Rockchip Developement Guide ISP2x

文件标识：RK-KF-GX-601

发布版本：V1.2.0

日期：2020-09-25

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
RV1126/RV1109	Linux 4.19

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
RV1126	Y	N	N	N
RV1109	Y	N	N	N

修订记录

版本号	作者	修改日期	修改说明
V1.0.0	钟以崇 张云龙 徐鸿飞	2020-06-09	初始版本
V1.0.1	张云龙 徐鸿飞 朱林靖 池晓芳	2020-08-04	增加统计信息章节
V1.1.0	张云龙	2020-08-14	增加背光补偿、强光抑制等接口说明
v1.2.0	胡克俊	2020-9-25	增加AF统计值说明

目录

概述

ISP20 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。

ISP20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

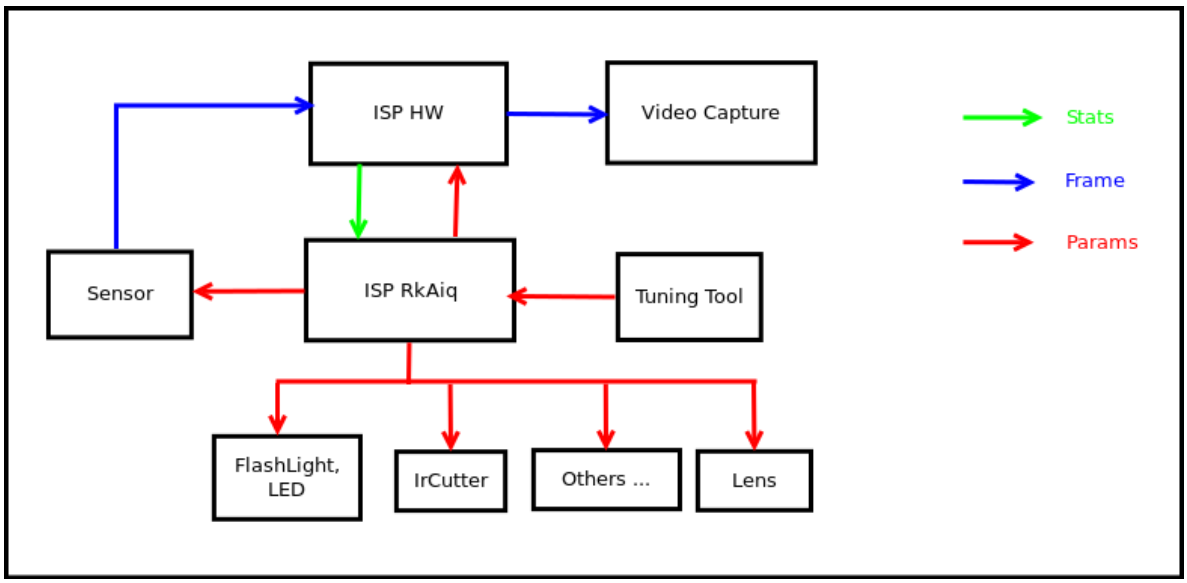

图1-1 ISP20 系统框图

ISP20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

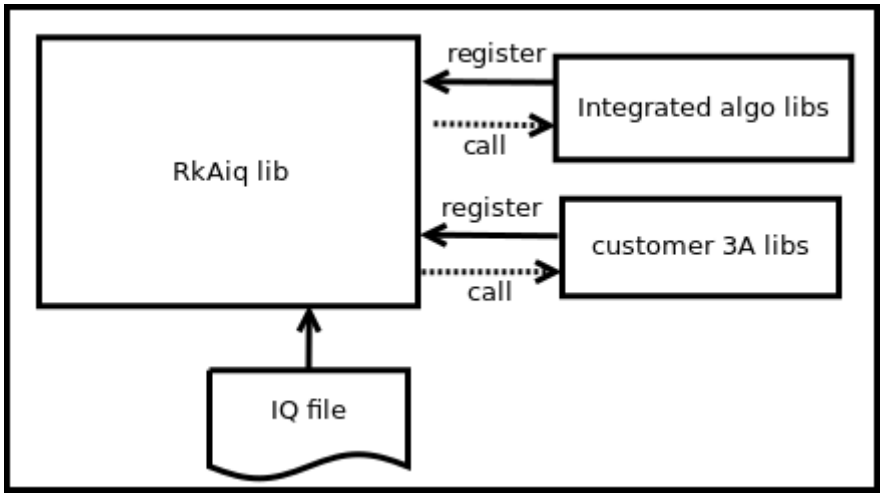

图1-2 RkAiq总体架构图

ISP20 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

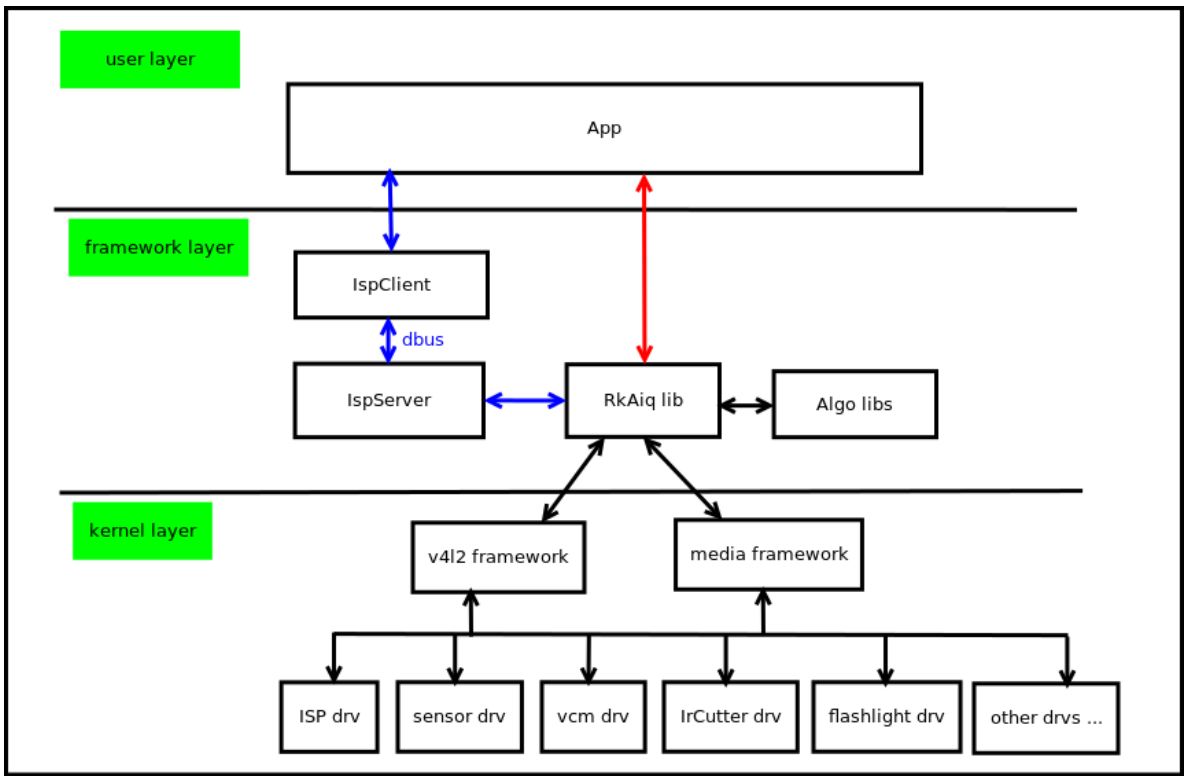

图1-3 软件架构框图

ISP20 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

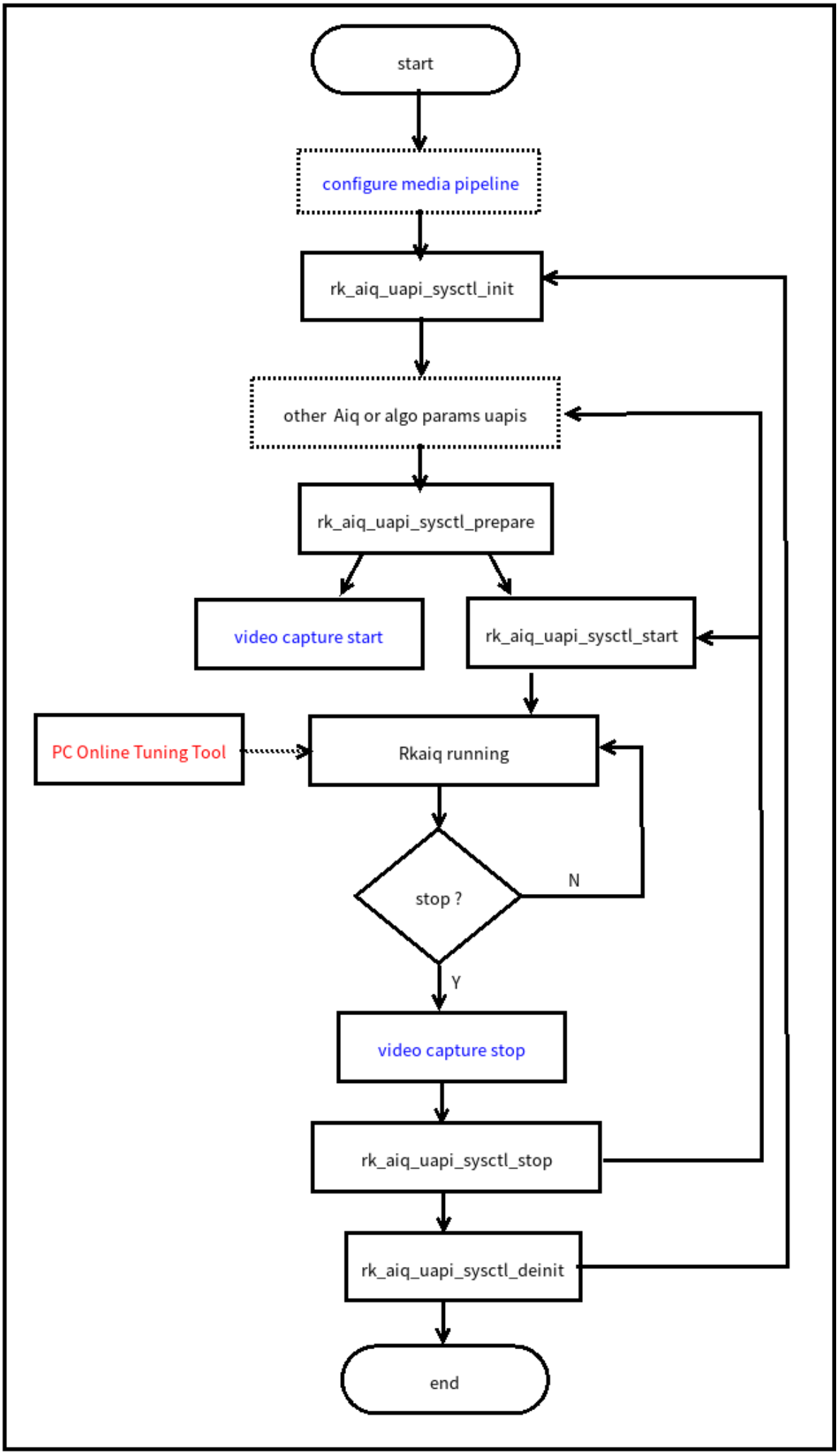
start
configure media pipeline
rk_aiq_uapi_sysctl_init
other Aiq or algo params uapis
rk_aiq_uapi_sysctl_prepare
video capture start
rk_aiq_uapi_sysctl_start
PC Online Tuning Tool
Rkaiq running
stop ?
N
Y
video capture stop
rk_aiq_uapi_sysctl_stop
rk_aiq_uapi_sysctl_deinit
end

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ISP20 pipeline，如sensor输出分辨率等等，驱动已有默认配置。
- rk_aiq_uapi_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_uapi_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_uapi_sysctl_prepare后调用。
- rk_aiq_uapi_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_uapi_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_uapi_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。其中功能级别API是基于模块级别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提供对该模块的详细参数设置以及查询，未对功能进行API区分。

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_uapi_sysctl_init

【描述】
初始化AIQ上下文。

【语法】

```
rk_aiq_sys_ctx_t*
rk_aiq_uapi_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_api_sysctl.h
- 库文件：librkaiq.so

【注意】

- 应先于其他函数调用。

rk_aiq_uapi_sysctl_deinit

【描述】
反初始化AIQ上下文环境。

【语法】

```
void
rk_aiq_uapi_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_api_sysctl.h
- 库文件：librkaiq.so

【注意】

- 不应在AIQ处于start状态调用。

rk_aiq_uapi_sysctl_prepare

【描述】
准备AIQ运行环境。

【语法】

```
XCamReturn
rk_aiq_uapi_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_api_sysctl.h
- 库文件：librkaiq.so

【注意】

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

rk_aiq_uapi_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_uapi_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_api_sysctl.h
- 库文件：librkaiq.so

【注意】

- 应在rk_aiq_uapi_sysctl_prepare函数之后调用。

