能，参考rk_aiq_uapi_sync_t定义说明，参见 **“概述/API说明”** 章节
en	使能/关闭ldch
correct_level	设置ldch校正级别：[0~255]

DeBayer

功能描述

DeBayer完成将由 sensor 采集到的，带有 CFA 属性的图像通过插值算法还原成为具有完整像素信息的RGB图像。

该模块支持Bayer raw数据，包含 RGGB、BGGR、GRBG、GBRG 四种 pattern 模式。暂不支持 其他CFA数据，例如：RCCB, RGB-IR等CFA。

模块级API参考

rk_aiq_user_api_adebayer_SetAttrib

【描述】
设置去马赛克属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_adebayer_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
adebayer_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去马赛克属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_adebayer.h
- 库文件：librkaiq.so

rk_aiq_user_api_adebayer_GetAttrib

【描述】
获取去马赛克属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_adebayer_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
adebayer_attrib_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	去马赛克属性	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_uapi2_adebayer_int.h
- 库文件：librkaiq.so

数据类型

rk_aiq_debayer_op_mode_t

【说明】
定义debayer工作模式。

【定义】

```
typedef enum rk_aiq_debayer_op_mode_s {
    RK_AIQ_DEBAYER_MODE_INVALID                     = 0,        /**<
initialization value */
    RK_AIQ_DEBAYER_MODE_MANUAL                      = 1,        /**< run manual
lens shading correction */
    RK_AIQ_DEBAYER_MODE_AUTO                        = 2,        /**< run auto
lens shading correction */
    RK_AIQ_DEBAYER_MODE_MAX
} rk_aiq_debayer_op_mode_t;
```

【成员】

成员名称	描述
RK_AIQ_DEBAYER_MODE_INVALID	Debayer无效模式
RK_AIQ_DEBAYER_MODE_MANUAL	Debayer手动模式
RK_AIQ_DEBAYER_MODE_AUTO	Debayer自动模式

adebayer_attrib_manual_t

【说明】
定义debayer手动模式下的可配置属性。

【定义】

```
typedef struct adebayer_attrib_manual_s {
    int8_t      filter1[5];
    int8_t      filter2[5];
    uint8_t     gain_offset;
    uint8_t     sharp_strength;
    uint8_t     hf_offset;
    uint8_t     offset;
    uint8_t     clip_en;
    uint8_t     filter_g_en;
    uint8_t     filter_c_en;
    uint8_t     thed0;
    uint8_t     thed1;
    uint8_t     dist_scale;
    uint8_t     cnr_strength;
    uint8_t     shift_num;
} adebayer_attrib_manual_t;
```

【成员】

成员名称	描述
filter1	低频梯度滤波器； 取值范围：[-8，7]
filter2	高频梯度滤波器； 取值范围：[-8，7]
gain_offset	计算G通道插值系数中锐化权重时梯度的偏移值； 取值范围：[0，15]
sharp_strength	G通道插值锐化力度，值越大，锐化力度越大； 取值范围：[0，4]
hf_offset	插值方向性控制： 值越小，平坦区域容易出现伪纹理；值越大，边缘区域越容易出现锯齿现象。 取值范围：[0，4095]
offset	G通道clip的offset。值越大， clip范围越大。 取值范围：[0，65535]
clip_en	G通道插值 clip 开关，0：关闭，1：打开；
filter_g_en	G通道散点去除开关，0：关闭，1：打开；
filter_c_en	伪色去除开关，0：关闭，1：打开；
thed0	控制高低频权重选取，值越大选取高频权重概率越小； 取值范围：[0，16]
thed1	控制高低频权重选取，值越大选取低频权重概率越小； 取值范围：[0，16]
dist_scale	控制高低频权重选取，值越大选取高频权重概率越小； 取值范围：[0，16]
cnr_strength	伪色去除力度，值越大，力度越大； 取值范围：[0，9]
shift_num	色差图滤波时上下限加减的偏移值移位大小，值越小，色差 clip 的范围越大； 取值范围[0，4]

adebayer_attrib_auto_t

【说明】
定义debayer自动模式下的可配置属性。

【定义】

```
typedef struct adebayer_attrib_auto_s {
    uint8_t     sharp_strength[9];
    uint8_t     low_freq_thresh;
    uint8_t     high_freq_thresh;
} adebayer_attrib_auto_t;
```

【成员】

成员名称	描述
enhance_strength[9]	不同ISO下的细节纹理增强强度 index 0 - ISO 50 index 1 - ISO 100 index 2 - ISO 200 index 3 - ISO 400 index 4 - ISO 800 index 5 - ISO 1600 index 6 - ISO 3200 index 7 - ISO 6400 index 8 - ISO 12800 值越大细节细碎度和清晰度越好，同时伪细节也会相应增强 取值范围：[0~7]
low_freq_thresh	低频权重选取阈值 值越大选取低频权重的概率越小 取值范围：[0~15]
high_freq_thresh	高频权重选取阈值 值越大选取高频权重的概率越小 取值范围：[0~15]

adebayer_attrib_t

【说明】
定义debayer可配置属性。

【定义】

```
typedef struct adebayer_attrib_s {
    rk_aiq_uapi_sync_t          sync;

    uint8_t                     enable;
    rk_aiq_debayer_op_mode_t    mode;
    adebayer_attrib_manual_t    stManual;
    adebayer_attrib_auto_t      stAuto;
} adebayer_attrib_t;
```

【成员】

成员名称	描述
sync	同步/异步接口模式，参考rk_aiq_uapi_sync_t定义说明，参见 **“概述/API说明”** 章节
enable	debayer模块使能开关，0：bypass，1：打开；
mode	debayer工作模式
stManual	debaye手动控制模式下的配置属性
stAuto	debayer自动控制模式下的配置属性

DPCC

功能描述

检测并消除图像中的坏点。建议详见《Rockchip_Tuning_Guide_ISP30》文档中4.6章节"DPCC"中“功能描述”。

模块级API参考

rk_aiq_user_api2_adpcc_SetAttrib

【描述】
设定 DPCC软件属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_adpcc_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_dpcc_attrib_V20_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	DPCC软件属性结构体	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_adpcc.h
- 库文件：librkaiq.so

【说明】

rk_aiq_user_api2_adpcc_GetAttrib

【描述】
获取 DPCC软件属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_adpcc_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_dpcc_attrib_V20_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	DPCC软件属性结构体	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_adpcc.h
- 库文件：librkaiq.so

【说明】

数据类型

rk_aiq_dpcc_attrib_V20_t

【说明】
定义坏点消除属性。

【定义】

```
typedef struct rk_aiq_dpcc_attrib_V20_s {
    AdpccOPMode_t eMode;
    Adpcc_Auto_Attr_t stAuto;
    Adpcc_Manual_Attr_t stManual;
    rk_aiq_uapi_sync_t sync;
} rk_aiq_dpcc_attrib_V20_t;
```

【成员】

成员名称	描述
eMode	定义DPCC工作模式，详见**AdpccOPMode_t**说明
stAuto	自动模式的参数设定
stManual	手动模式的参数设定
sync	接口同步/异步功能，参考rk_aiq_uapi_sync_t定义说明，参见 **“概述/API说明”** 章节

AdpccOPMode_t

【说明】
定义DPCC工作模式

【定义】

```
typedef enum AdpccOPMode_e {
    ADPCC_OP_MODE_INVALID        = 0,
    ADPCC_OP_MODE_AUTO           = 1,
    ADPCC_OP_MODE_MANUAL         = 2,
    ADPCC_OP_MODE_TOOL           = 3,
    ADPCC_OP_MODE_MAX
} AdpccOPMode_t;
```

【成员】

成员名称	描述
ADPCC_OP_MODE_INVALID	Api无效模式
ADPCC_OP_MODE_AUTO	Api自动模式
ADPCC_OP_MODE_MANUAL	Api手动模式
ADPCC_OP_MODE_TOOL	Api工具模式
ADPCC_OP_MODE_MAX	

Adpcc_basic_params_select_t

【说明】
定义DPCC基本参数属性

【定义】

```
typedef struct Adpcc_basic_params_select_s
{
    int           iso;
    unsigned char stage1_enable;
    unsigned char grayscale_mode;
    unsigned char enable;
    unsigned char sw_rk_out_sel;
    unsigned char sw_dpcc_output_sel;
    unsigned char stage1_rb_3x3;
    unsigned char stage1_g_3x3;
    unsigned char stage1_incl_rb_center;
    unsigned char stage1_incl_green_center;
    unsigned char stage1_use_fix_set;
    unsigned char stage1_use_set_3;
    unsigned char stage1_use_set_2;
    unsigned char stage1_use_set_1;
    unsigned char sw_rk_red_blue1_en;
    unsigned char rg_red_blue1_enable;
    unsigned char rnd_red_blue1_enable;
    unsigned char ro_red_blue1_enable;
    unsigned char lc_red_blue1_enable;
    unsigned char pg_red_blue1_enable;
    unsigned char sw_rk_green1_en;
    unsigned char rg_green1_enable;
    unsigned char rnd_green1_enable;
    unsigned char ro_green1_enable;
    unsigned char lc_green1_enable;
```

```
    unsigned char pg_green1_enable;
    unsigned char sw_rk_red_blue2_en;
    unsigned char rg_red_blue2_enable;
    unsigned char rnd_red_blue2_enable;
    unsigned char ro_red_blue2_enable;
    unsigned char lc_red_blue2_enable;
    unsigned char pg_red_blue2_enable;
    unsigned char sw_rk_green2_en;
    unsigned char rg_green2_enable;
    unsigned char rnd_green2_enable;
    unsigned char ro_green2_enable;
    unsigned char lc_green2_enable;
    unsigned char pg_green2_enable;
    unsigned char sw_rk_red_blue3_en;
    unsigned char rg_red_blue3_enable;
    unsigned char rnd_red_blue3_enable;
    unsigned char ro_red_blue3_enable;
    unsigned char lc_red_blue3_enable;
    unsigned char pg_red_blue3_enable;
    unsigned char sw_rk_green3_en;
    unsigned char rg_green3_enable;
    unsigned char rnd_green3_enable;
    unsigned char ro_green3_enable;
    unsigned char lc_green3_enable;
    unsigned char pg_green3_enable;
    unsigned char sw_mindis1_rb;
    unsigned char sw_mindis1_g;
    unsigned char line_thr_1_rb;
    unsigned char line_thr_1_g;
    unsigned char sw_dis_scale_min1;
    unsigned char sw_dis_scale_max1;
    unsigned char line_mad_fac_1_rb;
    unsigned char line_mad_fac_1_g;
    unsigned char pg_fac_1_rb;
    unsigned char pg_fac_1_g;
    unsigned char rnd_thr_1_rb;
    unsigned char rnd_thr_1_g;
    unsigned char rg_fac_1_rb;
    unsigned char rg_fac_1_g;
    unsigned char sw_mindis2_rb;
    unsigned char sw_mindis2_g;
    unsigned char line_thr_2_rb;
    unsigned char line_thr_2_g;
    unsigned char sw_dis_scale_min2;
    unsigned char sw_dis_scale_max2;
    unsigned char line_mad_fac_2_rb;
    unsigned char line_mad_fac_2_g;
    unsigned char pg_fac_2_rb;
    unsigned char pg_fac_2_g;
    unsigned char rnd_thr_2_rb;
    unsigned char rnd_thr_2_g;
    unsigned char rg_fac_2_rb;
    unsigned char rg_fac_2_g;
    unsigned char sw_mindis3_rb;
    unsigned char sw_mindis3_g;
    unsigned char line_thr_3_rb;
    unsigned char line_thr_3_g;
    unsigned char sw_dis_scale_min3;
```

```
    unsigned char sw_dis_scale_max3;
    unsigned char line_mad_fac_3_rb;
    unsigned char line_mad_fac_3_g;
    unsigned char pg_fac_3_rb;
    unsigned char pg_fac_3_g;
    unsigned char rnd_thr_3_rb;
    unsigned char rnd_thr_3_g;
    unsigned char rg_fac_3_rb;
    unsigned char rg_fac_3_g;
    unsigned char ro_lim_3_rb;
    unsigned char ro_lim_3_g;
    unsigned char ro_lim_2_rb;
    unsigned char ro_lim_2_g;
    unsigned char ro_lim_1_rb;
    unsigned char ro_lim_1_g;
    unsigned char rnd_offs_3_rb;
    unsigned char rnd_offs_3_g;
    unsigned char rnd_offs_2_rb;
    unsigned char rnd_offs_2_g;
    unsigned char rnd_offs_1_rb;
    unsigned char rnd_offs_1_g;

} Adpcc_basic_params_select_t;
```

Adpcc_basic_params_t

【说明】
定义DPCC基本参数属性

【定义】

```
typedef struct Adpcc_basic_params_s
{
    Adpcc_basic_params_select_t arBasic[DPCC_MAX_ISO_LEVEL];
} Adpcc_basic_params_t;
```

【成员】

成员名称	描述
arBasic	DPCC基本参数属

Adpcc_bpt_params_t

【说明】
定义自动DPCC属性

【定义】

```
typedef struct Adpcc_bpt_params_s
{
    unsigned char      bpt_rb_3x3;
    unsigned char      bpt_g_3x3;
    unsigned char      bpt_incl_rb_center;
    unsigned char      bpt_incl_green_center;
    unsigned char      bpt_use_fix_set;
    unsigned char      bpt_use_set_3;
```

```
    unsigned char       bpt_use_set_2;
    unsigned char       bpt_use_set_1;
    unsigned char       bpt_cor_en;
    unsigned char       bpt_det_en;
    unsigned short int  bp_number;
    unsigned short int  bp_table_addr;
    unsigned short int  bpt_v_addr;
    unsigned short int  bpt_h_addr;
    unsigned int        bp_cnt;
} Adpcc_bpt_params_t;
```

dpcc_pdaf_point_t

【说明】

【定义】

```
typedef struct dpcc_pdaf_point_s
{
    unsigned char y;
    unsigned char x;
} dpcc_pdaf_point_t;
```

该模块还未实现

Adpcc_pdaf_params_t

【说明】
定义自动模式下PDAF模式属性

【定义】

```
typedef struct Adpcc_pdaf_params_s
{
    unsigned char       sw_pdaf_en;
    unsigned char       pdaf_point_en[DPCC_PDAF_POINT_NUM];
    unsigned short int  pdaf_offsety;
    unsigned short int  pdaf_offsetx;
    unsigned char       pdaf_wrapy;
    unsigned char       pdaf_wrapx;
    unsigned short int  pdaf_wrapy_num;
    unsigned short int  pdaf_wrapx_num;
    dpcc_pdaf_point_t   point[DPCC_PDAF_POINT_NUM];
    unsigned char       pdaf_forward_med;
} Adpcc_pdaf_params_t;
```

该模块还未实现

CalibDb_Dpcc_Fast_Mode_t

【说明】
定义自动模式下Fast mode属性

【定义】

```
typedef struct CalibDb_Dpcc_Fast_Mode_s
{
    int fast_mode_en;
    int ISO[CALIBDB_DPCC_MAX_ISO_LEVEL];
    int fast_mode_single_en;
    int fast_mode_single_level[CALIBDB_DPCC_MAX_ISO_LEVEL];
    int fast_mode_double_en;
    int fast_mode_double_level[CALIBDB_DPCC_MAX_ISO_LEVEL];
    int fast_mode_triple_en;
    int fast_mode_triple_level[CALIBDB_DPCC_MAX_ISO_LEVEL];
} CalibDb_Dpcc_Fast_Mode_t;
```

