DC100 ISP Development Guide

Release Version：V1.0.0

Release Date：20xx－xx－xx

Security Level： □Top-Secret □Secret □Internal ■Public

Overview

This article aims to describe the role of the DcAiq (Dc Auto Image Quality) module, the overall workflow, and related API interfaces. Mainly to Engineers who use RkAiq module for ISP function development provide help.

Product Version

Chipset	Kernel Version
RV1126	Linux 4.19

Intended Audience

This document (this guide) is mainly applicable to the following engineers:

- ISP module software development engineer
- System Integration Software Development Engineer

Revision History

Version	Author	Date	Revision History
V1.0.0	LEE	20xx-xx-xx	Initial version

Contents

DC100 ISP Development Guide

1. Overview

1.1 Overview

ISP contains a series of image processing algorithm modules, mainly including: dark current correction, dead pixel correction, 3A, HDR, lens shading correction, lens distortion correction, 3DLUT, denoising (including RAW domain denoising, multi-frame denoising, Color denoising, etc.), sharpening, etc.

ISP includes hardware algorithm realization and software logic control part, RkAiq is the realization of software logic control part.

The main functions of the RkAiq software module are: obtaining image statistics from the ISP driver, combining with IQ Tuning parameters, using a series of algorithms to calculate new ISP, Sensor and other hardware parameters, and continuously iterating the process to finally achieve the optimal image effect.

1.2 Function description

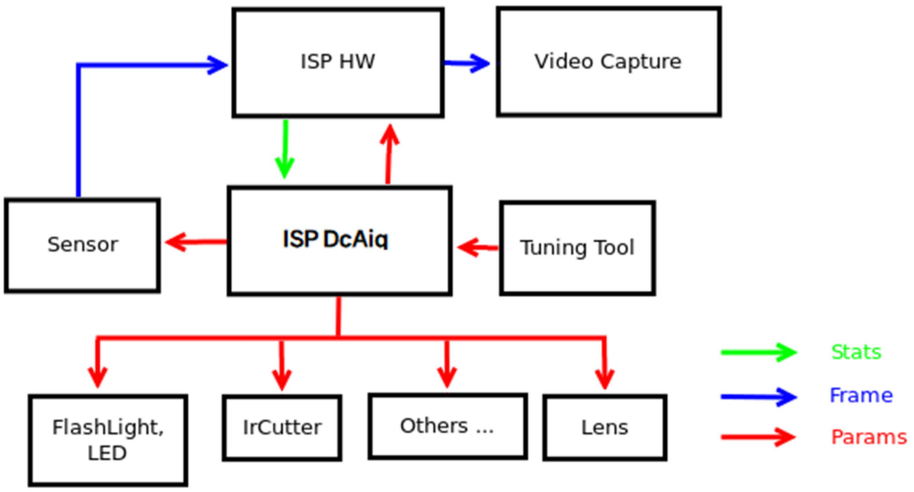


Figure 1-1 ISP system block diagram

The overall software and hardware block diagram of ISP is shown in Figure 1-1. The Sensor outputs the data stream to the ISP HW, and the ISP HW outputs the image after a series of image processing algorithms. DcAiq continuously obtains statistical data from ISP HW, and generates new parameters through 3A and other algorithms to feed back to each hardware module. Tuning tool can debug the parameters online in real time, and save and generate a new iq parameter file after debugging.

1.3 DcAiq Architecture

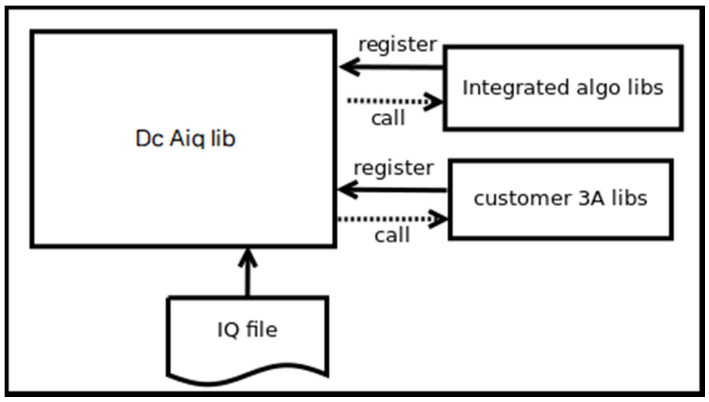


Figure 1-2 RkAiq overall architecture diagram

The design idea of ISP DcAiq software is shown in Figure 1-2. Mainly divided into the following four parts:

1. DcAiq lib dynamic library. The library contains the main logic part, which is responsible for obtaining statistics from the driver and transmitting them to each algorithm library.

2. Integrated algo libs. The static algorithm library provided by Dc100 has been registered to the DcAiq lib dynamic library by default.

3. customer 3A libs. Customers can implement their own 3A algorithm library or other algorithm libraries according to the algorithm library interface definition. After registering the custom algorithm library to the DcAiq lib dynamic library, you can choose to run the custom library or the Dc100 library according to the provided interface.