rk_aiq_uapi_sysctl_stop

【描述】
停止AIQ控制系统。

【语法】

```
XCamReturn
rk_aiq_uapi_sysctl_stop(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_api_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi_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_uapi_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_api_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi_sysctl_enumStaticMetas

【描述】
枚举AIQ获取到的静态信息。

【语法】

```
XCamReturn
rk_aiq_uapi_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_api_sysctl.h

- 库文件：librkaiq.so

rk_aiq_uapi_sysctl_setModuleCtl

【描述】
AIQ模块开关设置。

【语法】

```
XCamReturn
rk_aiq_uapi_sysctl_setModuleCtl(const rk_aiq_sys_ctx_t* ctx, rk_aiq_module_id_t
mId, bool mod_en);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mId	模块ID	输入
mod_en	true为开启，false为关闭	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_getModuleCtl

【描述】
AIQ模块状态查询。

【语法】

```
XCamReturn
rk_aiq_uapi_sysctl_getModuleCtl(const rk_aiq_sys_ctx_t* ctx, rk_aiq_module_id_t
mId, bool *mod_en);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mId	模块ID	输入
mod_en	当前状态	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_api_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi_sysctl_regLib

【描述】
注册自定义算法库。

【语法】

```
XCamReturn
rk_aiq_uapi_sysctl_regLib(const rk_aiq_sys_ctx_t* ctx,
                          RkAiqAlgoDesComm* algo_lib_des);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
algo_lib_des	算法描述结构体，字段id为AIQ生成的标识ID	输入&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_api_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi_sysctl_unRegLib

【描述】
注销自定义算法库。

【语法】

```
XCamReturn
rk_aiq_uapi_sysctl_unRegLib(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
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_enableAxlib

【描述】
设置自定义算法库运行状态。

【语法】

```
XCamReturn
rk_aiq_uapi_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_api_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_uapi_sysctl_getAxlibStatus

【描述】
获取算法库状态。

【语法】

```
bool
rk_aiq_uapi_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_api_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi_sysctl_getEnabledAxlibCtx

【描述】
获取使能算法库的上下文结构体。

【语法】

```
const RkAiqAlgoContext*
rk_aiq_uapi_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_api_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_uapi_sysctl_enableAxlib之后调用，否则将返回NULL。

rk_aiq_uapi_sysctl_setCpsLtCfg

【描述】
设置补光灯控制信息。

【语法】

```
XCamReturn
rk_aiq_uapi_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_api_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi_sysctl_getCpsLtInfo

【描述】
获取补光灯控制信息。

【语法】