【成员】

成员名称	描述
Fast_mode_enable	Fast_mode开关功能，0：关闭，1：打开
ISO	环境ISO
fast_mode_single_en	单坏点去除开关，0：关闭，1：打开
fast_mode_single_level	单坏点去除力度，取值范围[0, 10]
fast_mode_double_en	双坏点去除开关，0：关闭，1：打开
fast_mode_double_level	双坏点去除力度，取值范围[0, 10]
fast_mode_triple_en	多坏点去除开关，0：关闭，1：打开
fast_mode_triple_level	多坏点去除力度，取值范围[0, 10]

CalibDb_Dpcc_Sensor_t

【说明】
定义自动模式下Fast mode属性

【定义】

```
typedef struct CalibDb_Dpcc_Sensor_s
{
    float en;
    float max_level;
    float iso[CALIBDB_DPCC_MAX_ISO_LEVEL];
    float level_single[CALIBDB_DPCC_MAX_ISO_LEVEL];
    float level_multiple[CALIBDB_DPCC_MAX_ISO_LEVEL];
} CalibDb_Dpcc_Sensor_t;
```

【成员】

成员名称	描述
en	sensor dpcc开关功能，0：关闭，1:打开
max_level	去除坏点最大力度
iso	环境ISO
level_single	去除单个坏点力度
level_multiple	去除多个坏点力度

Adpcc_bpt_params_select_t

【说明】
定义自动模式下选择的Fast mode属性

【定义】

```
typedef Adpcc_bpt_params_t Adpcc_bpt_params_select_t;
```

Adpcc_pdaf_params_select_t

【说明】
定义自动模式下选择的PDAF模式属性

【定义】

```
typedef Adpcc_pdaf_params_t Adpcc_pdaf_params_select_t
```

Adpcc_Auto_Attr_t

【说明】
定义自动DPCC属性

【定义】

```
typedef struct Adpcc_Auto_Attr_s
{
    Adpcc_basic_params_t        stBasicParams;
    Adpcc_bpt_params_t          stBptParams;
    Adpcc_pdaf_params_t         stPdafParams;
    CalibDb_Dpcc_Fast_Mode_t    stFastMode;
    CalibDb_Dpcc_Sensor_t       stSensorDpcc;
    Adpcc_basic_params_select_t stBasicSelect;
    Adpcc_bpt_params_select_t   stBptSelect;
    Adpcc_pdaf_params_select_t  stPdafSelect;

} Adpcc_Auto_Attr_t;
```

【成员】

成员名称	描述
stBasicParams	自动模式下基本参数
stBptParams	自动模式下坏点参数
stPdafParams	自动模式下PDAF模式参数
stFastMode	自动模式下快速模式参数
stSensorDpcc	自动模式下Sensor坏点功能参数
stBasicSelect	自动模式下选择的基本参数
stBptSelect	自动模式下选择的坏点参数
stPdafSelect	自动模式下选择的PDAF模式参数

Adpcc_fast_mode_attr_t

【说明】
定义手动模式下快速模式属性

【定义】

```
typedef struct Adpcc_fast_mode_attr_s
{
    bool fast_mode_en;
    bool fast_mode_single_en;
    int  fast_mode_single_level;
    bool fast_mode_double_en;
    int  fast_mode_double_level;
    bool fast_mode_triple_en;
    int  fast_mode_triple_level;
} Adpcc_fast_mode_attr_t;
```

【成员】

成员名称	描述
Fast_mode_en	Fast_mode开关功能
fast_mode_single_en	单坏点去除开关
fast_mode_single_level	单坏点去除力度，取值范围[0, 10]
fast_mode_double_en	双坏点去除开关
fast_mode_double_level	双坏点去除力度，取值范围[0, 10]
fast_mode_triple_en	多坏点去除开关
fast_mode_triple_level	多坏点去除力度，取值范围[0, 10]

Adpcc_sensor_dpcc_attr_t

【说明】
定义手动模式下Sensor坏点功能属性

【定义】

```
typedef struct Adpcc_sensor_dpcc_attr_s
{
    bool en;
    int  max_level;
    int  single_level;
    int  double_level;
} Adpcc_sensor_dpcc_attr_t;
```

【成员】

成员名称	描述
en	sensor dpcc开关功能
max_level	去除坏点最大力度
single_level	去除单个坏点力度
double_level	去除多个坏点力度

Adpcc_Manual_Attr_t

【说明】
定义手动DPCC属性

【定义】

```
typedef struct Adpcc_Manual_Attr_s
{
    unsigned char enable;
    Adpcc_onfly_cfg_t stOnfly;
    Adpcc_bpt_params_select_t   stBpt;
    Adpcc_pdaf_params_select_t  stPdaf;
    Adpcc_sensor_dpcc_attr_t    stSensorDpcc;
} Adpcc_Manual_Attr_t;
```

【成员】

成员名称	描述
enable	手动模式DPCC开关
stOnfly	手动模式下ISP DPCC模块动态坏点检测以及校正功能参数
stBptParams	手动模式下ISP DPCC模块静态坏点校正功能参数
stPdafParams	手动模式下ISP DPCC模块PDAF SPC功能参数
stSensorDpcc	手动模式下Sensor端DPCC 模块坏点校正功能参数

Adpcc_onfly_cfg_t

【说明】
定义手动DPCC基本参数

【定义】

```
typedef struct Adpcc_onfly_cfg_s {
   Adpcc_onfly_mode_t mode;
   Adpcc_fast_mode_attr_t fast_mode;
   Adpcc_basic_cfg_params_t expert_mode;
} Adpcc_onfly_cfg_t;
```

【成员】

成员名称	描述
mode	快速模式或者专家模式
fastmode	手动模式下快速模式参数设置
expert_mode	手动模式下专家模式参数设置

Adpcc_onfly_mode_t

【说明】
定义手动DPCC基本参数

【定义】

```
typedef enum Adpcc_onfly_mode_e {
    ADPCC_ONFLY_MODE_FAST        = 0,                  /**< dpcc manual fast
mode */
    ADPCC_ONFLY_MODE_EXPERT      = 1,                  /**< dpcc manual expert
mode */
    ADPCC_ONFLY_MODE_MAX                               /**< max */
} Adpcc_onfly_mode_t;
```

【成员】

成员名称	描述
ADPCC_ONFLY_MODE_FAST	快速模式
ADPCC_ONFLY_MODE_EXPERT	专家模式
ADPCC_ONFLY_MODE_MAX	无效参数设置

CalibDb_Dpcc_Pdaf_t

【说明】
定义工具PDAF SPC功能属性参数

【定义】

```
typedef struct CalibDb_Dpcc_Pdaf_s
{
    unsigned char      en;
    unsigned char      point_en[16];
    unsigned short int offsetx;
    unsigned short int offsety;
```

```
    unsigned char       wrapx;
    unsigned char       wrapy;
    unsigned short int wrapx_num;
    unsigned short int wrapy_num;
    unsigned char       point_x[16];
    unsigned char       point_y[16];
    unsigned char       forward_med;
} CalibDb_Dpcc_Pdaf_t;
```

CalibDb_Dpcc_set_RK_t

【说明】
定义RK算法属性

【定义】

```
typedef struct CalibDb_Dpcc_set_RK_s
{
    unsigned char rb_enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char rb_sw_mindis[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_sw_mindis[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char sw_dis_scale_min[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char sw_dis_scale_max[CALIBDB_DPCC_MAX_ISO_LEVEL];
} CalibDb_Dpcc_set_RK_t;
```

【成员】

成员名称	描述
rb_enable	红、蓝通道坏点检测开关
g_enable	绿通道坏点检测开关
rb_sw_mindis	红、蓝通道坏点阈值1
g_sw_mindis	绿通道坏点阈值1
sw_dis_scale_min	坏点阈值2
sw_dis_scale_max	坏点阈值3

CalibDb_Dpcc_set_LC_t

【说明】
定义LC算法属性

【定义】

```
typedef struct CalibDb_Dpcc_set_LC_s
{
    unsigned char rb_enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char rb_line_thr[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_line_thr[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char rb_line_mad_fac[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_line_mad_fac[CALIBDB_DPCC_MAX_ISO_LEVEL];
} CalibDb_Dpcc_set_LC_t;
```

【成员】

成员名称	描述
rb_enable	红、蓝通道坏点检测开关
g_enable	绿通道坏点检测开关
rb_line_thr	红、蓝通道坏点阈值1
g_line_thr	绿通道坏点阈值1
rb_line_mad_fac	红、蓝通道坏点阈值2
g_line_mad_fac	绿通道坏点阈值2

CalibDb_Dpcc_set_PG_t

【说明】
定义PG算法属性

【定义】

```
typedef struct CalibDb_Dpcc_set_PG_s
{
    unsigned char rb_enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char rb_pg_fac[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_pg_fac[CALIBDB_DPCC_MAX_ISO_LEVEL];
} CalibDb_Dpcc_set_PG_t;
```

【成员】

成员名称	描述
rb_enable	红、蓝通道坏点检测开关
g_enable	绿通道坏点检测开关
rb_pg_fac	红、蓝通道坏点阈值
g_pg_fac	绿通道坏点阈值

CalibDb_Dpcc_set_RND_t

【说明】
定义RND算法属性

【定义】

```
typedef struct CalibDb_Dpcc_set_RND_s
{
    unsigned char rb_enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char rb_rnd_thr[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_rnd_thr[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char rb_rnd_offs[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_rnd_offs[CALIBDB_DPCC_MAX_ISO_LEVEL];
} CalibDb_Dpcc_set_RND_t;
```

【成员】

成员名称	描述
rb_enable	红、蓝通道坏点检测开关
g_enable	绿通道坏点检测开关
rb_rnd_thr	红、蓝通道坏点阈值1
g_rnd_thr	绿通道坏点阈值1
rb_rnd_offs	红、蓝通道坏点阈值2
g_rnd_offs	绿通道坏点阈值2

CalibDb_Dpcc_set_RG_t

【说明】
定义RK算法属性

【定义】

```
typedef struct CalibDb_Dpcc_set_RG_s
{
    unsigned char rb_enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char rb_rg_fac[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_rg_fac[CALIBDB_DPCC_MAX_ISO_LEVEL];
} CalibDb_Dpcc_set_RG_t;
```

【成员】

成员名称	描述
rb_enable	红、蓝通道坏点检测开关
g_enable	绿通道坏点检测开关
rb_rg_fac	红、蓝通道坏点阈值
g_rg_fac	绿通道坏点阈值

CalibDb_Dpcc_set_RO_t

【说明】
定义RO算法属性

【定义】

```
typedef struct CalibDb_Dpcc_set_RO_s
{
    unsigned char rb_enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char rb_ro_lim[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char g_ro_lim[CALIBDB_DPCC_MAX_ISO_LEVEL];
} CalibDb_Dpcc_set_RO_t;
```

【成员】

成员名称	描述
rb_enable	红、蓝通道坏点检测开关
g_enable	绿通道坏点检测开关
rb_ro_lim	红、蓝通道坏点阈值
g_ro_lim	绿通道坏点阈值

CalibDb_Dpcc_set_t

【说明】
定义坏点判断条件属性

【定义】

```
typedef struct CalibDb_Dpcc_set_s
{
    CalibDb_Dpcc_set_RK_t  rk;
    CalibDb_Dpcc_set_LC_t  lc;
    CalibDb_Dpcc_set_PG_t  pg;
    CalibDb_Dpcc_set_RND_t rnd;
    CalibDb_Dpcc_set_RG_t  rg;
    CalibDb_Dpcc_set_RO_t  ro;
} CalibDb_Dpcc_set_t;
```

【成员】

成员名称	描述
rk	RK算法
lc	LC算法
pg	PG算法
rnd	RND算法
rg	RG算法
ro	RO算法

CalibDb_Dpcc_Expert_Mode_t

【说明】
定义工具专家模式属性

【定义】

```
typedef struct CalibDb_Dpcc_Expert_Mode_s
{
    float               iso[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char       stage1_Enable[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char       grayscale_mode;
    unsigned char       rk_out_sel[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char       dpcc_out_sel[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char       stage1_rb_3x3[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char       stage1_g_3x3[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char       stage1_inc_rb_center[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char       stage1_inc_g_center[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char       stage1_use_fix_set[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char       stage1_use_set3[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char       stage1_use_set2[CALIBDB_DPCC_MAX_ISO_LEVEL];
    unsigned char       stage1_use_set1[CALIBDB_DPCC_MAX_ISO_LEVEL];
    CalibDb_Dpcc_set_t set[3];
} CalibDb_Dpcc_Expert_Mode_t;
```

【成员】

成员名称	描述
iso	环境ISO
stage1_Enable	默认值1
grayscale_mode	黑白模式开关，0：关闭，1：打开
rk_out_sel	RK坏点判断模式，0：模式1，1：模式2，2：模式3
dpcc_out_sel	坏点矫正模式，0：中值，1：RK模式
stage1_rb_3x3	默认值0
stage1_g_3x3	默认值0
stage1_inc_rb_center	默认值1
stage1_inc_g_center	默认值1
stage1_use_fix_set	内置坏点判定条件开关，0：关闭， 1：打开
stage1_use_set3	set_cell中第三种坏点判断条件开关，0：关闭，1：打开
stage1_use_set2	set_cell中第二种坏点判断条件开关，0：关闭，1：打开
stage1_use_set1	set_cell中第一种坏点判断条件开关，0：关闭，1：打开
set	坏点判断条件

CalibDb_Dpcc_t

【说明】
定义工具DPCC属性

【定义】

```
typedef struct CalibDb_Dpcc_s
{
    int                         enable;
    char                        version[64];
    CalibDb_Dpcc_Fast_Mode_t    fast;
    CalibDb_Dpcc_Expert_Mode_t expert;
    CalibDb_Dpcc_Pdaf_t         pdaf;
    CalibDb_Dpcc_Sensor_t       sensor_dpcc;
} CalibDb_Dpcc_t;
```

【成员】

成员名称	描述
enable	开关功能
version	版本
fast	快速模式
expert	专家模式
pdaf	PADF SPC功能模式下坏点条件
sensor_dpcc	Sensor端DPCC模块坏点校正参数

rk_aiq_dpcc_attrib_t

【说明】
定义DPCC属性

【定义】

```
typedef struct rk_aiq_dpcc_attrib_s
{
    AdpccOPMode_t       eMode;
    Adpcc_Auto_Attr_t   stAuto;
    Adpcc_Manual_Attr_t stManual;
    CalibDb_Dpcc_t      stTool;
} rk_aiq_dpcc_attrib_t;
```

【成员】

成员名称	描述
eMode	api模式
stAuto	自动DPCC模式
stManual	手动DPCC模式
stTool	工具DPCC模式

LSC

功能描述

镜头阴影校正（Lens Shading Correction）是为了解决由于lens的光学特性，由于镜头对于光学折射不均匀导致的镜头周围出现阴影的情况。

模块级API参考

rk_aiq_user_api2_alsc_SetAttrib

【描述】
设置镜头阴影校正属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_alsc_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_lsc_attrib_t attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	镜头阴影校正属性	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_alsc.h
- 库文件：librkaiq.so

rk_aiq_user_api2_alsc_GetAttrib

【描述】
获取镜头阴影校正属性。

【语法】

```
XCamReturn
rk_aiq_user_api2_adebayer_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_lsc_attrib_t *attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	镜头阴影校正属性	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_alsc.h
- 库文件：librkaiq.so

数据类型

rk_aiq_lsc_attrib_t

【说明】
定义ISP镜头阴影校正属性。

【定义】