```
XCamReturn
rk_aiq_uapi_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_api_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi_sysctl_queryCpsLtCap

【描述】
查询补光灯的支持能力。

【语法】

```
XCamReturn
rk_aiq_uapi_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_api_sysctl.h
- 库文件：librkaiq.so

rk_aiq_uapi_sysctl_getBindedSnsEntNmByVd

【描述】
查询video结点所对应的sensor entity name。

【语法】

```
const char* rk_aiq_uapi_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
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_api_sysctl.h
- 库文件：librkaiq.so

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

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 模块有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_uapi_setExpMode

【描述】
设置曝光模式。

【语法】

```
XCamReturn rk_aiq_uapi_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

【需求】

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

rk_aiq_uapi_getExpMode

【描述】
获取曝光模式。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setAeMode

【描述】
设置AE工作模式。

【语法】

```
XCamReturn rk_aiq_uapi_setAeMode(const rk_aiq_sys_ctx_t* ctx, aeMode_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	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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getAeMode

【描述】
获取AE工作模式。

【语法】

```
XCamReturn rk_aiq_uapi_getAeMode(const rk_aiq_sys_ctx_t* ctx, aeMode_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	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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setExpGainRange

【描述】
设置增益范围。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getExpGainRange

【描述】
获取增益范围。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setExpTimeRange

【描述】
设置曝光时间范围。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getExpTimeRange

【描述】
获取曝光时间范围。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setBLCMode

【描述】
背光补偿开关、区域设置。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setBLCStrength

【描述】
设置暗区提升强度。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setHLCMode

【描述】
强光抑制开关。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setHLCStrength

【描述】
设置强光抑制强度。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setAntiFlickerMode

【描述】
设置抗闪模式。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getAntiFlickerMode

【描述】
获取抗闪模式。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setExpPwrLineFreqMode

【描述】
设置抗闪频率。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getExpPwrLineFreqMode

【描述】
获取抗闪频率。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_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

aeMode_t

【说明】
定义AE工作模式

【定义】

```
typedef enum aeMode_e {
    AE_AUTO = 0,
    AE_IRIS_PRIOR = 1,
    AE_SHUTTER_PRIOR = 2
} aeMode_t;
```

【成员】

成员名称	描述
OP_AUTO	自动选择
AE_IRIS_PRIOR	光圈优先
AE_SHUTTER_PRIOR	快门优先

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_api_ae_setExpSwAttr

【描述】

设定 AE曝光软件属性。

【语法】

```
XCamReturn
rk_aiq_user_api_ae_setExpSwAttr(const rk_aiq_sys_ctx_t* ctx,
                                const Uapi_ExpSwAttr_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曝光软件属性结构体	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_api_ae.h、rk_aiq_uapi_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_ae_getExpSwAttr

【描述】

获取 AE 曝光软件属性。

【语法】

```
XCamReturn
rk_aiq_user_api_ae_getExpSwAttr(const rk_aiq_sys_ctx_t* ctx,
                        Uapi_ExpSwAttr_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_api_ae.h、rk_aiq_uapi_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_ae_setLinAeRouteAttr

【描述】
设置线性模式下的AE曝光分配策略。

【语法】

```
XCamReturn
rk_aiq_user_api_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_api_ae.h、rk_aiq_uapi_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_ae_getLinAeRouteAttr

【描述】
获取线性模式下的AE曝光分配策略。

【语法】

```
XCamReturn
rk_aiq_user_api_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_api_ae.h、rk_aiq_uapi_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_ae_setLinAeRouteAttr

【描述】
设置HDR模式下的AE曝光分配策略。

【语法】

```
XCamReturn
rk_aiq_user_api_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_api_ae.h、rk_aiq_uapi_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_ae_getHdrAeRouteAttr

【描述】
获取HDR模式下的AE曝光分配策略。

【语法】

```
XCamReturn
rk_aiq_user_api_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_api_ae.h、rk_aiq_uapi_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_ae_queryExpResInfo

【描述】
获取 AE 内部状态信息。

【语法】

```
XCamReturn
rk_aiq_user_api_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曝光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_api_ae.h、rk_aiq_uapi_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_ae_setLinExpAttr

【描述】
设置AE线性模式曝光参数。

【语法】

```
XCamReturn
rk_aiq_user_api_ae_setLinExpAttr(const rk_aiq_sys_ctx_t* ctx, const
Uapi_LinExpAttr_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_api_ae.h、rk_aiq_uapi_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_ae_getLinExpAttr

【描述】
获取AE线性模式曝光参数。

【语法】

```
XCamReturn
rk_aiq_user_api_ae_getLinExpAttr(const rk_aiq_sys_ctx_t* ctx, Uapi_LinExpAttr_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_api_ae.h、rk_aiq_uapi_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_ae_setHdrExpAttr

【描述】
设置AE HDR模式曝光参数。

【语法】

```
XCamReturn
rk_aiq_user_api_ae_setHdrExpAttr(const rk_aiq_sys_ctx_t* ctx, const
Uapi_HdrExpAttr_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_api_ae.h、rk_aiq_uapi_ae_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_ae_getHdrExpAttr

【描述】
获取AE HDR模式曝光参数。

【语法】

```
XCamReturn
rk_aiq_user_api_ae_getHdrExpAttr(const rk_aiq_sys_ctx_t* ctx, Uapi_HdrExpAttr_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_api_ae.h、rk_aiq_uapi_ae_int.h
- 库文件：librkaiq.so

模块级API数据类型

CalibDb_AecDayNightMode_t

【说明】
定义白天、夜晚模式

【定义】

```
typedef enum _CalibDb_AecDayNightMode_e {
    AEC_DNMODE_MIN = -1,
    AEC_DNMODE_DAY = 0,
    AEC_DNMODE_NIGHT = 1,
    AEC_DNMODE_MAX = 2,
} CalibDb_AecDayNightMode_t;
```

【成员】

成员名称	描述
AEC_DNMODE_DAY	白天模式
AEC_DNMODE_NIGHT	夜晚模式

CalibDb_FlickerFreq_t

【说明】
定义抗闪频率

【定义】

```
typedef enum _CalibDb_FlickerFreq_e {
    AEC_FLICKER_FREQUENCY_OFF   = 0,
    AEC_FLICKER_FREQUENCY_50HZ = 1,
    AEC_FLICKER_FREQUENCY_60HZ = 2,
} CalibDb_FlickerFreq_t;
```

【成员】

成员名称	描述
AEC_FLICKER_FREQUENCY_OFF	不设定频率，采用自动模式
AEC_FLICKER_FREQUENCY_50HZ	50赫兹
AEC_FLICKER_FREQUENCY_60HZ	60赫兹

CalibDb_AntiFlickerMode_t

【说明】
定义抗闪模式

【定义】

```
typedef enum _CalibDb_AntiFlickerMode_e {
    AEC_ANTIFLICKER_NORMAL_MODE = 0,
    AEC_ANTIFLICKER_AUTO_MODE = 1,
} CalibDb_AntiFlickerMode_t;
```

【成员】

成员名称	描述
AEC_ANTIFLICKER_NORMAL_MODE	普通抗闪模式
AEC_ANTIFLICKER_AUTO_MODE	自动抗闪模式

【注意事项】

- AEC_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。

- AEC_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。

CalibDb_AntiFlickerAttr_t

【说明】
定义抗闪属性

【定义】

```
typedef struct CalibDb_AntiFlickerAttr_s {
    bool                        enable;
    CalibDb_FlickerFreq_t       Frequency;
    CalibDb_AntiFlickerMode_t   Mode;
} CalibDb_AntiFlickerAttr_t;
```