```
typedef struct rk_aiq_lsc_attrib_s {
    bool byPass;
    rk_aiq_lsc_op_mode_t mode;
    rk_aiq_lsc_mlsc_attrib_t stManual;
    rk_aiq_uapi_sync_t sync;
} rk_aiq_lsc_attrib_t;
```

【成员】

成员名称	描述
sync	接口同步/异步功能，参考rk_aiq_uapi_sync_t定义说明，参见 **“概述/API说明”** 章节
bypass	LSC模块使能 1：关闭 0： 使能
mode	模式设置
stManual	手动模式下的参数设置

rk_aiq_lsc_op_mode_t

【说明】
定义ISP镜头阴影校正工作模式。

【定义】

```
typedef enum rk_aiq_lsc_op_mode_s {
    RK_AIQ_LSC_MODE_INVALID                     = 0,        /**< initialization
value */
    RK_AIQ_LSC_MODE_MANUAL                      = 1,        /**< run manual lens
shading correction */
    RK_AIQ_LSC_MODE_AUTO                        = 2,        /**< run auto lens
shading correction */
    RK_AIQ_LSC_MODE_MAX
} rk_aiq_lsc_op_mode_t;
```

【成员】

成员名称	描述
RK_AIQ_LSC_MODE_INVALID	无效模式
RK_AIQ_LSC_MODE_MANUAL	手动模式
RK_AIQ_LSC_MODE_AUTO	自动模式

rk_aiq_lsc_table_t

【说明】
定义ISP镜头阴影校正表。

【定义】

```
typedef struct rk_aiq_lsc_table_s {
    unsigned short r_data_tbl[LSC_DATA_TBL_SIZE];
    unsigned short gr_data_tbl[LSC_DATA_TBL_SIZE];
    unsigned short gb_data_tbl[LSC_DATA_TBL_SIZE];
    unsigned short b_data_tbl[LSC_DATA_TBL_SIZE];
} rk_aiq_lsc_table_t;
```

【成员】

成员名称	描述
r_data_tbl	R分量校正表 元素个数为[17x17] 取值范围[1~7.99] 高3bit整数位低8bit小数位
gr_data_tbl	GR分量校正表 元素个数为[17x17] 取值范围[1~7.99] 高3bit整数位低8bit小数位
gb_data_tbl	GB分量校正表 元素个数为[17x17] 取值范围[1~7.99] 高3bit整数位低8bit小数位
b_data_tbl	B分量校正表 元素个数为[17x17] 取值范围[1~7.99] 高3bit整数位低8bit小数位

GIC

功能描述

GIC 模块用于矫正Gr与Gb两个通道的失衡，提高部分场景的图像质量。

模块级API参考

rk_aiq_user_api2_agic_v2_SetAttrib

【描述】
设定GIC软件属性。

【语法】

```
XCamReturn rk_aiq_user_api2_agic_v2_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
const rkaiq_gic_v2_api_attr_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ上下文指针	输入
attr	GIC软件属性结构体	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_agic.h
- 库文件：librkaiq.so

【说明】

无。

rk_aiq_user_api2_agic_v2_GetAttrib

【描述】
获取 Gamma软件属性。

【语法】

```
XCamReturn rk_aiq_user_api2_agic_v2_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rkaiq_gic_v2_api_attr_t* attr);
```

【参数】

参数名称	描述	输入/输出
sys_ctx	AIQ 上下文指针	输入
attr	GIC 软件属性结构体	输出

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api2_agic.h
- 库文件：librkaiq.so

【说明】

模块级API数据类型

rkaiq_gic_api_op_mode_t

【说明】
定义GIC工作模式

【定义】

```
typedef enum rkaiq_gic_api_op_mode_e {
    RKAIQ_GIC_API_OPMODE_OFF    = 0,
    RKAIQ_GIC_API_OPMODE_AUTO   = 1,
    RKAIQ_GIC_API_OPMODE_MANUAL = 2,
} rkaiq_gic_api_op_mode_t;
```

【成员】

成员名称	描述
RKAIQ_GIC_API_OPMODE_OFF	GIC 关闭模式
RKAIQ_GIC_API_OPMODE_AUTO	Api手动模式
RKAIQ_GIC_API_OPMODE_MANUAL	手动模式

rkaiq_gic_v2_param_selected_t

【说明】
定义RK3588下自动/手动的属性

【定义】

```
typedef struct rkaiq_gic_v2_param_selected_s {
    uint32_t iso;
    uint8_t bypass;
    uint8_t gr_ratio;
    uint16_t min_busy_thre;
    uint16_t min_grad_thr1;
    uint16_t min_grad_thr2;
    uint16_t k_grad1;
    uint16_t k_grad2;
    uint16_t gb_thre;
    uint16_t maxCorV;
    uint16_t maxCorVboth;
    uint16_t dark_thre;
    uint16_t dark_threHi;
    uint16_t k_grad1_dark;
    uint16_t k_grad2_dark;
    uint16_t min_grad_thr_dark1;
    uint16_t min_grad_thr_dark2;
    float NoiseScale;
    float NoiseBase;
    float noiseCurve_0;
    float noiseCurve_1;
    float globalStrength;
    uint16_t diff_clip;
} rkaiq_gic_v2_param_selected_t;
```

【成员】

成员名称	描述
iso	环境iso
bypass	预留，暂不使用
gr_ratio	预留，暂不使用
min_busy_thre	busy区域检测能力，取值范围[0, 1023]，默认值160
min_grad_thr1	非边缘区域的数量阈值1，GIC强度控制值，取值范围[0, 1023]，默认值32
min_grad_thr2	非边缘区域的数量阈值2，GIC强度控制值，取值范围[0, 1023]，默认值32
k_grad1	边缘（水平、垂直梯度）的响应程度阈值1，取值范围[0, 15]，默认值5
k_grad2	边缘（水平、垂直梯度）的响应程度阈值2，取值范围[0, 15]，默认值1
gb_thre	缩放的比例系数，取值范围[0, 15]，默认值7
maxCorV	限制边缘区域gb的最大补偿值，取值范围[0, 1023]，默认值40
maxCorVboth	限制平坦（非边缘） 区域gb最大补偿值，取值范围[0, 1023]，默认值8
dark_thre	定义暗部区域的阈值1，取值范围[0, 2047]，默认值120
dark_threHi	定义暗部区域的阈值2，取值范围[0, 2047]，默认值240
k_grad1_dark	图像暗部的边缘（水平、垂直梯度）响应程度阈值1，取值范围[0, 15]，默认值6
k_grad2_dark	图像暗部的边缘（水平、垂直梯度）响应程度阈值2，取值范围[0, 15]，默认值1
min_grad_thr_dark1	图像暗部的非边缘区域的数量阈值1，取值范围[0, 1023]，默认值64
min_grad_thr_dark2	图像暗部的非边缘区域的数量阈值2，取值范围[0, 1023]，默认值32
noiseCurve_0	噪声曲线参数1
noiseCurve_1	噪声曲线参数2
globalStrength	全局控制调整gb补偿值的强度，取值范围[0, 2]，默认值1
NoiseScale	根据噪声曲线获取当前点噪 声标准差， 利用 noise_std *noise_scale来确定最大gb补偿值
NoiseBase	惩罚图像边缘调整阈值，根据第一梯度和第二梯度计算结果 加上noise_offset，然后进行比较只要一个方向gradx>2*grady 就认为是边缘，不做调整
diff_clip	限制最大gb的最大补偿值

rkaiq_gic_v2_api_attr_t

【说明】
定义GIC属性

【定义】

```
typedef struct rkaiq_gic_v2_api_attr_s {
    rk_aiq_uapi_sync_t sync;
    uint8_t gic_en;
    rkaiq_gic_api_op_mode_t op_mode;
    uint32_t iso_cnt;
    rkaiq_gic_v2_param_selected_t auto_params[RKAIQ_GIC_MAX_ISO_CNT];
    rkaiq_gic_v2_param_selected_t manual_param;
} rkaiq_gic_v2_api_attr_t;
```

【成员】

成员名称	描述
sync	参考rk_aiq_uapi_sync_t定义，参见 **“概述/API说明”** 章节
gic_en	开关
op_mode	工作模式
iso_cnt	自动参数中的ISO个数
auto_params	自动参数，每个iso一组，软件自动根据iso插值计算
manual_param	手动参数，固定使用改组参数

#####

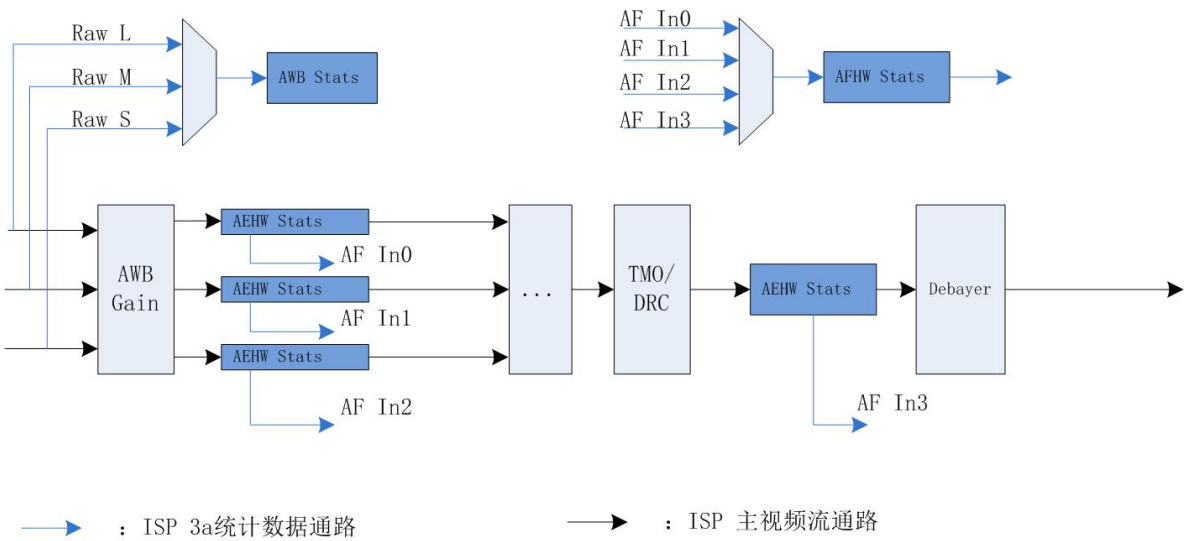
统计信息

概述

概述

ISP30支持对图像数据处理获取到AWB / AE / AF 3A控制算法需要的相关的统计信息，大致框图如下：

ISP30 3A统计框图



功能描述

AE统计信息

AE硬件统计信息主要包含以下几个部分：

基于raw域的256段加权直方图统计信息、分块R/G/B/Y 均值统计信息；

基于raw图的AE统计

- 该模块统计分为分块亮度统计和直方图统计。根据支持的分块大小和是否含有独立子窗口统计，统计模式又可分为big模式、lite模式。
- big模式：最大支持15X15非独立子窗口分块，最小支持1X1非独立子窗口分块，每个分块均可输出10bit R/B通道亮度均值和12bit G通道均值，默认采用15X15分块；同时支持设置4个独立子窗口，每个独立子窗口均可输出29bit R/B通道亮度总和和32bit G通道总和，亮度均值需要在软件中除以每个独立子窗口的像素数求得。该模式下的加权直方图统计，根据分块数和对应分配的权重，进行256段8bit亮度统计，每个亮度分段内像素数的有效bit数为28bit。
- lite模式：最大支持5X5非独立子窗口分块，最小支持1X1非独立子窗口分块，每个分块均可输出10bit R/B通道亮度均值和12bit G通道均值，默认采用5X5分块；不支持设置独立子窗口。该模式下的加权直方图统计，根据分块数和对应分配的权重，进行256段8bit亮度统计，每个亮度分段内像素数的有效bit数为28bit。
- 3588平台，默认非HDR曝光模式，配置BIG模式（15X15非独立子窗口）；HDR 2帧模式，各帧皆为BIG模式（15X15非独立子窗口）；HDR 3帧模式，短帧和长帧为BIG模式（15X15非独立子窗口），中帧为LITE模式（5X5非独立子窗口）

AWB统计信息

AWB硬件统计数据源：

- HDR多帧模式下，支持选择L / M / S 3个通道中的其中1个通道。
- 线性模式下，仅有1个通道数据。
- 在Raw-BLC-LSC之后的数据上进行统计

AWB硬件统计信息包含白点统计信息和分块统计信息

- 白点统计信息

  记录主窗口内7个光源2种大小的白区里的RGin,BGain累加值及个数

  记录子窗口内4个光源2种大小的白区里的RGin,BGain累加值及个数

  4个UV或XY域附加框里的RGin,BGain累加值及个数

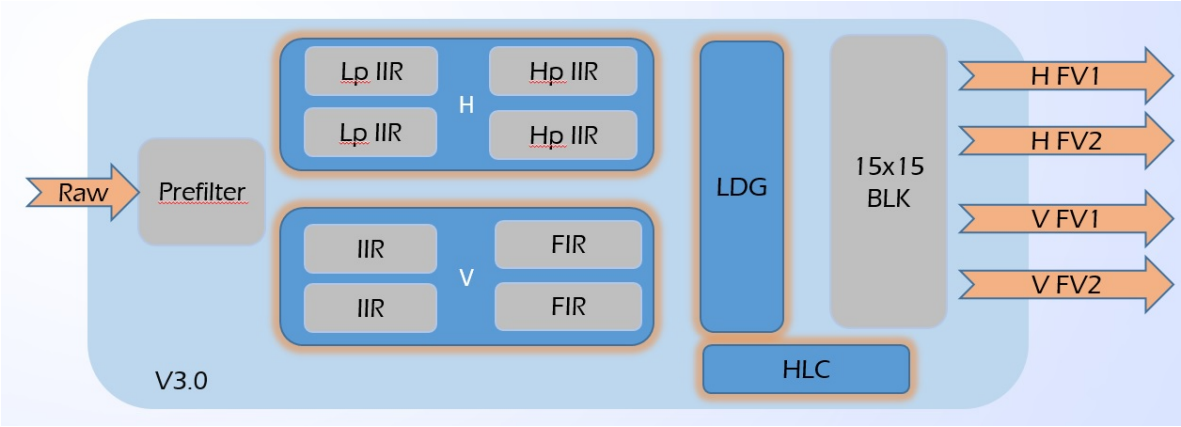
- 分块统计信息
  图像15x15分块，每个块所有点或白点的R,G,B均值

AF统计信息

AFHW Stats硬件统计：

- HDR多帧模式下，支持选择L / M / S 3个通道中的其中1个通道。同时支持3通道合成后的数据作为输入。
- 线性模式下，仅有1个通道数据。
- 支持水平方向2个可配置频段的FV输出，垂直方向上2个可配置频段的FV输出

ISP30 AF硬件统计模块框图



API参考

rk_aiq_uapi_sysctl_get3AStatsBlk

【描述】

同步获取3A统计信息。

【语法】

```
XCamReturn
rk_aiq_uapi_sysctl_get3AStatsBlk(const rk_aiq_sys_ctx_t* ctx, rk_aiq_isp_stats_t
**stats, int timeout_ms);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
stats	统计信息结构体指针	输出
timeout_ms	超时时间，-1意思是无限等待，直到有统计数据	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi_sysctl_release3AStatsRef

【描述】

释放获取的3A统计信息，与rk_aiq_uapi_sysctl_get3AStatsBlk配套使用。

【语法】

```
void
rk_aiq_uapi_sysctl_release3AStatsRef(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_isp_stats_t *stats);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
stats	统计信息结构体指针	输入