【成员】

成员名称	描述
enable	使能状态
Frequency	抗闪频率
Mode	抗闪模式

CalibDb_AeSpeed_t

【说明】
定义AE条件速度属性

【定义】

```
typedef struct CalibDb_AeSpeed_s {
    float                   DampOverStill;
    float                   DampUnderStill;
    float                   DampDark2BrightStill;
    float                   DampBright2DarkStill;
    float                   DampOverVideo;
    float                   DampUnderVideo;
} CalibDb_AeSpeed_t;
```

【成员】

成员名称	描述
DampOverStill	图像亮度高于目标值的曝光调节速度，取值范围[0, 1]
DampUnderStill	图像亮度低于目标值的曝光调节速度，取值范围[0, 1]
DampDark2BrightStill	场景从暗到亮的曝光调节速度，取值范围[0, 1]
DampBright2DarkStill	场景从亮到暗的曝光调节速度，取值范围[0, 1]
DampOverVideo	图像亮度高于目标值的曝光调节速度，取值范围[0, 1]
DampUnderVideo	图像亮度低于目标值的曝光调节速度，取值范围[0, 1]

CalibDb_AeRange_t

【说明】
定义AE参数范围

【定义】

```
typedef struct CalibDb_AeRange_s {
    float                   Min;
    float                   Max;
} CalibDb_AeRange_t;
```

【成员】

成员名称	描述
Min	下限值
Max	上限值

CalibDb_LinAeRange_t

【说明】
定义AE线性模式的参数范围

【定义】

```
typedef struct CalibDb_LinAeRange_s {
    CalibDb_AeRange_t       stExpTimeRange;
    CalibDb_AeRange_t       stGainRange;
    CalibDb_AeRange_t       stIspDGainRange;
    CalibDb_AeRange_t       stPIrisRange;
} CalibDb_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	光圈大小范围，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/最小值以第一次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第二次校正，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/最小值以第一次校正的曝光分解路线节点最大值和最小值为准。

CalibDb_HdrAeRange_t

【说明】
定义AE HDR模式的参数范围

【定义】

```
typedef struct CalibDb_HdrAeRange_s {
    CalibDb_AeRange_t      stExpTimeRange[3];
    CalibDb_AeRange_t      stGainRange[3];
    CalibDb_AeRange_t      stIspDGainRange[3];
    CalibDb_AeRange_t      stPIrisRange;
} CalibDb_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	光圈值范围，设置最大值和最小值

【注意事项】

- stExpTimeRange[3] 预定义3个元素，表示最多支持长、中、短 3帧HDR，实际使用的元素个数以senosr的支持情况为准。
- 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/最小值以第一次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第二次校正，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/最小值以第一次校正的曝光分解路线节点最大值和最小值为准。

CalibDb_AeFrmRateAttr_t

【说明】
定义AE 帧率属性

【定义】

```
typedef struct CalibDb_AeFrmRateAttr_s {
    bool             isFpsFix;
    uint8_t          FpsValue;
} CalibDb_AeFrmRateAttr_t;
```

【成员】

成员名称	描述
isFpsFix	自动曝光帧率固定使能，默认值为FALSE,即自动降帧模式；值为TRUE时，表示固定帧率
FpsValue	仅在固定帧率时有效，默认值为0时，固定使用驱动帧率；值不会0时，使用设定的帧率值

CalibDb_AeAttr_t

【说明】
定义AE 属性

【定义】

```
typedef struct CalibDb_AeAttr_s {
    CalibDb_AeSpeed_t        stAeSpeed;
    uint8_t                  BlackDelayFrame;
    uint8_t                  WhiteDelayFrame;
    bool                     SetAeRangeEn;
    CalibDb_LinAeRange_t     stLinAeRange;
    CalibDb_HdrAeRange_t     stHdrAeRange;
    CalibDb_AeFrmRateAttr_t  stFrmRate;
} CalibDb_AeAttr_t;
typedef CalibDb_AeAttr_t  Uapi_AeAttr_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
SetAeRangeEn	是否设置AE参数范围
stLinAeRange	线性模式自动曝光量范围
stHdrAeRange	Hdr模式自动曝光量范围
stFrmRate	自动曝光帧率模式，固定帧率模式或自动降帧模式

AecExpSeparateName_t

【说明】
定义Name字符串类型

【定义】

```
#define AEC_EXP_SEPARATE_NAME       ( 20U )
typedef  char  AecExpSeparateName_t[AEC_EXP_SEPARATE_NAME];
```

CalibDb_LinAeRoute_Attr_t

【说明】
定义AE线性策略属性

【定义】

```
typedef struct CalibDb_LinAeRoute_Attr_s {
    AecExpSeparateName_t     name;
    float                    TimeDot[AEC_ROUTE_MAX_NODES];
    float                    GainDot[AEC_ROUTE_MAX_NODES];
    float                    IspgainDot[AEC_ROUTE_MAX_NODES];
    float                    PIrisDot[AEC_ROUTE_MAX_NODES];
    int                      array_size;
} CalibDb_LinAeRoute_Attr_t;
typedef CalibDb_LinAeRoute_Attr_t   Uapi_LinAeRouteAttr_t;
```

【成员】

成员名称	描述
name	模式名称，分为normal模式和IR模式
TimeDot	曝光时间节点，单位为s
GainDot	增益节点
IspgainDot	Isp数字增益节点
PIrisDot	光圈节点
array_size	曝光分解节点个数

【注意事项】

- 曝光分解节点个数默认为6，建议至少设置6个节点。
- 节点的曝光量是曝光时间、sensor模拟增益、sensor数字增益、ISP数字增益、光圈的乘积。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。
- 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。
- 设置的曝光分解路线节点不是最终生效的曝光分解路线。
- 如果相邻节点的曝光量增加，则应该有一个曝光分量增加，其他曝光分量固定。

【相关定义】

- AEC_ROUTE_MAX_NODES
- AecExpSeparateName_t

CalibDb_HdrAeRoute_Attr_t

【说明】
定义AE HDR策略属性

【定义】

```
typedef struct CalibDb_HdrAeRoute_Attr_s {
    AecExpSeparateName_t     name;
    float                    HdrTimeDot[3][AEC_ROUTE_MAX_NODES];
    float                    HdrGainDot[3][AEC_ROUTE_MAX_NODES];
    float                    HdrIspDGainDot[3][AEC_ROUTE_MAX_NODES];
    float                    PIrisDot[AEC_ROUTE_MAX_NODES];
    int                      array_size;
} CalibDb_HdrAeRoute_Attr_t;
typedef CalibDb_HdrAeRoute_Attr_t Uapi_HdrAeRouteAttr_t;
```

【成员】

成员名称	描述
name	模式名称，分为normal模式和IR模式
HdrTimeDot	曝光时间节点，单位为s，数组0/1/2分别为短帧、中帧、长帧
HdrGainDot	增益节点，数组0/1/2分别为短帧、中帧、长帧
HdrIspDGainDot	Isp数字增益节点，数组0/1/2分别为短帧、中帧、长帧
PIrisDot	光圈节点，数组0/1/2分别为短帧、中帧、长帧
array_size	曝光分解节点个数，数组0/1/2分别为短帧、中帧、长帧

【注意事项】

- 曝光分解节点个数默认为6，建议至少设置6个节点
- 节点的曝光量是曝光时间、sensor模拟增益、sensor数字增益、ISP数字增益、光圈的乘积。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。
- 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。
- 设置的曝光分解路线节点不是最终生效的曝光分解路线。
- 如果相邻节点的曝光量增加，则应该有一个曝光分量增加，其他曝光分量固定。

【相关定义】

- AEC_ROUTE_MAX_NODES
- AecExpSeparateName_t

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];
    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
HdrLumaDeviation	Hdr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
MeanLuma	线性模式下，平均亮度
HdrMeanLuma	HDR模式下平均亮度
CurExpInfo	当前曝光信息
Piris	光圈
LinePeriodsPerField	VTS
PixelPeriodsPerLine	HTS
PixelClockFreqMHZ	像素时钟频率(赫兹)

CalibDb_AecDynamicSetpoint_t

【说明】
定义AE动态目标值

【定义】

```
typedef struct CalibDb_AecDynamicSetpoint_s {
    AecDynamicSetpointName_t      name;
    float ExpValue[AEC_SETPOINT_MAX_NODES];
    float DySetpoint[AEC_SETPOINT_MAX_NODES];
    int   array_size;
} CalibDb_AecDynamicSetpoint_t;
```

【成员】

成员名称	描述
name	模式名称，分为normal模式和IR模式
ExpValue	动态曝光量节点属性，节点值为当前曝光量与最大曝光量的比值，取值范围为[0,1]
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
array_size	动态目标亮度值的节点数

【相关定义】

- AecDynamicSetpointName_t
- AEC_SETPOINT_MAX_NODES

Uapi_LinExpAttr_t

【说明】

定义AE线性曝光参数

【定义】

```
typedef struct Uapi_LinExpAttr_s {
    float                   SetPoint;
    float                   NightSetPoint;
    float                   EvBias;
    float                   Tolerance;
    int                     StrategyMode;
    bool                    DySetPointEn;
    Uapi_AeDySetpoint_t     DySetpoint[AEC_DNMODE_MAX];
} Uapi_LinExpAttr_t;
```

【成员】

成员名称	描述
SetPoint	normal模式下，自动曝光调节时的目标亮度值，取值范围[0,255]
NightSetPoint	IR/夜间模式下，自动曝光调节时的目标亮度值，取值范围[0,255]
EvBias	自动曝光调节时，曝光量的偏差百分比，单位为%，取值范围为[-200,+200]
Tolerance	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
DySetPointEn	自动曝光调节的动态目标亮度值使能开关。动态目标亮度值的使能，enable=TRUE时，采用动态目标亮度值，enable=FALSE时，使用固定目标亮度值，动态目标亮度值失效
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，分为normal模式和夜间/IR模式

【注意事项】

- SetPoint 代表normal模式下的目标亮度值，即未开启夜间模式或IR模式时使用的目标亮度值。NightSetPoint 代表夜间模式或IR模式下的目标亮度值。夜间模式与IR模式不可同时开启，IR模式的开启需要硬件上的支持。
- DySetPointEn = TRUE时，固定目标亮度值SetPoint、 NightSetPoint无效，使用动态目标亮度值；DySetPointEn = FALSE时，动态目标亮度值无效，所有场景始终使用同一目标亮度。
- 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/IRSetPoint）*（1+EvBias/100）
- 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*（1-Tolerance/100），真实生效目标亮度*（1+Tolerance/100）] 范围内。

【相关定义】

- Uapi_AeDySetpoint_t

Uapi_HdrExpAttr_t

【说明】
定义AE HDR曝光参数

【定义】

```
typedef struct Uapi_HdrExpAttr_s {
    float                   Tolerance;
    int                     StrategyMode;
    float                   Evbias;
    int                     ExpRatioType;
    Cam1x6FloatMatrix_t     RatioExpDot;
    Cam1x6FloatMatrix_t     M2SRatioFix;
    Cam1x6FloatMatrix_t     L2MRatioFix;
    Cam1x6FloatMatrix_t     M2SRatioMax;
    Cam1x6FloatMatrix_t     L2MRatioMax;
} Uapi_HdrExpAttr_t;
```

【成员】

成员名称	描述
Tolerance	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
Evbias	自动曝光调节时，曝光量的偏差百分比，单位为%，取值范围为[-200,+200]
ExpRatioType	曝光比模式，仅在Hdr模式多帧合成下有效，AUTO：根据场景，自动计算长短帧的曝光比；FIX：长短帧采用固定曝光比
RatioExpDot	表示曝光量节点，根据曝光量，动态设置曝光比固定值或曝光比最大值，二者一一对应。节点个数固定为6
M2SRatioFix	节点个数固定为6。ExpRatioType为AUTO时，无效。ExpRatioType为FIX时，表示中帧与短帧的曝光比，与曝光量节点RatioExpDot一一对应
L2MRatioFix	节点个数固定为6。ExpRatioType为AUTO时，无效。ExpRatioType为FIX时，表示长帧与中帧的曝光比,与曝光量节点RatioExpDot一一对应。Hdr为2帧合成时无效，3帧合成时有效
M2SRatioMax	节点个数固定为6。ExpRatioType为AUTO时，表示中帧与短帧的曝光比动态最大值，与曝光量节点RatioExpDot一一对应。ExpRatioType为FIX时，无效
L2MRatioMax	节点个数固定为6。ExpRatioType为AUTO时，表示长帧与中帧的曝光比动态最大值,与曝光量节点RatioExpDot一一对应。Hdr为2帧合成时无效，3帧合成时有效。ExpRatioType为FIX时，无效

【相关定义】

- Cam1x6FloatMatrix_t

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_uapi_setWBMode

【描述】
设置白平衡模式。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getWBMode

【描述】
获取白平衡模式。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_lockAWB

【描述】
锁定当前白平衡参数。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_unlockAWB

【描述】
解锁已被锁定的白平衡参数。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setMWBScene

【描述】
设置白平衡场景。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getMWBScene

【描述】
获取白平衡场景。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setMWBGain

【描述】
设置白平衡增益系数。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getMWBGain

【描述】
获取白平衡增益系数。

【语法】

```
XCamReturn rk_aiq_uapi_getMWBGain(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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setMWBCT

【描述】
设置白平衡色温参数。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getMWBCT

【描述】
获取白平衡增益系数。

【语法】

```
XCamReturn rk_aiq_uapi_getMWBCT(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_api_imgproc.h
- 库文件：librkaiq.so

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	手动模式
RK_AIQ_WB_MODE_AUTO	自动模式

rk_aiq_wb_scene_t

参见前述。

rk_aiq_wb_gain_t

参见前述。

rk_aiq_wb_cct_t

参见前述。

模块级API参考

rk_aiq_user_api_awb_SetAttrib

【描述】
获取白平衡属性。

【语法】

```
XCamReturn
rk_aiq_user_api_awb_SetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_wb_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_api_awb.h、rk_aiq_uapi_awb_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_awb_GetAttrib

【描述】
获取白平衡属性。

【语法】

```
XCamReturn
rk_aiq_user_api_awb_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
rk_aiq_wb_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_api_awb.h、rk_aiq_uapi_awb_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_awb_GetCCT

【描述】
获取白平衡色温参数。

【语法】

```
XCamReturn
rk_aiq_user_api_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_api_awb.h、rk_aiq_uapi_awb_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_awb_QueryWBInfo

【描述】
获取白平衡增益系数，检测色温。

【语法】

```
XCamReturn
rk_aiq_user_api_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_api_awb.h、rk_aiq_uapi_awb_int.h
- 库文件：librkaiq.so

模块级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_wb_awb_attrib_t

【说明】
定义自动白平衡属性

【定义】

```
typedef struct rk_aiq_wb_awb_attrib_s {
    rk_aiq_wb_awb_alg_method_t algMethod;
    float tolerance;
    unsigned int runInterval;
    bool sceneAdjustEn;
    bool colorBalanceEn;
    bool cagaEn;
    bool wbGainAdjustEn;
    bool wbGainDaylightClipEn;
    bool wbGainClipEn;
} rk_aiq_wb_awb_attrib_t;
```

【成员】

成员名称	描述
algMethod	白平衡策略选择
tolerance	容忍度
runInterval	运行帧间隔
sceneAdjustEn	场景调节
colorBalanceEn	色调调整使能
cagaEn	白平衡校正后的图像尽可能与人眼感知的色彩外貌一致的使能
wbGainAdjustEn	色调调整使能
wbGainDaylightClipEn	室外最低色温限制使能
wbGainClipEn	色温范围限制使能

rk_aiq_wb_attrib_t

【说明】
定义白平衡属性

【定义】