【返回值】

返回值	描述
0	成功
非0	失败，详见错误码表

【需求】

- 头文件：rk_aiq_user_api_sysctl.h
- 库文件：librkaiq.so

参考代码

sdk: external/camera_engine_rkaiq/rkisp_demo/demo/af_algo_demo

数据类型

rk_aiq_isp_stats_t

【说明】
AIQ 3A统计信息

【定义】

```
typedef struct {
    rk_aiq_isp_aec_stats_t aec_stats;
    rk_aiq_isp_awb_stats2_v3x_t awb_stats_v3x;
    rk_aiq_isp_af_stats_t  af_stats;
} rk_aiq_isp_stats_t;
```

【成员】

成员名称	描述
aec_stats	ae统计信息
rk_aiq_isp_awb_stats2_v3x_t	awb统计信息
af_stats	af统计信息

RKAiqAecStats_t

【说明】
定义AE数据信息，详细内容参见AE章节的功能描述。

【定义】

```
typedef struct RKAiqAecStats_s {
    RkAiqAecHwStatsRes_t ae_data;
    RKAiqAecExpInfo_t ae_exp;
} RKAiqAecStats_t;
```

【成员】

成员名称	描述
ae_data	AE模块硬件统计信息
ae_exp	AE模块sensor曝光信息

RKAiqAecExpInfo_t

【说明】
AE模块曝光参数信息

【定义】

```
typedef struct RKAiqAecExpInfo_s {
    RkAiqExpParamComb_t LinearExp;
    RkAiqExpParamComb_t HdrExp[3];
    RkAiqIrisParamComb_t Iris;
    uint16_t line_length_pixels;
    uint32_t frame_length_lines;
    float pixel_clock_freq_mhz;
    CISFeature_t CISFeature;
    RKAiqExpI2cParam_t exp_i2c_params;
} RKAiqAecExpInfo_t;
```

【成员】

成员名称	描述
LinearExp	非HDR模式的曝光参数信息，包含曝光实际值和RK格式的寄存器值
HdrExp	HDR模式的曝光参数信息，包含曝光实际值和RK格式的寄存器值，至多支持3帧曝光
exp_i2c_params	曝光参数的sensor寄存器值，第三方AE方案使用的曝光寄存器值参数
line_length_pixels	hts，其值由sensor的配置序列决定
frame_length_lines	vts，其值由sensor的配置序列决定
pixel_clock_freq_mhz	pclk，单位MHz，其值由sensor的配置序列决定

【注意事项】

- HdrExp表示HDR模式下的曝光参数信息，至多支持3TO1。HDR 2TO1: 下标0表示短帧曝光参数，下标1表示长帧曝光参数，下标2无效；HDR 3TO1：下标0表示短帧曝光参数，下标1表示中帧曝光参数，下标2表示长帧曝光参数。

RkAiqExpParamComb_t

【说明】
AE模块曝光参数信息详细内容

【成员】

```
typedef struct {
    RkAiqExpRealParam_t exp_real_params; //real value
    RkAiqExpSensorParam_t exp_sensor_params;//RK reg value
} RkAiqExpParamComb_t;
```

成员名称	描述
exp_real_params	曝光分量的实际物理值
exp_sensor_params	曝光分量的sensor寄存器值，遵循RK曝光设置方式

```
typedef struct RkAiqExpRealParam_s {
    float integration_time;
    float analog_gain;
    float digital_gain;
    float isp_dgain;
    int   iso;
    int   dcg_mode;
} RkAiqExpRealParam_t;
```

成员名称	描述
integration_time	曝光积分时间，单位为秒(s)
analog_gain	sensor的模拟增益/Total增益，单位为倍数
digital_gain	sensor的数字增益，单位为倍数，当sensor的数字增益起弥补模拟增益精度作用时，配置为1x，整体增益值写入analog_gain中
isp_dgain	isp数字增益，单位为倍数，目前暂无效，默认值为1x
iso	总增益值，iso表示系统增益，以常数50乘以倍数为单位，iso = total增益*50
dcg_mode	dual conversion gain模式

```
typedef struct RkAiqExpSensorParam_s {
    unsigned short fine_integration_time;
    unsigned short coarse_integration_time;
    unsigned short analog_gain_code_global;
    unsigned short digital_gain_global;
    unsigned short isp_digital_gain;
} RkAiqExpSensorParam_t;
```

成员名称	描述
fine_integration_time	sensor曝光积分时间的小数行寄存器值，仅当sensor支持小数行时，需要填写
coarse_integration_time	sensor曝光积分时间对应的寄存器值，单位为行数
analog_gain_code_global	sensor模拟增益对应的寄存器值
digital_gain_global	sensor数字增益对应的寄存器值
isp_digital_gain	isp数字增益寄存器值，暂时无效

【注意事项】

- 不同sensor的数字增益作用不同，有的是用于增大感光度范围，有的是用于补足模拟增益的精度。因此目前先不将数字增益单独列出，其大小和对应寄存器值全部并入模拟增益中。
- dual conversion gain模式共有三种状态，值为-1代表sensor不支持dcg，值为0代表LCG，值为1代表HCG

RKAiqExpI2cParam_t

【说明】
AE模块I2c曝光参数(一般为第三方AE使用)

【定义】

```
#define MAX_I2CDATA_LEN 64
typedef struct RKAiqExpI2cParam_s {
    bool           bValid;
    unsigned int   nNumRegs;
    unsigned int   RegAddr[MAX_I2CDATA_LEN];
    unsigned int   AddrByteNum[MAX_I2CDATA_LEN];
    unsigned int   RegValue[MAX_I2CDATA_LEN];
    unsigned int   ValueByteNum[MAX_I2CDATA_LEN];
    unsigned int   DelayFrames[MAX_I2CDATA_LEN];
} RKAiqExpI2cParam_t;
```

【成员】

成员名称	描述
bValid	I2c参数生效使能位 true：使用RKAiqExpI2cParam_t设置寄存器值（一般为第三方AE使用）； false：使用RK格式的寄存器参数进行寄存器值设置
nNumRegs	设置的寄存器值数量
RegAddr	寄存器地址数组，元素最大个数为64
AddrByteNum	寄存器地址的比特长度，元素最大个数为64
RegValue	寄存器值数组，元素最大个数为64
ValueByteNum	寄存器值的比特长度，元素最大个数为64
DelayFrames	寄存器值延迟生效帧数数组，元素最大个数为64

【注意事项】

- 该寄存器参数仅在第三方AE应用，且bValid为true时才有效，否则默认使用RkAiqExpParamComb_t中的RK格式寄存器参数
- 可支持设置的寄存器值最大个数为64

RkAiqIrisParamComb_t

【说明】
AE模块光圈参数

【定义】

```
typedef struct {
    RkAiqPIrisParam_t   PIris;
    RkAiqDCIrisParam_t  DCIris;
} RkAiqIrisParamComb_t;
typedef struct {
    int            step;
    int            gain;
    bool           update;
} RkAiqPIrisParam_t;
typedef struct {
    int        pwmDuty; //percent value,range = 0-100
    bool       update;
} RkAiqDCIrisParam_t;
```

【成员】

成员名称	子成员	描述
PIris	step gain update	P光圈参数： P光圈步进寄存器值 P光圈步进等效增益值 P光圈步进更新标志
DCIris	pwmDuty update	DC光圈参数： PWM占空比值，取值范围：1~100 DC光圈参数更新标志

RkAiqAecHwStatsRes_t

【说明】
AE模块硬件统计信息

【定义】

```
typedef struct RkAiqAecHwStatsRes_s {
    Aec_Stat_Res_t chn[3];
    Aec_Stat_Res_t extra;
    struct yuvae_stat yuvae;
    struct sihist_stat sihist;
} RkAiqAecHwStatsRes_t;
```

【成员】

成员名称	描述
chn[3]	AE模块基于raw图（BLC和AWB之后）的统计信息，兼容HDR与非HDR模式，至多支持HDR 3TO1 S/M/L的统计信息。
extra	AE模块基于raw图（Debayer之前）的统计信息，HDR模式下，其表示合成帧的统计信息
yuvae	AE模块基于gamma前RGB图的分块信息【3588平台不支持】
sihist	AE模块基于gamma前RGB图的直方图信息【3588平台不支持】

【注意事项】

- Aec_Stat_Res_t chn[3]： 代表HDR Merge模块前3个Raw数据通路的统计信息。非HDR模式，对应下标为0，其他下标均无效；HDR 2TO1模式，对应下标为0时表示短帧数据通路统计信息、下标1表示长帧数据通路统计信息，下标2无效；HDR 3TO1模式，对应下标为0时表示短帧数据通路统计信息、下标1表示中帧数据通路统计信息、下标2表示长帧数据通路统计信息。基于raw图的统计模块之前有BLC AWB模块，因此基于raw图的统计信息受BLC、AWB的增益值影响。
- Aec_Stat_Res_t extra：HDR模式下，extra表示HDR合成后的raw图统计信息。该统计模块之前有BLC、 AWB、 HDRMERGE、 TMO等模块，因此该模块的统计信息受BLC、 AWB、 HDRMERGE、TMO等模块的增益影响。**此外AF模块开启时，该部分统计信息无效。**
- 针对3588平台不支持基于gamma前RGB图的分块信息和直方图信息

Aec_Stat_Res_t

【说明】
AE模块基于raw图的统计信息

【定义】

```
typedef struct Aec_Stat_Res_s {
    //rawae
    struct rawaebig_stat rawae_big;
    struct rawaelite_stat rawae_lite;
    //rawhist
    struct rawhist_stat rawhist_big;
    struct rawhist_stat rawhist_lite;
} Aec_Stat_Res_t;
```

【成员】

成员名称	描述
rawae_big	基于raw图的big模式分块统计信息
rawae_lite	基于raw图的lite模式分块统计信息
rawhist_big	基于raw图的big模式直方图统计信息
rawhist_lite	基于raw图的lite模式直方图统计信息

【注意事项】

- 有关基于raw图统计的big、lite模式区别详见功能描述模块。需注意：big与lite模式的主要区别在于分块统计均值亮度的块数及是否支持独立子窗口均值亮度统计。基于raw图的big、lite模式直方图统计具有相同的数据结构。
- 3588平台，默认非HDR曝光模式，配置BIG模式（15X15非独立子窗口）；HDR 2帧模式，各帧皆为BIG模式（15X15非独立子窗口）；HDR 3帧模式，短帧和长帧为BIG模式（15X15非独立子窗口），中帧为LITE模式（5X5非独立子窗口）

rawaebig_stat

【说明】
基于raw图的big模式统计信息，包含15X15个非独立子窗口的R/G/B均值亮度、4个独立子窗口的R/G/B亮度总和

【定义】

```
struct rawaebig_stat {
    unsigned short channelr_xy[RAWAEBIG_WIN_NUM];
    unsigned short channelg_xy[RAWAEBIG_WIN_NUM];
    unsigned short channelb_xy[RAWAEBIG_WIN_NUM];
    unsigned int   channely_xy[RAWAEBIG_WIN_NUM]; //not HW!
    unsigned long int wndx_sumr[RAWAEBIG_SUBWIN_NUM];
    unsigned long int wndx_sumg[RAWAEBIG_SUBWIN_NUM];
    unsigned long int wndx_sumb[RAWAEBIG_SUBWIN_NUM];
    unsigned short wndx_channelr[RAWAEBIG_SUBWIN_NUM]; //not HW!
    unsigned short wndx_channelg[RAWAEBIG_SUBWIN_NUM]; //not HW!
    unsigned short wndx_channelb[RAWAEBIG_SUBWIN_NUM]; //not HW!
    unsigned char wndx_channely[RAWAEBIG_SUBWIN_NUM]; //not HW!
};
#define RAWAEBIG_WIN_NUM    225
```

```
#define RAWAEBIG_SUBWIN_NUM  4
```

【成员】

成员名称	描述
channelr_xy	big模式非独立子窗口分块的r通道均值亮度信息。有效比特数：10bit。
channelg_xy	big模式非独立子窗口分块的g通道均值亮度信息。有效比特数：12bit。
channelb_xy	big模式非独立子窗口分块的b通道均值亮度信息。有效比特数：10bit。
wndx_sumr	big模式独立子窗口的r通道亮度和信息。有效比特数：29bit。
wndx_sumg	big模式独立子窗口的g通道亮度和信息。有效比特数：32bit。
wndx_sumb	big模式独立子窗口的b通道亮度和信息。有效比特数：29bit。

【注意事项】

- 基于raw图的big模式统计信息，仅包含R/G/B 3通道的统计信息，如需Y通道统计信息，可在软件中添加代码根据R/G/B统计值计算。
- 基于raw图的big模式统计信息，包含15X15个非独立子窗口，非独立子窗口有最小尺寸限制，要求最小尺寸为16X4。
- 基于raw图的big模式统计信息，包含4个独立子窗口，其位置可重叠也可不重叠，有最大尺寸要求，不得超过2k x 1k。
- 结构体中的channely_xy、wndx_channelr、 wndx_channelg、wndx_channelb、wndx_channely参数皆为软件计算参数，需要软件添加代码，根据硬件统计值计算求得。

rawaelite_stat

【说明】
基于raw图的lite模式统计信息，包含5X5个非独立子窗口分块R/G/B均值亮度

【定义】

```
struct rawaelite_stat {
    unsigned short channelr_xy[RAWAELITE_WIN_NUM];
    unsigned short channelg_xy[RAWAELITE_WIN_NUM];
    unsigned short channelb_xy[RAWAELITE_WIN_NUM];
    unsigned int   channely_xy[RAWAELITE_WIN_NUM]; //not HW!
};
#define RAWAELITE_WIN_NUM  25
```

【成员】

成员名称	描述
channelr_xy	lite模式非独立子窗口分块的r通道均值亮度信息。有效比特数：10bit。
channelg_xy	lite模式非独立子窗口分块的g通道均值亮度信息。有效比特数：12bit。
channelb_xy	lite模式非独立子窗口分块的b通道均值亮度信息。有效比特数：10bit。

【注意事项】

- 基于raw图的lite模式统计信息，仅包含5X5的非独立子窗口R/G/B 3通道的统计信息，如需Y通道统计信息，可在软件中添加代码根据R/G/B统计值计算。非独立子窗口有最小尺寸限制，要求最小尺寸为16X4。
- 结构体中的channely_xy为软件计算参数，需要添加代码，根据硬件统计值计算求得。

rawhist_stat

【说明】
基于raw图的直方图统计信息

【定义】

```
struct rawhist_stat {
    unsigned int bins[RAWHIST_BIN_N_MAX];
};
#define RAWHIST_BIN_N_MAX  256
```