```
typedef struct rk_aiq_wb_attrib_s {
    bool byPass;
    rk_aiq_wb_op_mode_t mode;
    rk_aiq_wb_mwb_attrib_t stManual;
    rk_aiq_wb_awb_attrib_t stAuto;
} rk_aiq_wb_attrib_t;
```

【成员】

成员名称	描述
byPass	跳过模块处理
mode	模式选择
stManual	手动模式下参数配置
stAuto	自动模式下参数配置

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;
} 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

IMGPROC

概述

imgproc 是指影响图像效果的模块。

FEC

功能描述

光学系统、电子扫描系统失真而引起的斜视畸变、枕形、桶形畸变等，都可能使图像产生几何特性失真。鱼眼图像的畸变矫正是以某种变换方式将鱼眼图像转换为理想图像的操作。

重要概念

- 畸变实际上指的是拍摄出来的物体相对于物体本身而言的失真。

功能级API参考

rk_aiq_user_api_afec_enable

【描述】 开启畸变校正。

【语法】

```
XCamReturn rk_aiq_user_api_afec_enable(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_api_afec.h、rk_aiq_uapi_afec_int.h
- 库文件：librkaiq.so

rk_aiq_user_api_afec_disable

【描述】 关闭畸变校正。

【语法】

```
XCamReturn rk_aiq_user_api_afec_disable(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_api_afec.h、rk_aiq_uapi_afec_int.h
- 库文件：librkaiq.so

HDR

功能描述

功能级API参考

rk_aiq_uapi_setHDRMode

【描述】 设置HDR工作模式。

【语法】

```
XCamReturn rk_aiq_uapi_setHDRMode(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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getHDRMode

【描述】 获取HDR工作模式。

【语法】

```
XCamReturn rk_aiq_uapi_getHDRMode(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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setMHDRStrth

【描述】 设置手动模式下的HDR强度。

【语法】

```
XCamReturn rk_aiq_uapi_setMHDRStrth(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	强度，取值范围[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

【需求】

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

rk_aiq_uapi_getMHDRStrth

【描述】 获取手动模式下的HDR强度。

【语法】

```
XCamReturn rk_aiq_uapi_getMHDRStrth(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	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_api_imgproc.h
- 库文件：librkaiq.so

Noise Removal

功能描述

图像去噪

功能级API参考

rk_aiq_uapi_setNRMode

【描述】 设置去噪模式。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getNRMode

【描述】 获取当前去噪模式。

【语法】

```
XCamReturn rk_aiq_uapi_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setANRStrth

【描述】 设置普通去噪强度。

【语法】

```
XCamReturn rk_aiq_uapi_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	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_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getANRStrth

【描述】 获取普通去噪强度。

【语法】

```
XCamReturn rk_aiq_uapi_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	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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setMSpaNRStrth

【描述】 设置空域去噪强度。

【语法】

```
XCamReturn rk_aiq_uapi_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	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_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getMSpaNRStrth

【描述】 获取空域去噪强度。

【语法】

```
XCamReturn rk_aiq_uapi_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	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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setMTNRStrth

【描述】 设置时域去噪强度。

【语法】

```
XCamReturn rk_aiq_uapi_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	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_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getMTNRStrth

【描述】 获取时域去噪强度。

【语法】

```
XCamReturn rk_aiq_uapi_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	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_api_imgproc.h
- 库文件：librkaiq.so

Defog

功能描述

Defog 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。

功能级API参考

rk_aiq_uapi_setDhzMode

【描述】

设置去雾工作模式。

【语法】

```
XCamReturn rk_aiq_uapi_setDhzMode(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	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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getDhzMode

【描述】

获取当前去雾工作模式。

【语法】

```
XCamReturn rk_aiq_uapi_getDhzMode(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	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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setMDhzStrth

【描述】

设置去雾工作强度。

【语法】

```
XCamReturn rk_aiq_uapi_setMDhzStrth(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	强度	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_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getMDhzStrth

【描述】

获取去雾工作强度。

【语法】

```
XCamReturn rk_aiq_uapi_getMDhzStrth(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	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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_enableDhz

【描述】

开启去雾功能。

【语法】

```
XCamReturn rk_aiq_uapi_enableDhz(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

【注意】

- 开启去雾功能后，对比度设置自动失效。

【需求】

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

rk_aiq_uapi_disableDhz

【描述】

关闭去雾功能。

【语法】

```
XCamReturn rk_aiq_uapi_disableDhz(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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setContrast

【描述】

设置对比度。

【语法】

```
XCamReturn rk_aiq_uapi_setContrast(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	对比度强度，取值范围[0,100]	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】

- 开启去雾功能后，对比度设置自动失效。

【需求】

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

ACM

功能描述

ACM(Auto Color Managment) 提供基本的喜好色调节功能，通过对一定区间内的亮度、饱和度的调节，达到对喜好色的调节。

API参考

rk_aiq_uapi_setBrightness

【描述】

设置亮度。

【语法】

```
XCamReturn rk_aiq_uapi_setBrightness(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	亮度值百分比，取值范围[0,100]	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_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getBrightness

【描述】

获取亮度。

【语法】

```
XCamReturn rk_aiq_uapi_getBrightness(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	亮度值百分比	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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setSaturation

【描述】

设置饱和度。

【语法】

```
XCamReturn rk_aiq_uapi_setSaturation(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	饱和度百分比	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_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getSaturation

【描述】

获取饱和度。

【语法】

```
XCamReturn rk_aiq_uapi_getSaturation(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	饱和度百分比	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_api_imgproc.h
- 库文件：librkaiq.so

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_uapi_setSharpness

【描述】

设置锐化等级。

【语法】

```
XCamReturn rk_aiq_uapi_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	锐化等级百分比	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_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getSharpness

【描述】

设置锐化等级。

【语法】

```
XCamReturn rk_aiq_uapi_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	锐化等级百分比	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_api_imgproc.h
- 库文件：librkaiq.so

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。

功能级API参考

rk_aiq_uapi_setGammaCoef

【描述】

设置伽玛。

【语法】

```
XCamReturn rk_aiq_uapi_setGammaCoef(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	伽玛百分比	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_imgproc.h
- 库文件：librkaiq.so

ASD

功能级API参考

rk_aiq_user_api_asd_GetAttrib

【描述】

获取当前环境亮度的计算结果。

【语法】

```
XCamReturn rk_aiq_user_api_asd_GetAttrib(const rk_aiq_sys_ctx_t* sys_ctx,
asd_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	计算结果	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_api_asd.h
- 库文件：librkaiq.so

数据类型

asd_attrib_t

【说明】

当前环境亮度的计算结果

【定义】

```
typedef struct asd_attrib_s {
    float cur_m2r;
} asd_attrib_t;
```

【成员】

成员名称	描述
cur_m2r	当前平均亮度，计算方法为 exp_val_ratio = cur_exp_val / max_exp_va， cur_m2r = mean_luma / exp_val_ratio

其他

API参考

rk_aiq_uapi_setGrayMode

【描述】

设置黑白图像模式的工作方式。

【语法】

```
XCamReturn rk_aiq_uapi_setGrayMode(const rk_aiq_sys_ctx_t* ctx,
rk_aiq_gray_mode_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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getGrayMode

【描述】

设置黑白图像模式的工作方式。

【语法】

```
rk_aiq_gray_mode_t rk_aiq_uapi_setGrayMode(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
rk_aiq_gray_mode_t	工作模式

【需求】

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

rk_aiq_uapi_setFrameRate

【描述】

设置图像输出帧率。

【语法】