【成员】

成员名称	描述
bins	直方图的分段，共256段，每个分段内有效bit数：28bit

rk_aiq_isp_awb_stats2_v3x_t

【说明】
定义白平衡硬件统计信息，主要包含白点统计信息和分块统计信息

- 白点统计信息

  记录主窗口内7个光源2种大小的白区里的RGin,BGain累加值及个数；

  记录子窗口内4个光源2种大小的白区里的RGin,BGain累加值及个数；

  4个UV或XY域附加框里的RGin,BGain累加值及个数；

  白点亮度直方图；

- 分块统计信息
  图像15x15分块(如AWB分块示意图所示)，每个块所有点或白点的R,G,B均值

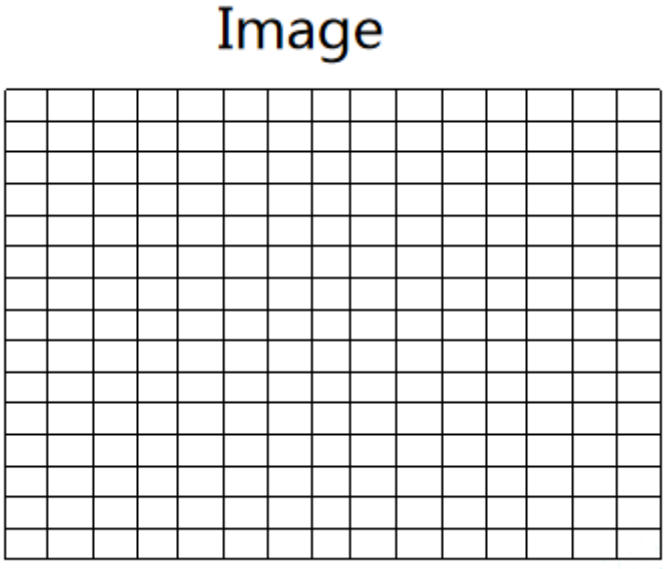


图 AWB分块示意图

【定义】

```
typedef struct rk_aiq_isp_awb_stats2_v3x_s {
    //method1
    rk_aiq_awb_stat_wp_res_light_v201_t light[RK_AIQ_AWB_MAX_WHITEREGIONS_NUM];
    int WpNo2[RK_AIQ_AWB_MAX_WHITEREGIONS_NUM];
    //method2
    rk_aiq_awb_stat_blk_res_v201_t   blockResult[RK_AIQ_AWB_GRID_NUM_TOTAL];
    //window in pixel domain
    rk_aiq_awb_stat_wp_res_light_v201_t
multiwindowLightResult[RK_AIQ_AWB_MAX_WHITEREGIONS_NUM];
    //window in xy or uv domain
    rk_aiq_awb_stat_wp_res_v201_t
excWpRangeResult[RK_AIQ_AWB_STAT_WP_RANGE_NUM_V201];
    //wpno histogram
    unsigned int WpNoHist[RK_AIQ_AWB_WP_HIST_BIN_NUM];
} rk_aiq_isp_awb_stats2_v3x_t;
```

【成员】

成员名称	描述
light	主窗口下不同光源下的白点统计结果，最多RK_AIQ_AWB_MAX_WHITEREGIONS_NUM个光源；
WpNo2	主窗口下不同光源下的xy域和uv域交集的白点个数，没有小数位。
blockResult	每个块的RGB累加 图像采用均匀分块方式，共15x15（RK_AIQ_AWB_GRID_NUM_TOTAL）块，如AWB分块示意图所示
multiwindowLightResult	几个子窗口内不同光源下的白点统计结果（只记录前4个光源，可以用时分复用方法记录所有光源），最多4个子窗口
excWpRangeResult	落在excludeWpRange区域里的点的统计结果（只会记录excludeWpRange前四个区域），最多4个区域
WpNoHist	白点直方图每个bin的白点个数，没有小数位。 统计的是XY大框还是XY中框的白点由寄存器xyRangeTypeForWpHist确定。

【注意事项】

如果用户希望获取主窗口全局的白点统计结果，根据所有光源下的白点统计结果可以简单换算得到。

rk_aiq_awb_stat_wp_res_light_v201_t

【说明】

定义某个光源下的白点统计结果

【定义】

```
typedef struct rk_aiq_awb_stat_wp_res_light_v201_s {
    rk_aiq_awb_stat_wp_res_v201_t xYType[RK_AIQ_AWB_XY_TYPE_MAX_V201];
} rk_aiq_awb_stat_wp_res_light_v201_t;
```

【成员】

成员名称	描述
xYType	某个光源下不同大小的XY框的白点统计结果，最多RK_AIQ_AWB_XY_TYPE_MAX_V201个框，即2个框

rk_aiq_awb_stat_wp_res_v201_t

【说明】
定义某个光源某个大小的XY框下的白点统计结果，后非白点区域里的非白点统计结果

【定义】

```
typedef struct rk_aiq_awb_stat_wp_res_v201_s {
    long long WpNo;
    long long RgainValue;
    long long BgainValue;
} rk_aiq_awb_stat_wp_res_v201_t;
```

【成员】

成员名称	描述
WpNo	白点数量，22bit整数位，10ibit小数位。其中 WpNo = ISP硬件统计白点数 / 2^10.
RgainValue	白点R通道的累加和，22bit整数位，10ibit小数位
BgainValue	白点G通道的累加和，22bit整数位，10ibit小数位

rk_aiq_awb_stat_blk_res_v201_t

【说明】
定义每个块的统计结果。

若blkMeasureMode = =RK_AIQ_AWB_BLK_STAT_MODE_ALL_V201, 记录的结果为块内所有点的累加和;

若blkMeasureMode = =RK_AIQ_AWB_BLK_STAT_MODE_REALWP_V201, 记录的结果为块内所有白点的累加和;

其中blkMeasureMode 为控制块统计内容的寄存器。

【定义】

```
typedef struct rk_aiq_awb_stat_blk_res_v201_s {
    long long WpNo;
    long long Rvalue;
    long long Gvalue;
    long long Bvalue;
} rk_aiq_awb_stat_blk_res_v201_t;
```

【成员】

成员名称	描述
WpNo	块内点的个数
Rvalue	块内所有点R通道的累加和
Gvalue	块内所有点RG通道的累加和
Bvalue	块内所有点RB通道的累加和

rk_aiq_af_algo_stat_v30_t

【说明】
定义AF统计信息

【定义】

```
typedef struct {
    unsigned int wndb_luma;
    unsigned int wndb_sharpness;
    unsigned int winb_highlit_cnt;
    unsigned int wnda_luma[RKAIQ_RAWAF_SUMDATA_NUM];
    unsigned int wnda_fv_v1[RKAIQ_RAWAF_SUMDATA_NUM];
    unsigned int wnda_fv_v2[RKAIQ_RAWAF_SUMDATA_NUM];
    unsigned int wnda_fv_h1[RKAIQ_RAWAF_SUMDATA_NUM];
    unsigned int wnda_fv_h2[RKAIQ_RAWAF_SUMDATA_NUM];
    unsigned int wina_highlit_cnt[RKAIQ_RAWAF_SUMDATA_NUM];
    unsigned int int_state;

    struct timeval focus_starttim;
    struct timeval focus_endtim;
    struct timeval zoom_starttim;
    struct timeval zoom_endtim;
    int64_t sof_tim;
    int focusCode;
    int zoomCode;
    bool focusCorrection;
    bool zoomCorrection;
    float angleZ;
} rk_aiq_af_algo_stat_v30_t;
```

【成员】

成员名称	描述
wndb_luma	独立窗口的亮度值
wndb_sharpness	独立窗口的清晰度值
winb_highlit_cnt	独立窗口的高亮计数值
wnda_luma	主窗口下15*15子窗口的亮度值
wnda_fv_v1	主窗口下15*15子窗口的V1清晰度值
wnda_fv_v2	主窗口下15*15子窗口的V2清晰度值
wnda_fv_h1	主窗口下15*15子窗口的H1清晰度值
wnda_fv_h2	主窗口下15*15子窗口的H2清晰度值
wina_highlit_cnt	主窗口下15*15子窗口的高亮计数值
int_state	RK AF算法使用，可以忽略
focus_starttim	RK AF算法使用，可以忽略
focus_endtim	RK AF算法使用，可以忽略
zoom_starttim	RK AF算法使用，可以忽略
zoom_endtim	RK AF算法使用，可以忽略
sof_tim	本次数据帧的帧开始时间，单位ns
focusCode	RK AF算法使用，可以忽略
zoomCode	RK AF算法使用，可以忽略
focusCorrection	RK AF算法使用，可以忽略
zoomCorrection	RK AF算法使用，可以忽略
angleZ	RK AF算法使用，可以忽略

Debug & FAQ

如何获取版本号

aiq提供了版本发布日期、aiq版本、iq解析器版本及isp各个算法模块的版本信息；

获取简略版本信息

```
strings librkaiq.so | grep -w AIQ
AIQ: v0.1.6
```

获取完整版本信息

1. SDK中aiq库默认编译为Release版本，需要改成Debug版本，重新编译aiq库后更新到设备。

`SDK/external/camera_engine_rkaiq/CMakeLists.txt:`

```
8
9   if(NOT CMAKE_BUILD_TYPE STREQUAL "Release")
10      add_definitions(-DBUILD_TYPE_DEBUG)
11  endif()
```

改成：

```
8
9   #if(NOT CMAKE_BUILD_TYPE STREQUAL "Release")
10      add_definitions(-DBUILD_TYPE_DEBUG)
11  #endif()
```

2. 默认打印级别下，加载运行的aiq库不会打印，可以设置xcore模块的log级别，以打印aiq版本信息：

```
export persist_camera_engine_log=0x1000000ff2
```

3. aiq启动时打印版本信息如下所示：

```
************************* VERSION INFOS *************************
version release date: 2020-06-05
                 AIQ: v0.1.6
           IQ PARSER: v1.0.0
RK INTEGRATED ALGO MODULES:
                 AWB: v0.0.9
                 AEC: v0.1.1
                  AF: v0.0.9
                AHDR: v0.0.9
                 ANR: v0.0.9
              ASHARP: v0.0.9
             ADEHAZE: v0.0.9
              AGAMMA: v0.0.9
              A3DLUT: v0.0.9
                ABLC: v0.0.9
                ACCM: v0.0.9
                ACGC: v0.0.9
                 ACP: v0.0.9
            ADEBAYER: v0.0.1
               ADPCC: v0.0.9
                AGIC: v0.0.9
                 AIE: v0.0.1
               ALDCH: v0.0.9
                ALSC: v0.0.9
                AORB: v0.0.9
                AR2Y: v0.0.9
                 ASD: v0.0.9
                AWDR: v0.0.9
************************ VERSION INFOS END ************************
```

版本号匹配规则说明

AIQ与IQ Tool、ISP Driver的版本匹配规则如下：

v A . B. C

其中B为16进制表示，bit[0:3] 标识 AIQ与IQ Tool的匹配版本，bit[4:7]标识AIQ与ISP driver的匹配版本，例如：

ISP driver：v 1. 0x3.0 与 AIQ：v1.0x30.0匹配，与AIQ：v1.0x40.0不匹配。

IQ tool：v 1. 0x3.0 与 AIQ：v1.0x33.0匹配，与AIQ：v1.0x30.0不匹配， 其中AIQ 版本号C不为0，有可能出现版本不匹配的情况，针对IQ Tool匹配建议优先采用C版本号为0的AIQ版本。

AIQ Log

概述

aiq采用64bits表示所有模块的log级别，表示各个模块的位图及说明如下：

```
bit: [63-39]   38      37       36    35   34     33    32    31
mean:  [U]   [CAMHW][ANALYZER][XCORE][ASD][AFEC][ACGC][AORB][ASHARP]

bit:  30   29   28     27     26       25      24   23       22
mean:[AIE][ACP][AR2Y][ALDCH][A3DLUT][ADEHAZE][AWDR][AGAMMA][ACCM]

bit:      21     20    19   18    17    16     15   14  13   12
mean:[ADEBAYER][AGIC][ALSC][ANR][AHDR][ADPCC][ABLC][AF][AWB][AEC]

bit:    11-4        3-0
mean:[sub modules][level]

    [U] means unused now.
    [level] : use 4 bits to define log levels.
    each module log has following ascending levels:
    0: error
    1: warning
    2: info
    3: debug
    4: verbose
    5: low1
    6-7: unused, now the same as debug
    [sub modules] : use bits 4-11 to define the sub modules of each module,
thespecific meaning of each bit is decided by the module itself. These bitsis
designed to implement the sub module's log switch.
    [modules] : AEC, AWB, AF ...

    set debug level example:
    eg. set module af log level to debug, and enable all sub modules of af:
    Android:
    setprop persist.vendor.rkisp.log 0x4ff4
    Linux:
    export persist_camera_engine_log=0x4ff4

    And if only want enable the sub module 1 log of af:
    Android:
    setprop persist.vendor.rkisp.log 0x4014
    Linux:
    export persist_camera_engine_log=0x4014
```

如上说明，linux环境下通过设置环境变量persist_camera_engine_log来控制各个模块的开关级别。

模块	Debug log	Verbose log
AE	export persist_camera_engine_log=0x1ff3	export persist_camera_engine_log=0x1ff4
AWB	export persist_camera_engine_log=0x2ff3	export persist_camera_engine_log=0x2ff4
AF	export persist_camera_engine_log=0x4ff3	export persist_camera_engine_log=0x4ff4
HDR	export persist_camera_engine_log=0x20ff3	export persist_camera_engine_log=0x20ff3
NR	export persist_camera_engine_log=0x40ff3	export persist_camera_engine_log=0x40ff4
Dehaze	export persist_camera_engine_log=0x2000ff3	export persist_camera_engine_log=0x2000ff3
Sharp	export persist_camera_engine_log=0x80000ff3	export persist_camera_engine_log=0x80000ff4
CAMHW	export persist_camera_engine_log=0x4000000ff3	export persist_camera_engine_log=0x4000000ff4

查看当前log级别可通过如下命令：

```
[root@RV1126_RV1109:/]# echo $persist_camera_engine_log
0x4014
```

AE

配置指南

Log level	描述以及使能建议
Error	默认使能 严重错误
Warning	默认未使能 警告错误，算法内部可能针对该错误进行了异常处理，算法仍可继续运行
Info	默认未使能 **基本调试信息:** 逐帧输出信息：无 事件信息：算法中运行状态信息及帧率相关信息，一般仅在初始化、切换分辨率、修改配置参数时打印 **建议：** 用于获知算法运行状态及帧率值
Debug	默认未使能 **基本调试信息(相对Info等级增加)**: 逐帧输出信息：帧号、 帧匹配曝光量、帧匹配AE统计信息(图像亮度)及曝光结果值 事件信息: **建议：** 通过连续的图像亮度、曝光量等信息来调试图像亮度闪烁等问题
Verbose	默认未使能 **基本调试信息(相对Debug等级增加):** 逐帧输出信息：算法相关运算结果信息 事件信息： **建议：** 一般不开启，当debug等级无法判断问题时再开启

Sub modules bit	描述以及使能建议
bit[4]	与AE参数配置相关的log等级 **建议**: 读取IQ或API设置参数时，可根据该等级log查看配置参数是否正确。
bit[5]	AE信息处理模块日志开关，该模块根据AE统计计算环境以及图像相关指标。 **建议**: 读取和统计值相关的亮度信息
bit[6]	AE曝光量计算模块日志开关。 **建议**: 读取和曝光结果相关的信息
bit[7]	与Airis光圈算法相关的log等级， **建议**: 可变光圈相关问题。
bit[8:11]	无效

log解读

由于篇幅限制，此处仅对debug等级（0x1ff3，对应AE所有子模块的debug级信息)的log进行内容解读。

- 线性模式AE LOG

```
rk_aiq_algo_ae_itf.cpp:262: Cur-Exp: FrmId=270,gain=0x36a,time=0x576,envChange=0,dcg=-1,pirs=0
rk_aiq_algo_ae_itf.cpp:266: Last-Res:FrmId=269,gain=0x356,time=0x576,pirs=0

rk_aiq_ae_algo.cpp:5861: ================================ Linear-AE (enter)==============================
rk_aiq_ae_algo.cpp:5881: >>> Framenum=270 Cur gain=6.826667,time=0.029987,pirisGain=0,RawMeanluma=29.564444,YuvMeanluma=34.875557,IsConverged=0
rk_aiq_ae_algo.cpp:2961: AecClmExecute: NewExposure(0.172051) SplitGain(5.735024) SplitIntegrationTime(0.030000) SplitPirisGain(0)
rk_aiq_ae_algo.cpp:5952: calc result:SetPoint=22.000000,gain=5.720671,time=0.029987,piris=0,reggain=845,regtime=1398
rk_aiq_ae_algo.cpp:6133: ================================ (exit)================================
```