```
XCamReturn rk_aiq_uapi_setFrameRate(const rk_aiq_sys_ctx_t* ctx, frameRateInfo_t
info);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
frameRateInfo_t	帧率信息结构体	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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getFrameRate

【描述】

获取图像输出帧率信息。

【语法】

```
XCamReturn rk_aiq_uapi_getFrameRate(const rk_aiq_sys_ctx_t* ctx,
frameRateInfo_t* info);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
frameRateInfo_t	帧率信息结构体	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_api_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_setMirroFlip

【描述】

设置图像镜像、翻转。

【语法】

```
XCamReturn rk_aiq_uapi_setMirroFlip(const rk_aiq_sys_ctx_t* ctx, bool mirror,
bool flip);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mirror	是否镜像	输入
flip	是否翻转	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_imgproc.h
- 库文件：librkaiq.so

rk_aiq_uapi_getMirroFlip

【描述】

获取图像镜像、翻转信息。

【语法】

```
XCamReturn rk_aiq_uapi_getMirrorFlip(const rk_aiq_sys_ctx_t* ctx, bool* mirror,
bool* flip);
```

【参数】

参数名称	描述	输入/输出
ctx	AIQ上下文指针	输入
mirror	是否镜像	输出
flip	是否翻转	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_api_imgproc.h
- 库文件：librkaiq.so

数据类型

rk_aiq_gray_mode_t

【说明】

黑白切换工作模式

【定义】

```
typedef enum rk_aiq_gray_mode_e {
    RK_AIQ_GRAY_MODE_CPSL,
    RK_AIQ_GRAY_MODE_OFF,
    RK_AIQ_GRAY_MODE_ON,
} rk_aiq_gray_mode_t;
```

【成员】

成员名称	描述
RK_AIQ_GRAY_MODE_CPSL	由cpsl算法控制
RK_AIQ_GRAY_MODE_OFF	关闭黑白模式
RK_AIQ_GRAY_MODE_ON	开启黑白模式

统计信息

概述

ISP提供的3A统计信息以及相关配置

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图的256段带权重直方图统计信息、基于raw图的分块R/G/B/Y 均值统计信息；基于gamma前RGB图的32段带权重直方图统计信息、基于gamma前RGB图的分块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：最大支持全局15X15分块，最小支持1X1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；在全局分块的基础上，支持独立设置4个子窗口，每个子窗口均可输出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子窗口的像素数求得。该模式下的带权重直方图统计，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分块，最小支持1X1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；不支持独立设置子窗口。该模式下的带权重直方图统计，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。

基于RGB图的AE统计

- 该模块统计分为分块亮度统计和直方图统计。

- 分块亮度统计，最大支持15X15分块，最小支持1x1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；在全局分块的基础上，支持独立设置4个子窗口，每个子窗口均可输出32bit Y通道亮度总和，亮度均值需要在软件中除以每个子窗口的像素数求得。
- 直方图统计，最大支持15X15分块，最新支持5X5分块，该模式下的带权重直方图统计，根据分块数和对应分配的权重，进行32段8bit亮度统计，每个亮度分段内像素数的有效bit数为16bit。

AWB统计信息

AWB硬件统计信息包含全局统计信息和区域统计信息。

全局统计信息：图像全局AWB统计窗口内分色温区域的R,G,B均值，以及有效统计点的个数，色温区域支持7个色温。

分块统计信息：图像全局AWB统计窗口内15x15分块，每个分块的R,G,B均值。

AF统计信息

AF硬件统计信息包含2个主窗口统计信息以及1个主窗口中分块统计信息。

主窗口统计信息：AF统计主窗口内AF统计信息。

分块统计信息：AF统计主窗口内15x15分块的统计信息。

API参考

rk_aiq_uapi_sysctl_get3AStats

【描述】
获取3A统计信息。

【语法】

```
XCamReturn
rk_aiq_uapi_sysctl_get3AStats(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_api_sysctl.h
- 库文件：librkaiq.so

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_awb_stat_res_v200_t awb_stats_v200;
    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
awb_stats_v200	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
RkAiqAecHwStatsRes_t	AE模块硬件统计信息
RKAiqAecExpInfo_t	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];
    unsigned short line_length_pixels;
    unsigned short frame_length_lines;
    float pixel_clock_freq_mhz;
} RKAiqAecExpInfo_t;
```

【成员】

成员名称	描述
LinearExp	非HDR模式的曝光参数信息
HdrExp	HDR模式的曝光参数信息
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

【定义】

```
typedef struct {
    RkAiqExpRealParam_t exp_real_params; //real value
    RkAiqExpSensorParam_t exp_sensor_params;//reg value
} RkAiqExpParamComb_t;
```

```
typedef struct RkAiqExpRealParam_s {
    float integration_time;
    float analog_gain;
    float digital_gain;
    float isp_dgain;
    int   iso;
    int   dcg_mode;
} RkAiqExpRealParam_t;
```

```
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】

成员名称	描述
integration_time	曝光积分时间，单位为秒
analog_gain	sensor的模拟增益/Total增益
digital_gain	sensor的数字增益，暂时无效。数字增益大小合并到analog_gain
isp_dgain	isp的数字增益，暂时无效
iso	感光度，暂时无效
dcg_mode	dual conversion gain模式
fine_integration_time	fine曝光积分时间寄存器值，暂时无效
coarse_integration_time	曝光积分时间寄存器值【行数】
analog_gain_code_global	sensor模拟增益寄存器值
digital_gain_global	sensor数字增益寄存器值，暂时无效
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

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
Aec_Stat_Res_t	AE模块基于raw图的统计信息，兼容HDR与非HDR模式，至多支持HDR 3TO1 S/M/L的统计信息。
yuvae_stat	AE模块基于gamma前RGB图的分块信息
sihist_stat	AE模块基于gamma前RGB图的直方图信息

【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合成后经debayer的raw图统计信息。该统计模块之前有BLC、 AWB、 HDRMERGE、 TMO模块，因此该模块的统计信息受BLC、 AWB、 HDRMERGE、TMO的增益影响。

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
rawaebig_stat	基于raw图的big模式分块统计信息
rawaelite_stat	基于raw图的lite模式分块统计信息
rawhist_stat	基于raw图的直方图统计信息

【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。由于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子窗口均值亮度统计，故此处基于raw图的big、lite模式直方图统计具有相同的数据结构。

rawaebig_stat

【说明】
基于raw图的big模式统计信息，包含全局窗口分块R/G/B均值亮度、子窗口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模式全局窗口分块的r通道均值亮度信息。有效比特数：10bit。
channelg_xy	big模式全局窗口分块的g通道均值亮度信息。有效比特数：12bit。
channelb_xy	big模式全局窗口分块的b通道均值亮度信息。有效比特数：10bit。
wndx_sumr	big模式子窗口的r通道亮度和信息。有效比特数：29bit。
wndx_sumg	big模式子窗口的g通道亮度和信息。有效比特数：32bit。
wndx_sumb	big模式子窗口的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模式全局窗口分块统计信息为做了除法的均值亮度统计信息，但子窗口为整个窗口的亮度和信息，需要在软件添加代码计算子窗口的均值亮度统计信息。
- 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，包含全局窗口分块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	big模式全局窗口分块的r通道均值亮度信息。有效比特数：10bit。
channelg_xy	big模式全局窗口分块的g通道均值亮度信息。有效比特数：12bit。
channelb_xy	big模式全局窗口分块的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。
- 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，有效bit数：28bit

yuvae_stat

【说明】
基于gamma前RGB图的分块均值亮度统计信息，包含全局窗口分块Y通道均值亮度、子窗口Y通道亮度总和

【定义】