图3-1 线性模式AE LOG

如图3-1所示为线性模式的AE LOG示例。

Line1：

> Cur-Exp: FrmId=270,gain=0x36a,time=0x576,envChange=0,dcg=-1,pirs=0

当前帧的曝光参数信息。

成员名称	描述
FrmId	当前帧的帧号
gain	当前帧对应的sensor曝光增益寄存器值
time	当前帧对应的sensor曝光时间寄存器值
envChange	当前帧是否发生环境亮度突变。0：环境亮度无突变；1：环境亮度突变
dcg	当前帧对应的dcg模式。-1：sensor不支持dcg模式 或 sensor内部进行dcg模式的切换；0：LCG模式；1：HCG模式
pirs	当前帧对应的p-iris光圈步进电机位置。若Airis功能关闭，该参数无效，无意义。

Line2：

> Last-Res:FrmId=269,gain=0x356,time=0x576,pirs=0

上次运行AE设置的新曝光参数，部分参数与（1）中含义一致，此处不再赘述。通过比较Line1与Line2的曝光参数LOG信息，可知当前曝光是否与新曝光一致，即新曝光是否已经生效。

Line3：

> ================================ Linear-AE (enter)==============================

进入AE控制算法模块，Linear-AE表示当前为线性曝光模式。

Line4：

> Framenum=270
>
> Cur gain=6.826667,time=0.029987,pirisGain=0,RawMeanluma=29.564444,YuvMeanluma=34.875557,IsConverged=0

成员名称	描述
Framenum	当前帧的帧号
gain	当前帧对应的sensor曝光增益值
time	当前帧对应的sensor曝光时间值
pirisGain	当前帧对应的p-iris光圈等效增益值。若Airis功能关闭，该参数无效，无意义。
RawMeanluma	当前帧对应的debayer前raw图亮度，已扣除黑电平，并乘上awb gain。
YuvMeanluma	当前帧对应的gamma前RGB图亮度，用于判断debayer后的模块对亮度的影响。
IsConverged	当前帧曝光是否已经收敛。0：曝光未收敛；1：曝光已收敛

Line 5：

> AecClmExecute: NewExposure(0.180993) SplitGain(6.033096) SplitIntegrationTime(0.030000) SplitPirisGain(0)

成员名称	描述
NewExposure	AE控制算法得出的新曝光量值
SplitGain	新sensor曝光增益
SplitIntegrationTime	新sensor曝光时间
SplitPirisGain	新p-iris光圈等效增益值。若Airis功能关闭，该参数无效，无意义。

Line6:

> calc result:SetPoint=22.000000,gain=6.023529,time=0.029987,piris=0,reggain=854,regtime=1398

最终设置的新曝光

成员名称	描述
SetPoint	目标亮度值
gain	最终的新曝光增益值
time	最终的新曝光时间值
piris	最终的新p-iris光圈等效增益值。若Airis功能关闭，该参数无效，无意义。
reggain	最终的新曝光增益值对应的寄存器值
regtime	最终的新曝光时间值对应的寄存器值

综上，可得知当前帧的画面亮度RawMeanLuma，以及对应的目标亮度setpoint。通过比较画面亮度和目标亮度，计算新曝光。

- Hdr模式AE LOG:

```
rk_aiq_algo_ae_itf.cpp:246: Cur-Exp: FrmId=22,S-gain=0x0,S-time=0x2b6,M-gain=0xb,M-time=0x1a5e,L-gain=0x0,L-time=0x0,envChange=1,dcg=-1--1--1,Piris=0
rk_aiq_algo_ae_itf.cpp:254: Last-Res:FrmId=20,S-gain=0x5,S-time=0x8ca,M-gain=0x11,M-time=0x1a5e,L-gain=0x0,L-time=0x0

rk_aiq_ae_algo.cpp:5983: ================================ HDR-AE (enter)==============================
rk_aiq_ae_algo.cpp:6004: AecRun: SMeanLuma=9.342692, MMeanLuma=37.698597,LMeanLuma=0.000000,TmoMeanluma=37.571430,Isconverged=0,Longfrm=0
rk_aiq_ae_algo.cpp:6013: >>> Framenum=22 Cur Piris=0, Sgain=1.000000,Stime=0.002570,mgain=1.462177,mtime=0.025000,lgain=1.000000,ltime=0.000000
rk_aiq_ae_algo.cpp:3308: S-HighLightLuma=197.250000,S-Target=100.000000,S-GlobalLuma=9.342692,S-Target=19.959433
rk_aiq_ae_algo.cpp:3733: L-LowLightLuma=29.626642,L-Target=48.572094,L-GlobalLuma=37.698597,L-Target=77.620155
rk_aiq_ae_algo.cpp:6110: calc result:piris=0,sgain=1.000000,stime=0.005081,mgain=1.862087,mtime=0.025000,lgain=0.000000,ltime=0.000000
rk_aiq_ae_algo.cpp:6133: ================================ (exit)==============================
```

图3-2 Hdr模式AE LOG

Line1:

> Cur-Exp: FrmId=22,S-gain=0x0,S-time=0x2b6,M-gain=0xb,M-time=0x1a5e,L-gain=0x0,L-time=0x0,envChange=1,dcg=-1--1--1,Piris=0

当前帧的曝光参数信息。

成员名称	描述
FrmId	当前帧的帧号
S/M/L-gain	当前Hdr各帧对应的sensor曝光增益寄存器值。HDR 2帧模式时，S/M有效；HDR 3帧模式时，S/M/L皆有效。
S/M/L-time	当前Hdr各帧对应的sensor曝光时间寄存器值。HDR 2帧模式时，S/M有效；HDR 3帧模式时，S/M/L皆有效。
envChange	当前帧是否发生环境亮度突变。0：环境亮度无突变；1：环境亮度突变
dcg	当前帧对应的dcg模式，分别对应短中长3帧。Hdr 2帧模式时，前两个数值有效，分别代表短长帧的dcg模式；Hdr 3帧模式时，三个数值分别代表短中长的dcg模式。-1：sensor不支持dcg模式 或 sensor内部进行dcg模式的切换；0：LCG模式；1：HCG模式
pirs	当前帧对应的p-iris光圈步进电机位置。若Airis功能关闭，该参数无效，无意义。

Line2:

> Last-Res:FrmId=20,S-gain=0x5,S-time=0x8ca,M-gain=0x11,M-time=0x1a5e,L-gain=0x0,L-time=0x0

上次运行AE设置的新曝光参数，部分参数与（1）中含义一致，此处不再赘述。通过比较Line1与Line2的曝光参数LOG信息，可知当前曝光是否与新曝光一致，即新曝光是否已经生效。

Line3:

> ================================ HDR-AE (enter)==============================

进入AE控制算法模块，HDR-AE表示当前为HDR曝光模式。

Line4:

> AecRun: SMeanLuma=9.342692, MMeanLuma=37.698597,LMeanLuma=0.000000,TmoMeanluma=37.571430,Isconverged=0,Longfrm=0

成员名称	描述
S/M/LMeanLuma	当前Hdr各帧的亮度均值。HDR 2帧模式时，S/M有效；HDR 3帧模式时，S/M/L皆有效。
TmoMeanluma	当前帧TMO模块输出的亮度均值。
Isconverged	当前Hdr各帧曝光量是否收敛。0：曝光未收敛；1：曝光已收敛。
Longfrm	当前帧的长帧模式状态。0：长帧模式关闭；1：长帧模式开启。

Line5：

> Framenum=22 Cur Piris=0,
> Sgain=1.000000,Stime=0.002570,mgain=1.462177,mtime=0.025000,lgain=1.000000,ltime=0.000000

成员名称	描述
Framenum	当前帧的帧号
s/m/lgain	当前帧对应的sensor曝光增益值。HDR 2帧模式时，s/m有效；HDR 3帧模式时，s/m/l皆有效。
s/m/ltime	当前帧对应的sensor曝光时间值。HDR 2帧模式时，s/m有效；HDR 3帧模式时，s/m/l皆有效。
piris	当前帧对应的p-iris光圈等效增益值。若Airis功能关闭，该参数无效，无意义。

Line6:

> S-HighLightLuma=197.250000,S-Target=100.000000,S-GlobalLuma=9.342692,S-Target=19.959433

短帧控制算法LOG

成员名称	描述
S-HighLightLuma	当前短帧高亮区域亮度。
S-Target	当前短帧高亮区域目标亮度。
S-GlobalLuma	当前短帧全局区域平均亮度。
S-Target	当前短帧全局区域目标亮度。

Line7:

> L-LowLightLuma=29.626642,L-Target=48.572094,L-GlobalLuma=37.698597,L-Target=77.620155

长帧控制算法LOG

成员名称	描述
L-LowLightLuma	当前长帧暗区亮度
L-Target	当前长帧暗区目标亮度。
L-GlobalLuma	当前长帧全局区域平均亮度。
L-Target	当前长帧全局区域目标亮度。

Line8:

> calc result:piris=0,sgain=1.000000,stime=0.005081,mgain=1.862087,mtime=0.025000,lgain=0.000000,ltime=0.000000

AE控制算法输出的曝光结果

成员名称	描述
s/m/lgain	最终新sensor曝光增益值。HDR 2帧模式时，s/m有效；HDR 3帧模式时，s/m/l皆有效。
s/m/ltime	最终新sensor曝光时间值。HDR 2帧模式时，s/m有效；HDR 3帧模式时，s/m/l皆有效。
piris	最终新p-iris光圈等效增益值。若Airis功能关闭，该参数无效，无意义。

AWB

配置指南

Log level	描述以及使能建议
Error	默认使能 严重错误
Warning	默认使能 警告错误，算法内部可能针对该错误进行了异常处理.
Info	默认未使能 **基本调试信息:** 逐帧输出信息：wbgain结果 事件信息： **建议**：
debug	默认未使能 **基本调试信息(相对Info等级增加)**: 逐帧输出信息：白平衡收敛状态，环境亮度，白点数量，总的色温信息，概率大的前四个光源的概率等等 事件信息:awb的api调用后的相关属性信息打印 **建议：**当对输出log大小有限制时，可以采用该等级获取关键log信息用于debug
Verbose	默认未使能 **基本调试信息(相对Debug等级增加):** 逐帧输出信息：各个光源白点统计信息（白平衡增益，白点数量），及策略相关的中间量结果等等 事件信息： **建议：**推荐使用该等级抓完整的log用于debug

Sub modules bit	描述以及使能建议
bit[4:11]	无效

log解读

参考《**Rockchip_Color_Optimization_Guide_ISP2x_CN**》

AF

配置指南

Log level	描述以及使能建议
Error	默认使能 严重错误
Warning	默认未使能 警告错误，算法内部可能针对该错误进行了异常处理.
Info	默认未使能 **基本调试信息:** 逐帧输出信息：对焦搜索算法最终结果 事件信息： **建议**：
debug	默认未使能 **基本调试信息(相对Info等级增加)**: 逐帧输出信息：FV统计值，马达移动位置信息 事件信息: 对焦触发、清晰位置搜索、变焦、跟焦、对焦 **建议：** 推荐使用该等级抓完整的log用于debug
Verbose	默认未使能 **基本调试信息(相对Debug等级增加):** 无 逐帧输出信息： 事件信息： **建议：**无

Sub modules bit	描述以及使能建议
bit[4:11]	无效

log解读

MERGE

配置指南

Log level	描述以及使能建议
Info	默认未使能 **基本调试信息:** 逐帧输出信息：暂未使用 事件信息：暂未使用 **建议**：关闭
debug	默认未使能 **基本调试信息:** 逐帧输出信息：Merge调试信息 事件信息：暂未使用 **建议**：当画面亮度不符合预期，或者TMO之后对比度不够时候开启
Verbose	默认未使能 **基本调试信息(相对Verbose等级增加):** 逐帧输出信息：Mrerge曝光相关信息 事件信息：暂未使用 **建议：**当画面出现闪烁，且从AE log中确认是TMO在闪烁时开启，供RK人员debug使用

Sub modules bit	描述以及使能建议
bit[4:11]	无效

log解读

当模式为线性时，merge日志等级如下：

```
[AMERGE]:XCAM DEBUG rk_aiq_algo_amerge_itf.cpp:256: AmergeProcess FrameID:0, It's in Linear Mode, Merge function bypass
[AMERGE]:XCAM DEBUG rk_aiq_algo_amerge_itf.cpp:256: AmergeProcess FrameID:1, It's in Linear Mode, Merge function bypass
[AMERGE]:XCAM DEBUG rk_aiq_algo_amerge_itf.cpp:256: AmergeProcess FrameID:2, It's in Linear Mode, Merge function bypass
[AMERGE]:XCAM DEBUG rk_aiq_algo_amerge_itf.cpp:256: AmergeProcess FrameID:3, It's in Linear Mode, Merge function bypass
[AMERGE]:XCAM DEBUG rk_aiq_algo_amerge_itf.cpp:256: AmergeProcess FrameID:4, It's in Linear Mode, Merge function bypass
[AMERGE]:XCAM DEBUG rk_aiq_algo_amerge_itf.cpp:256: AmergeProcess FrameID:5, It's in Linear Mode, Merge function bypass
[AMERGE]:XCAM DEBUG rk_aiq_algo_amerge_itf.cpp:256: AmergeProcess FrameID:6, It's in Linear Mode, Merge function bypass
```

当模式为HDR且基准帧为长帧时，merge日志等级为10000000ff3时

```
[AMERGE]:XCAM DEBUG rk_aiq_algo_amerge_itf.cpp:143: AmergeProcess:/###################################Amerge Start###################################/
[AMERGE]:XCAM DEBUG rk_aiq_amerge_algo.cpp:1274: AmergeByPassProcessing: FrameID:0 HDRFrameNum:2 LongFrmMode:0 MergeApiMode:0 EnvLv:0.000000 MoveCoef:0.000000 bypass:0
[AMERGE]:XCAM DEBUG rk_aiq_amerge_algo.cpp:950: MergeDamping: Current BaseFrm:0 OECurve_smooth:80.000000 OECurve_offset:280.000000
[AMERGE]:XCAM DEBUG rk_aiq_amerge_algo.cpp:953: MergeDamping: Current MDCurveMS_smooth:80.000000 MDCurveMS_offset:38.000000 MDCurveLM_smooth:80.000000 MDCurveLM_offset:38.000000
[AMERGE]:XCAM DEBUG rk_aiq_algo_amerge_itf.cpp:253: AmergeProcess:/###################################Amerge Over###################################/
```

名称	描述
FrameID	帧号
HDRFrameNum	HDR帧数
LongFrmMode	长帧模式开关，0：关闭，1：开启
MergeApiMode	API模式
EnvLv	环境亮度
MoveCoef	运动变量
bypass	当前帧bypass开关，0：关闭，1：开启
BaseFrm	基准帧模式，0：长帧模式，1：短帧模式
OECurve_smooth	当前帧过曝曲线的斜率，取值范围[0,200]，由json中参数逆归一化得到，系数200。
OECurve_offset	当前帧过曝曲线的偏移值，取值范围[108,280]。
MDCurveMS_smooth	当前帧中帧和短帧之间运动曲线斜率，取值范围为[0,200]，由json中参数逆归一化得到，系数200。
MDCurveMS_offset	当前帧中帧和短帧之间运动曲线偏移值，取值范围为[0.26,100]，由json中参数逆归一化得到，系数100。
MDCurveLM_smooth	当前帧长帧和中帧之间运动曲线斜率，取值范围为[0,200]，由json中参数逆归一化得到，系数200。
MDCurveLM_offset	当前帧长帧和中帧之间运动曲线偏移值，取值范围为[0.26,100]，由json中参数逆归一化得到，系数100。

当模式为HDR且基准帧为短帧时，merge日志等级为10000000ff3时

```
[AMERGE]:XCAM DEBUG rk_aiq_algo_amerge_itf.cpp:143: AmergeProcess:/###################################Amerge Start###################################/
[AMERGE]:XCAM DEBUG rk_aiq_amerge_algo.cpp:1274: AmergeByPassProcessing: FrameID:6 HDRFrameNum:2 LongFrmMode:0 MergeApiMode:0 EnvLv:0.198943 MoveCoef:0.000000 bypass:0
[AMERGE]:XCAM DEBUG rk_aiq_amerge_algo.cpp:950: MergeDamping: Current BaseFrm:0 OECurve_smooth:80.000000 OECurve_offset:234.612076
[AMERGE]:XCAM DEBUG rk_aiq_amerge_algo.cpp:957: MergeDamping: Current MDCurve_Coef:0.050000 MDCurve_ms_thd0:0.000000 MDCurve_lm_thd0:0.000000
[AMERGE]:XCAM DEBUG rk_aiq_algo_amerge_itf.cpp:253: AmergeProcess:/###################################Amerge Over###################################/
```

名称	描述
FrameID	帧号
HDRFrameNum	HDR帧数
LongFrmMode	长帧模式开关，0：关闭，1：开启
MergeApiMode	API模式
EnvLv	环境亮度
MoveCoef	运动变量
bypass	当前帧bypass开关，0：关闭，1：开启
BaseFrm	基准帧模式，0：长帧模式，1：短帧模式
OECurve_smooth	当前帧过曝曲线的斜率，取值范围[0,200]，由json中参数逆归一化得到，系数200。
OECurve_offset	当前帧过曝曲线的偏移值，取值范围[108,280]。
MDCurve_Coef	当前帧控制系数，取值范围[0,1]，默认值为0.05，精度0.0001。
MDCurve_ms_thd0	当前帧中短帧控制系数，取值范围[0,1]，默认值为0.0，精度0.1。
MDCurve_lm_thd0	当前帧长中帧控制系数，取值范围[0,1]，默认值为0.0，精度0.1

DRC

配置指南

Log level	描述以及使能建议
Info	默认未使能 **基本调试信息:** 逐帧输出信息：暂未使用 事件信息：暂未使用 **建议**：关闭
debug	默认未使能 **基本调试信息:** 逐帧输出信息：DRC调试信息 事件信息：暂未使用 **建议**：当画面亮度不符合预期，或者TMO之后对比度不够时候开启
Verbose	默认未使能 **基本调试信息(相对Verbose等级增加):** 逐帧输出信息：DRC曝光相关信息 事件信息：暂未使用 **建议：**当画面出现闪烁，且从AE log中确认是TMO在闪烁时开启，供RK人员debug使用

Sub modules bit	描述以及使能建议
bit[4:11]	无效

log解读

DRC日志等级为20ff3时

```
|[ATMO]:XCAM DEBUG rk_aiq_algo_adrc_itf.cpp:157: processing:////////////////////////////////////ADRC Start////////////////////////////////////
|[ATMO]:XCAM DEBUG rk_aiq_adrc_algo.cpp:1834: AdrcByPassProcessing: FrameID:4 HDRFrameNum:2 LongFrmMode:0 DRCApiMode:0 EnvLv:0.282684 bypass:0
|[ATMO]:XCAM DEBUG rk_aiq_adrc_algo.cpp:1543: AdrcTuningParaProcessingV30: Current Enable:1 DrcGain:1.000000 Alpha:0.200000 Clip:16.000000 Strength:0.000000 CompressMode:0
|[ATMO]:XCAM DEBUG rk_aiq_adrc_algo.cpp:1546: AdrcTuningParaProcessingV30: Current LocalWeit:1.000000 LocalAutoEnable:1 LocalAutoWeit:0.037477 GlobalContrast:0.000000 LoLitContrast:0.000000
|[ATMO]:XCAM DEBUG rk_aiq_adrc_algo.cpp:1283: AdrcDampingV30: Current damp DrcGain:1.000000 Alpha:0.200000 Clip:16.000000 Strength:0.000000 CompressMode:0
|[ATMO]:XCAM DEBUG rk_aiq_adrc_algo.cpp:1286: AdrcDampingV30: Current damp LocalWeit:1.000000 LocalAutoEnable:1 LocalAutoWeit:0.037477 GlobalContrast:0.000000 LoLitContrast:0.000000
|[ATMO]:XCAM DEBUG rk_aiq_algo_adrc_itf.cpp:251: processing:////////////////////////////////////ADRC Over////////////////////////////////////
```

名称	描述
FrameID	帧号
HDRFrameNum	HDR帧数
LongFrmMode	长帧模式开关，0：关闭，1：开启
DRCApiMode	API模式
EnvLv	环境亮度
bypass	当前帧bypass开关，0：关闭，1：开启
Enable	DRC开关
DrcGain	当前帧DRC模块增益，取值范围[1,8]。
Alpha	当前帧DrcGain Alpha值，取值范围[0,1]。
Clip	当前帧DrcGain Clip值，取值范围[0,64]。
Strength	当前帧HiLight高光区域细节，取值范围[0,1]。
CompressMode	当前帧CompressSetting模式。
LocalWeit	当前帧Local权重，取值范围[0,1]，0：Global，1：全Local，默认值0。
LocalAutoEnable	当前帧自动LocalWeit开关，取值范围[0,1]，默认值为1，精度1。
LocalAutoWeit	当前帧自动LocalWeit值，取值范围[0,1]，默认值为0.4，精度0.01。
GlobalContrast	当前帧全局对比度，取值范围[0,1]，默认值为0，精度0.01。
LoLitContrast	当前帧低量区对比度，取值范围[0,1]，默认值为0，精度0.01。

NR&Sharp

配置指南

Log level	描述以及使能建议
error	默认使能 **基本调试信息:** 逐帧输出信息：严重错误地方，可能会导致软件崩溃的打印 事件信息：暂未使用 **建议：**开启
Info	默认未使能 **基本调试信息:** 逐帧输出信息：所有函数调用信息，没有寄存器值打印。整体信息较多。 事件信息：暂未使用 **建议**：效果没有异常出错时候，不建议打开此项。一般建议使用io命令打印寄存器值。
debug	默认未使能 **基本调试信息(相对Info等级增加):** 逐帧输出信息：所有函数调用信息，包括寄存器值打印，寄存器值打印信息较多。 事件信息:暂未使用 **建议**：当效果出现严重错误时候，建议打开此项，提供给rk debug使用。

Sub modules bit	描述以及使能建议
bit[4:11]	无效

log解读

Dhz&Ehz

配置指南

Log level	描述以及使能建议
Info	默认未使能 **基本调试信息:** 逐帧输出信息：暂未使用 事件信息：暂未使用 **建议**:
debug	默认未使能 **基本调试信息(相对Info等级增加):** **逐帧输出信息：Dhz&Ehz调试信息** **事件信息:** 建议：**
Verbose	默认未使能 **基本调试信息(相对Verbose等级增加):** 逐帧输出信息：暂未使用 事件信息：暂未使用 **建议:**

Sub modules bit	描述以及使能建议
bit[4:11]	无效

Log解读

当Enhance功能开启时，Dehaze日志等级为2000ff3时

```
[ADEHAZE]:XCAM DEBUG rk_aiq_algo_adhaz_itf.cpp:145: /************************Adehaze Start******************/
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:2402: AdehazeByPassProcessing:FrameID:5 byPassProc:0 ISO:50.000000
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:1423:  AdehazeEnhanceApiBypassV30Process: Adehaze Api off!!!
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:223:  EnableSettingV30: Dehaze module en:1 Dehaze en:0, Enhance en:1, Hist en:0
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:1022: GetEnhanceParamsV30 EnvLv:0.457436 enhance_value:1.300000 enhance_chroma:1.000000
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:1024: GetEnhanceParamsV30 enhance_value_reg:0x533 enhance_chroma_reg:0x400
[ADEHAZE]:XCAM DEBUG rk_aiq_algo_adhaz_itf.cpp:188: /************************Adehaze over******************/
```

名称	描述
FrameID	帧号
byPassProc	当前帧bypass开关，0：关闭，1：开启
ISO	ISO
Dehaze module en	Dehaze模块开关
Dehaze en	Dehaze功能开关
Enhance en	Enhance功能开关
Hist en	Hist功能开关
EnvLv	当前帧环境亮度
enhance_value	当前帧通用对比度力度，取值范围[0，16]，推荐范围[1, 2]。
enhance_chroma	当前帧色度的增强调节参数，取值范围[0，16]，推荐范围[1, 2]。
enhance_value_reg	当前帧通用对比度力度reg值。
enhance_chroma_reg	当前帧色度的增强调节参数reg值。

当Dehaze功能开启且cfg_alpha=0时，Dehaze日志等级为2000ff3时

```
[ADEHAZE]:XCAM DEBUG rk_aiq_algo_adhaz_itf.cpp:145: /************************Adehaze Start******************/
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:2402: AdehazeByPassProcessing:FrameID:3 byPassProc:0 ISO:50.000000
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:1423:  AdehazeEnhanceApiBypassV30Process: Adehaze Api off!!!
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:223:  EnableSettingV30: Dehaze module en:1 Dehaze en:1, Enhance en:0, Hist en:0
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:960: GetDehazeParamsV30 cfg_alpha:0 EnvLv:0.433231 air_max:250.000000 air_min:200.000000 tmax_base:125.000000 wt_max:0.900000
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:961: GetDehazeParamsV30 cfg_alpha_reg:0x0 air_max:0xfa air_min:0xc8 tmax_base:0x7d wt_max:0xe6
[ADEHAZE]:XCAM DEBUG rk_aiq_algo_adhaz_itf.cpp:188: /************************Adehaze over******************/
```

名称	描述
FrameID	帧号
byPassProc	当前帧bypass开关，0：关闭，1：开启
ISO	ISO
Dehaze module en	Dehaze模块开关
Dehaze en	Dehaze功能开关
Enhance en	Enhance功能开关
Hist en	Hist功能开关
cfg_alpha	软件配置占比，取值范围[0,1]，默认值1，精度0.01。
EnvLv	当前帧环境亮度
air_max	当前帧air自适应的最大值限制，取值范围[230, 250]，默认值250。
air_min	当前帧air自适应的最小值限制，取值范围[230, 250]，默认值200。
tmax_base	当前帧tmax自适应基础值，默认125，对应配置如下，200(131)，210(125)，220(119)，230(114)，240(109)，250(105)，推荐131-105
wt_max	当前帧wt自适应的最大值限制，取值范围[0.75, 0.9]，默认值0.9。

当Dehaze功能开启且cfg_alpha=1时，Dehaze日志等级为2000ff3时

```
[ADEHAZE]:XCAM DEBUG rk_aiq_algo_adhaz_itf.cpp:145: /*************************Adehaze Start******************/
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:2402: AdehazeByPassProcessing:FrameID:4 byPassProc:0 ISO:50.000000
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:1423:  AdehazeEnhanceApiBypassV30Process: Adehaze Api off!!!
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:223:  EnableSettingV30: Dehaze module en:1 Dehaze en:1, Enhance en:0, Hist en:0
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:954: GetDehazeParamsV30 cfg_alpha:1 EnvLv:0.467077 cfg_air:210.000000 cfg_tmax:0.200000 cfg_wt:0.800000
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:955: GetDehazeParamsV30 cfg_alpha_reg:0x255 cfg_air:0xd2 cfg_tmax:0xcc cfg_wt:0xcc
[ADEHAZE]:XCAM DEBUG rk_aiq_algo_adhaz_itf.cpp:188: /*************************Adehaze over******************/
```

名称	描述
FrameID	帧号
byPassProc	当前帧bypass开关，0：关闭，1：开启
ISO	ISO
Dehaze module en	Dehaze模块开关
Dehaze en	Dehaze功能开关
Enhance en	Enhance功能开关
Hist en	Hist功能开关
cfg_alpha	软件配置占比，取值范围[0,1]，默认值1，精度0.01。
EnvLv	当前帧环境亮度
cfg_air	当前帧软件配置air，大气光系数，取值范围[0, 255]，默认值210。
cfg_tmax	当前帧软件配置tmax，去雾的最大值，取值范围[0, 1]，默认值0.2。
cfg_wt	当前帧软件配置wt，图像去雾力度，取值范围[0, 1]，默认值0.8。

当Hist功能开启时，Dehaze日志等级为2000ff3时

```
[ADEHAZE]:XCAM DEBUG rk_aiq_algo_adhaz_itf.cpp:145: /************************Adehaze Start*****************/
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:2402: AdehazeByPassProcessing:FrameID:3 byPassProc:0 ISO:50.000000
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:1423:  AdehazeEnhanceApiBypassV30Process: Adehaze Api off!!!
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:223:  EnableSettingV30: Dehaze module en:1 Dehaze en:0, Enhance en:0, Hist en:1
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:1137: GetHistParamsV30 cfg_alpha:0.000000 EnvLv:0.452185 hist_para_en:1 hist_gratio:2.000000 hist_th_off:64.000000 hist_k:2.000000 hist_min:0.015000 hist_scale:0.090000 cfg_gratio:2.0000
[ADEHAZE]:XCAM DEBUG rk_aiq_adehaze_algo.cpp:1139: GetHistParamsV30 cfg_alpha_reg:0x0 hist_gratio_reg:0x10 hist_th_off_reg:0x40 hist_k_reg:0x8 hist_min_reg:0x3 hist_scale_reg:0x17 cfg_gratio_reg:0x200
[ADEHAZE]:XCAM DEBUG rk_aiq_algo_adhaz_itf.cpp:188: /************************Adehaze over*****************/
```

名称	描述
FrameID	帧号
byPassProc	当前帧bypass开关，0：关闭，1：开启
ISO	ISO
Dehaze module en	Dehaze模块开关
Dehaze en	Dehaze功能开关
Enhance en	Enhance功能开关
Hist en	Hist功能开关
cfg_alpha	软件配置占比，取值范围[0,1]，默认值1，精度0.01。
EnvLv	当前帧环境亮度
hist_para_en	当前帧直方图拉伸控制开关。
hist_gratio	当前帧直方图拉伸倍数，直方图均衡控制系数，取值范围[0, 32]。
hist_th_off	当前帧直方图统计阈值，取值范围[0, 255]，默认值64。
hist_k	当前帧直方图自适应阈值放大倍数，取值范围[0, 7)，默认值2。
hist_min	当前帧直方图统计阈值的最小值，取值范围[0,2)，默认值0.016。
hist_scale	当前帧直方图均衡控制系数，取值范围[0, 32]。
cfg_gratio	当前帧软件配置直方图拉伸倍数，直方图均衡控制系数，取值范围[0, 32)。

CamHW

配置指南

Log level	描述以及使能建议
Error	默认使能 严重错误
Warning	默认未使能 警告错误，算法内部可能针对该错误进行了异常处理.
Info	默认未使能 **基本调试信息:** 逐帧输出信息： 事件信息： **建议：**
Debug	默认未使能 **基本调试信息(相对Info等级增加)**: 逐帧输出信息： 事件信息: **建议：**
Verbose	默认未使能 **基本调试信息(相对Debug等级增加):** 逐帧输出信息： 事件信息： **建议：**存在单次大量信息输出，谨慎使用
Low1	默认未使能 **基本调试信息(相对Debug等级增加):** 逐帧输出信息： 事件信息：函数调用路径信息 **建议：**