```
struct yuvae_stat {
    unsigned long int ro_yuvae_sumy[YUVAE_SUBWIN_NUM];
    unsigned char mean[YUVAE_WIN_NUM];
};
#define YUVAE_SUBWIN_NUM 4
#define YUVAE_WIN_NUM 225
```

【成员】

成员名称	描述
ro_yuvae_sumy	子窗口的Y通道亮度总和，有效bit数：32bit
mean	全局窗口分块Y通道均值亮度，有效bit数：8bit

【注意事项】

- 基于raw图的lite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。
- 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。

sihist_stat

【说明】
基于gamma前RGB图的直方图计信息

【定义】

```
struct sihist_stat {
    unsigned int bins[SIHIST_BIN_N_MAX];
};
#define SIHIST_BIN_N_MAX  32
```

【成员】

成员名称	描述
bins	直方图的分段，共32段，有效比特数：16bit

rk_aiq_awb_stat_res_v200_t

【说明】
定义白平衡硬件统计信息

【定义】

```
typedef struct rk_aiq_awb_stat_res_v200_s {
    rk_aiq_awb_stat_wp_res_light_v200_t light[RK_AIQ_AWB_MAX_WHITEREGIONS_NUM];
    rk_aiq_awb_stat_blk_res_v200_t blockResult[RK_AIQ_AWB_GRID_NUM_TOTAL];
    rk_aiq_awb_stat_wp_res_light_v200_t
 multiwindowLightResult[RK_AIQ_AWB_MAX_WHITEREGIONS_NUM];
    rk_aiq_awb_stat_wp_res_v200_t
excWpRangeResult[RK_AIQ_AWB_STAT_WP_RANGE_NUM_V200];
} rk_aiq_awb_stat_res_v200_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；
blockResult	每个块的RGB累加，图像进行不重叠同尺寸的15x15（RK_AIQ_AWB_GRID_NUM_TOTAL）分块
multiwindowLightResult	几个子窗口内不同光源下的白点统计结果，最多RK_AIQ_AWB_MAX_WHITEREGIONS_NUM个光源；
excWpRangeResult	落在非白点区域里的非白点统计结果，最多RK_AIQ_AWB_STAT_WP_RANGE_NUM_V200个非白点区域

【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_v200_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_v200_s {
    rk_aiq_awb_stat_wp_res_v200_t xYType[RK_AIQ_AWB_XY_TYPE_MAX_V200];
} rk_aiq_awb_stat_wp_res_light_v200_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_V200个框

rk_aiq_awb_stat_wp_res_v200_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_v200_s {
    unsigned int WpNo;
    unsigned int Rvalue;
    unsigned int Gvalue;
    unsigned int Bvalue;
} rk_aiq_awb_stat_wp_res_v200_t;
```

【成员】

成员名称	描述
WpNo	（非）白点数量
Rvalue	（非）白点R通道的累加和
Gvalue	（非）白点G通道的累加和
Bvalue	（非）白点B通道的累加和

rk_aiq_awb_stat_blk_res_v200_t

【说明】

定义每个块的统计结果

【定义】

```
typedef struct rk_aiq_awb_stat_blk_res_v200_s {
    unsigned int Rvalue;
    unsigned int Gvalue;
    unsigned int Bvalue;
    bool isWP[RK_AIQ_AWB_STORE_LS_WPFLAG_NUM];
} rk_aiq_awb_stat_blk_res_v200_t;
```

【成员】

成员名称	描述
isWP	块内是否包含某个光源白点的标志，最多纪录RK_AIQ_AWB_STORE_LS_WPFLAG_NUM个光源的标志
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_t

【说明】
定义AF统计信息

【定义】

```
typedef struct {
    unsigned int roia_sharpness;
    unsigned int roia_luminance;
    unsigned int roib_sharpness;
    unsigned int roib_luminance;
    unsigned int global_sharpness[RKAIQ_RAWAF_SUMDATA_NUM];
    struct timeval focus_starttim;
    struct timeval focus_endtim;
    int64_t sof_tim;
} rk_aiq_af_algo_stat_t;
```

【成员】

成员名称	描述
roia_sharpness	主窗口的清晰度值;
roia_luminance	主窗口的亮度值;
roib_sharpness	独立窗口的清晰度值;
roib_luminance	独立窗口的亮度值;
global_sharpness	主窗口下15*15子窗口的清晰度值;
focus_starttim	最近一次VCM移动的起始时间;
focus_endtim	最近一次VCM移动的结束时间;
sof_tim	本次数据帧的帧开始时间;

【注意事项】

roia_sharpness/roia_luminance/roib_sharpness/roib_luminance/global_sharpness为AF硬件统计信息。
focus_starttim/focus_endtim/sof_tim为VCM移动时间和数据帧的帧开始时间，辅助确认VCM是否移动结束，AF硬件统计信息是否可靠。

Debug

版本获取

1. 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；
2. 默认打印级别下，加载运行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. 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 ************************
```

log开关

1. 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
```

2. 模块log级别配置：

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。

例如开启af模块的log开关，并且级别为verbose，则bit[14] = 1, bit[3-0] = 4， 所以在应用程序执行前执行：

```
export persist_camera_engine_log=0x4014
```

查看当前log级别可通过如下命令：

```
[root@RV1126_RV1109:/]# echo $persist_camera_engine_log
0x4014
```

动态抓取raw/yuv图像

抓取raw图原理说明

目前软件isp的数据流粗略流程为： sensor(raw) -> csi-tx -> isp-rx -> ... -> isp-> ... ->ispp -> ... -> out-yuv, 其中csi-tx -> isp-rx的raw图数据可以在aiq的hwi层获取到。aiq根据/tmp/.capture_cnt中间文件获取用户想保存raw文件的帧数，aiq将对应帧数的raw图写入/tmp目录下。

抓raw图步骤

1. 运行应用程序，如运行rkisp_demo，其他应用程序也支持

```
rkisp_demo --device /dev/video14 --width 1280 --height 720 --vop --rkaiq --hdr
```

2. echo要抓取的raw图帧数, 例如抓取3帧

```
echo 3 > /tmp/.capture_cnt
```

3. 在/tmp目录下会生成抓取的raw图及对应的meta信息

```
[root@RV1126_RV1109:/]# ls -l /tmp/raw_2017-08-15_20-40-58/
total 35932
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame476_2688x1520_long.raw
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame476_2688x1520_short.raw
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame477_2688x1520_long.raw
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame477_2688x1520_short.raw
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame478_2688x1520_long.raw
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame478_2688x1520_short.raw
-rw-r--r-- 1 root root     381 Aug 15 20:40 meta_data
```

运行rkisp_demo,抓raw及对应的yuv图像步骤

1. 加--sync-to-raw参数，运行rkisp_demo，只有rkisp_demo支持

```
rkisp_demo --device /dev/video14 --width 1280 --height 720 --vop --rkaiq --hdr --sync-to-raw
```

2. echo要抓取的raw/yuv图帧数, 例如抓取3帧

```
echo 3 > /tmp/.capture_cnt
```

3. 在/tmp目录下会生成抓取的raw图/meta信息/yuv图像

```
[root@RV1126_RV1109:/]# ls -l /tmp/raw_2017-08-15_20-40-58/
total 35932
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame476_2688x1520_long.raw
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame476_2688x1520_short.raw
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame477_2688x1520_long.raw
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame477_2688x1520_short.raw
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame478_2688x1520_long.raw
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame478_2688x1520_short.raw
-rw-r--r-- 1 root root     381 Aug 15 20:40 meta_data

[root@RV1126_RV1109:/]# ls -l /tmp/yuv_2017-08-15_20-40-58/
total 17964
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame476.yuv
-rw-r--r-- 1 root root 6128640 Aug 15 20:40 frame477.yuv
-rw-r--r-- 1 root root 6128640 Aug 15 20:41 frame478.yuv
[root@RV1126_RV1109:/]#
```

4. 如上所示，raw图/meta信息/yuv图像是一一对应

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