Sub modules bit	描述以及使能建议
bit[4]	30 子模块log开关。 负责 HWI 流程控制部分。 **建议**: 有以下各子模块问题时建议一起使能。
bit[5]	Isp30Params 及 Isp30PollThread 子模块log开关。 负责 isp 参数、数据流、事件等处理部分。 **建议**: 疑似参数与视频帧未匹配、时序设置异常等问题 回读离线模式下，数据流断流等问题
bit[6]	SensorHw 子模块log开关。 负责 camera sensor 相关部分。 **建议**: 疑似CIS曝光设置等问题
bit[7]	FlashLight 子模块log开关。 负责补光灯控制部分。 **建议**: 补光灯控制问题。
bit[8]	LensHw 子模块log开关。 负责镜头马达控制部分。 **建议**: 马达控制等问题
bit[9]	30SpThread 子模块开关。 内部模块保留使用。
bit[10:11]	无效

log解读

- **CamHwIsp20 子模块**

```
[CAMHW]:XCAM INFO CamHwIsp20.cpp:519: model(rkisp0): isp_info(0): ispp-subdev entity name: /dev/v4l-subdev5

[CAMHW]:XCAM INFO CamHwIsp20.cpp:343: model(rkispp0): ispp_info(0): ispp-subdev entity name: /dev/v4l-subdev0

[CAMHW]:XCAM INFO CamHwIsp20.cpp:932: match the sensor_name(m01_f_imx415 1-001a) media link

[CAMHW]:XCAM INFO CamHwIsp20.cpp:967: vicap rkcif_mipi_lvds, linked_vicap rkcif_mipi_lvds
[CAMHW]:XCAM INFO CamHwIsp20.cpp:973: vicap link to isp(0) to ispp(0)

[CAMHW]:XCAM INFO CamHwIsp20.cpp:979: sensor m01_f_imx415 1-001a adapted to pp media 2:/dev/media2
```

上图log 仅在 AIQ 库加载时的打印一次，从中能得到camera sensor、isp 及 ispp 之间的链接关系，该信息从解析media节点拓扑结构得到，从该信息可知道camera sensor是否正常探测到。

```
CamHwIsp20.cpp:3427: ispparam ens 0xcfffe473b, en_up 0xdfffee73f, cfg_up 0xdfffe473b
CamHwIsp20.cpp:3431: device(/dev/video15) queue buffer index 0, queue cnt 1, check exit status again[exit:
```

上图log表示的是hwi层往isp驱动配置新参数，目前基本每一帧都会配置，具体含义如下表所示:

成员名称	描述
ispparam ens	模块使能位，模块bit位定义具体参考 hwi/isp20/rk_isp20_hw.h
en_up	模块使能更新位，对应位为1表示该模块使能位需要更新
cfg_up	模块参数更新位，对应位为1表示该模块参数需要更新
device	ISP参数模块对应的video节点

```
[CAMHW]:XCAM DEBUG CamHwIsp20.cpp:3737: ispp full en update 0x7, ens 0x7, cfg update 0x7
[CAMHW]:XCAM DEBUG CamHwIsp20.cpp:3758: ispp new en update 0x0, ens 0x7, cfg update 0x7
[CAMHW]:XCAM DEBUG CamHwIsp20.cpp:3761: module_init_ens frome drv 0x7

[CAMHW]:XCAM DEBUG CamHwIsp20.cpp:3789: device(/dev/video23) queue buffer index 0, queue cnt 1, check exit status again[exit:
```

上图log表示的是hwi层往ispp驱动配置新参数，目前基本每一帧也都会配置，具体说明前述isp模块。

- **Isp20PollThread 子模块****

```
[CAMHW]:XCAM DEBUG Isp20PollThread.cpp:976: frame[48]: sof_ts 2800346ms, frame_ts 2800379ms, globalTmo(0), readback(1)
```

上图log为回读模式时启动一帧回读，只要工作在回读模式，正常运行时每一帧都会打印，具体含义如下表：

成员	描述
frame	帧ID，表示第几帧RAW，从0开始。
sof_ts	该帧 SOF的时间戳
frame_ts	采集到ddr的完成时刻时间戳
readback	isp回读次数

```
[CAMHW]:XCAM DEBUG Isp20PollThread.cpp:679: handle_rx_buf dev_index:0 index:0 fd:30

[CAMHW]:XCAM DEBUG Isp20PollThread.cpp:769: handle_tx_buf dev_index:0 sequence:49
```

上图log为回读模式时RAW buf的轮转过程，只要工作在回读模式时，每一帧都会打印该log，具体含义如下表：

成员	描述
handle_rx_buf	isp处理完一帧，还给vicap/isp_tx
dev_index	长、中及短帧设备索引号，handle_tx_buf 及 handle_rx_buf 的log中dev_index相同时，表示同一路数据
sequence	帧号，从0开始

- **SensorHw 子模块**

```
[CAMHW]:XCAM DEBUG SensorHw.cpp:685: handle_sof: frameid=13, exp_list size=1, gain_list size=0
[CAMHW]:XCAM DEBUG SensorHw.cpp:726: handle_sof: working_mode=0,frameid=13, status: set_time=1,set_gain=0

[CAMHW]:XCAM DEBUG SensorHw.cpp:180: setLinearSensorExposure: frameid: 13, a-gain: 17, time: 2024, dcg: -1, snr: 0
```

上图log为线性模式时设置新曝光到sensor驱动流程，有新曝光时才会调用，具体含义如下表：

成员	描述
frameid	sof 事件帧号
handle_sof	sof 事件回调函数，每一帧回调一次，曝光设置在该回调中处理
exp_list size	曝光列表中还有多少个新曝光时间未设置到驱动
gain_list size	曝光列表中有多少个新曝光增益未设置到驱动
working_mode	当前工作模式， 0 为 normal
set_time	该sof消息中是否有新曝光时间需要设置到驱动
set_gain	该sof消息中是否有新曝光增益需要设置到驱动
a-gain, time	新的曝光时间、增益寄存器值

如何采集Raw/YUV图像

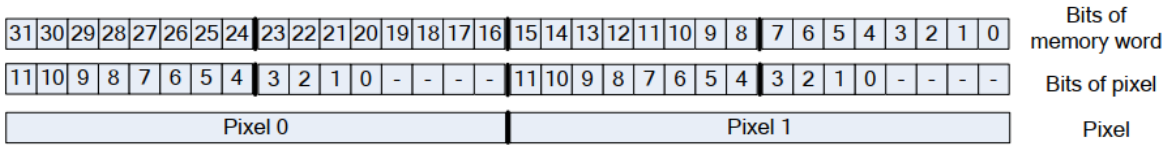
Raw数据特指CIS原始输出的数据，未经过任何图像处理。针对Bayer raw sensor， Raw数据即8/12/14 bit bayer rgb 数据。一般情况下，以下几种情况需要获取CIS Raw数据：

1. ISP处理后的YUV数据输出异常的情况下，希望动态截存此时ISP输入的Raw数据分析问题是否是CIS问题
2. 图像质量效果调试前，需要采集CIS Raw数据进行模组参数的标定

Raw数据存储格式

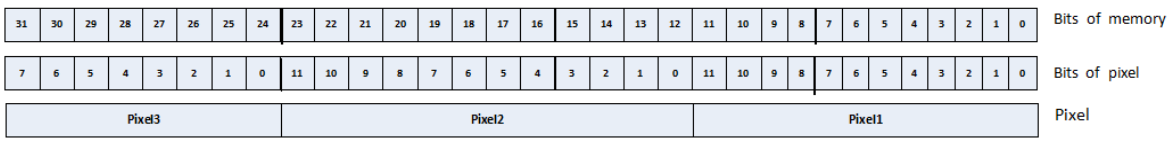
非紧凑型存储格式

对于raw12数据在内存中的存储排列方式，以4字节的内存片段为例，数据的存储方式如下所示：



紧凑型存储格式

对于raw12数据在内存中的存储排列方式，以4字节的内存片段为例，数据的存储方式如下所示：



RK-RAW V1.0

项目	参数名称	数据类型定义	长度	描述
文件头	Identifier	unsigned short	2	固定 0x8080
	Header length	unsigned short	2	固定 128
	Frame index	unsigned int	4	帧索引号
	Width	unsigned short	2	图像宽度
	Height	unsigned short	2	图像高度
	Bit depth	unsigned char	1	图像位宽
	Bayer format	unsigned char	1	0: BGGR; 1: GBRG; 2: GRBG; 3: RGGB;
	Number of HDR Frame	unsigned char	1	帧类型： 1：表示线性模式，短帧 2：表示 2 帧 HDR，长帧+短帧 3：表示 3 帧 HDR，长帧+中帧+短帧
	Current Frame type	unsigned char	1	当前帧类型： 1：短帧 2：中帧 3：长帧
	Storage type	unsigned char	1	0：紧凑型 1：非紧凑型
	Line stride	unsigned short	2	单位为字节
	Effective line stride	unsigned short	2	单位为字节
	Reserved	unsigned char	107	保留段
RAW DATA	RAW DATA	RAW	W * H *BPP	RAW DATA

RK-RAW V2.0

RK-RAW V2.0 格式由起始标识符、数据块以及结束标识符构成。

格式数据块定义

数据块有三部分组成：标识符ID，块长度，块数据。每个标识符ID都有唯一的固定值。

数据块定义				
项目	数据类型	长度(Byte)	默认值	说明
起始标识符	u16	2	0xFF00	固定值：文件起始
块标识符	u16	2	0xFF01	标识符：Raw信息
块长度	u32	4	-	
块数据				
块标识符	u16	2	0xFF02	标识符：Normal模式　Raw数据
块长度	u32	4	-	
块数据				
块标识符	u16	2	0xFF03	标识符：HDR2/HDR3短帧　Raw数据
块长度	u32	4	-	
块数据				
块标识符	u16	2	0xFF04	标识符：HDR2长帧/HDR3中帧　Raw数据
块长度	u32	4	-	
块数据				
块标识符	u16	2	0xFF05	标识符：HDR3长帧　Raw数据
块长度	u32	4	-	
块数据				
块标识符	u16	2	0xFF06	标识符：帧统计信息
块长度	u32	4	-	
块数据				
块标识符	u16	2	0xFF07	标识符：ISP寄存器格式
块长度	u32	4	-	
块数据				
块标识符	u16	2	0xFF08	标识符：ISP寄存器
块长度	u32	4	-	
块数据				
块标识符	u16	2	0xFF09	标识符：ISPP寄存器格式
块长度	u32	4	-	
块数据				
块标识符	u16	2	0xFF0a	标识符：ISPP寄存器
块长度	u32	4	-	
块数据				
块标识符	u16	2	0xFF0b	标识符：平台信息
块长度	u32	4	-	
块数据				
...				
结束标识符	u16	2	0x00FF	固定值：文件结尾

图表中的数据块并不都是必需的，可以选择其中的几个数据块组合使用。例如，一个RKRawV2文件可能只包含'Raw信息块'、'Raw数据块'和'帧统计信息块'，我们建议至少包含这三个数据块，因为这样的文件才有意义。

Raw信息块定义

数据块定义：Raw信息 0xFF01				
项目	数据类型	长度(Byte)	默认值	说明
版本号	u16	2	0x0200	版本号：0x0200=v2.0
Sensor名	string	32	-	Sensor型号
场景/光源	string	32	-	采集场景/光源
帧号	u32	4	-	帧号
宽度	u16	2	-	图像宽度,单位为像素
高度	u16	2	-	图像高度,单位为像素
位宽	u8	1	-	图像位宽
Bayer格式	u8	1	-	0(BGGR);1(GBRG); 2(GRBG);3(RGGB);
HDR合成帧数	u8	1	-	1(线性模式);2(长+短帧);3(长+中+短帧);
存储格式	u8	1	-	0(紧凑);1(非紧凑);
行长	u16	2	-	单位为字节
有效行长	u16	2	-	单位为字节
大小端	u8	1	-	0(大端);1(小端);

Raw数据块定义

该数据段中可以存放Raw图像数据，也可以存放指向Raw图像数据的buffer fd或虚拟地址。在使用相关API的时候需要传参指明该数据块中存储的类型，具体请参见rk_aiq_rawbuf_type_t。

数据块定义：Raw数据 0xFF02 - 0xFF05			
项目	数据类型	长度(Byte)	说明
Raw数据	-	长*宽*N 非紧凑：N=2 紧凑：N=bpp/8	Raw数据

帧统计信息块定义

数据块：帧统计信息 0xFF06				
项目	数据类型	长度(Byte)	默认值	说明
版本号	u16	2	0x0200	Raw格式版本号：0x0200=v2.0
帧号	u32	4	-	帧号
Normal_Exp	float	4	-	线性模式：快门时间
Normal_Gain	float	4	-	线性模式：曝光增益
Normal_Exp_REG	u32	4	-	线性模式：快门时间的SENSOR寄存器值
Normal_Gain_REG	u32	4	-	线性模式：曝光增益的SENSOR寄存器值
HDR_Exp_L	float	4	-	HDR3：长帧快门时间
HDR_Gain_L	float	4	-	HDR3：长帧曝光增益
HDR_Exp_L_REG	u32	4	-	HDR3：长帧快门时间的SENSOR寄存器值
HDR_Gain_L_REG	u32	4	-	HDR3：长帧曝光增益的SENSOR寄存器值
HDR_Exp_M	float	4	-	HDR3：中帧快门时间 HDR2：长帧快门时间
HDR_Gain_M	float	4	-	HDR3：中帧曝光增益 HDR2：长帧曝光增益
HDR_Exp_M_REG	u32	4	-	HDR3：中帧快门时间的SENSOR寄存器值 HDR2：长帧快门时间的SENSOR寄存器值
HDR_Gain_M_REG	u32	4	-	HDR3：中帧曝光增益的SENSOR寄存器值 HDR2：长帧曝光增益的SENSOR寄存器值
HDR_Exp_S	float	4	-	HDR3：短帧快门时间 HDR2：短帧快门时间
HDR_Gain_S	float	4	-	HDR3：短帧曝光增益 HDR2：短帧曝光增益
HDR_Exp_S_REG	u32	4	-	HDR3：短帧快门时间的SENSOR寄存器值 HDR2：短帧快门时间的SENSOR寄存器值
HDR_Gain_S_REG	u32	4	-	HDR3：短帧曝光增益的SENSOR寄存器值 HDR2：短帧曝光增益的SENSOR寄存器值
AWB_Rgain	float	4	-	白平衡增益
AWB_Bgain	float	4	-	

寄存器格式块定义

数据块定义：寄存器格式 0xFF07/0xFF09				
项目	数据类型	长度(Byte)	默认值	说明
版本号	u16	2	0x0200	版本号：0x0200=v2.0
帧号	u32	4	-	寄存器基地址
基地址	u32	4	-	寄存器基地址
偏移地址	u32	4	-	起始偏移地址
数目	u32	4	-	寄存器数目

寄存器块定义

数据块定义：寄存器 0xFF08/0xFF0a			
项目	数据类型	长度(Byte)	说明
寄存器数据	-	-	寄存器数据

平台信息块定义

数据块定义：平台信息 0xFF0b			
项目	数据类型	长度(Byte)	说明
芯片型号	string	32	
ISP版本	string	32	
AIQ版本	string	32	

Raw/YUV数据采集方式

错误码

错误代码	描述
0	成功
-1	失败
-2	参数无效
-3	内存不足
-4	文件操作失败
-5	ANALYZER模块出错
-6	ISP模块出错
-7	sensor驱动出错
-8	线程操作出错
-9	IOCTL操作出错
-10	时序出错
-20	超时
-21	超出范围
-255	未知错误

缩略语

缩写	全称
CIS	Camera Image Sensor
RkAiq	Rockchip Automatical Image Quality
ISP	Image Signal Process
IQ Tuning	Image Quality Tuning