4. IQ file. The iq tuning result file saves algorithm related parameters and some system static parameters such as CIS.

1.4 Software Architecture

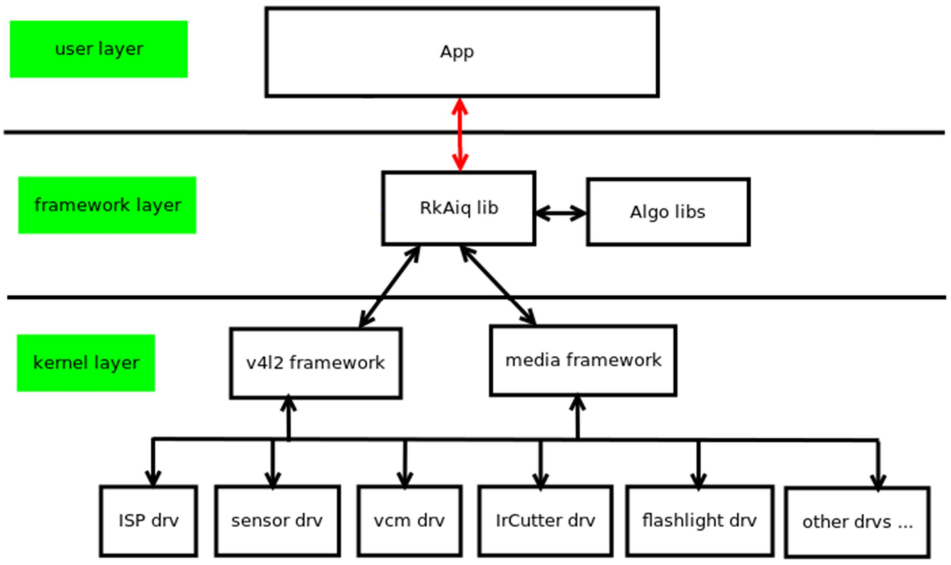


Figure 1-3 Software architecture block diagram

The ISP software block diagram is shown in Figure 1-3. Mainly divided into the following three layers:

1. Kernel layer. This layer contains all the hardware drivers of the Camera system, mainly including ISP driver, sensor driver, vcm driver, flashlight driver, IrCutter driver and so on. The drivers are implemented based on the V4L2 and Media framework.

2. framework layer. This layer is the integration layer of DcAiq lib. DcAiq lib can be directly integrated into the application.

3. User layer. User application layer.

1.4 API description

The API provided by DcAiq is divided into two levels: function level API and module level API. Among them, the function-level API is packaged based on the module-level API, mainly for some simple functional designs based on the module for product applications. The module level API provides detailed parameter settings and queries for the module, and does not differentiate between functions.

2. System Control

2.1 Overview

The system control part includes AIQ public attribute configuration, initialize AIQ, run AIQ, exit AIQ, set AIQ modules and other functions.

2.2 API Reference

2.2.1 dc_aiq_uapi_sysctl_init

【Description】

Initialize the AIQ context.

【Grammar】

```
dc_aiq_sys_ctx_t* rk_aiq_uapi_sysctl_init (const char* sns_ent_name, const char* iq_file_dir,
                                           dc_aiq_error_cb err_cb, dc_aiq_metas_cb metas_cb);
```

【Parameter】

Parameter name	Description	Input/Output
sns_ent_name	Sensor entity name	Input
iq_file_dir	Calibration parameter file path	Input
err_cb	Error callback function, can be NULL	Input
metas_cb	Meta data callback function, can be NULL	Input

【Return value】

Return value type	Description
dc_aiq_sys_ctx_t*	AIQ context pointer.

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

Should be called before other functions..

2.2.2 dc_aiq_uapi_sysctl_deinit

【Description】

Deinitialize the AIQ context environment.

【Grammar】

void dc_aiq_uapi_sysctl_deinit (dc_aiq_sys_ctx_t* ctx);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input

【Return value】

No.

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

It should not be called when the AIQ is in the start state.

2.2.3 dc_aiq_uapi_sysctl_prepare

【Description】

Prepare the AIQ operating environment.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_prepare (const dc_aiq_sys_ctx_t*ctx, uint32_t width, uint32_t height, dc_aiq_working_mode_t mode);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
width	The resolution width of sensor output, only for verification	Input
height	The resolution height output by the sensor, only for verification	Input
mode	ISP Pipeline working mode (NORMAL/HDR)	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details.

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

-It should be called before the dc_aiq_uapi_sysctl_start function.

-If you need to call this function after dc_aiq_uapi_sysctl_start, first call the dc_aiq_uapi_sysctl_stop function, and then call dc_aiq_uapi_sysctl_prepare to prepare the operating environment again.

2.2.4 dc_aiq_uapi_sysctl_start

【Description】

Start the AIQ control system. After the AIQ is started, it will continuously obtain 3A statistics from the ISP driver, run the 3A algorithm, and apply the calculated new parameters.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_start (const dc_aiq_sys_ctx_t* ctx);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see Error Code for details.

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

It should be called after the dc_aiq_uapi_sysctl_prepare function.

2.2.5 dc_aiq_uapi_sysctl_stop

【Description】

Stop the AIQ control system.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_stop (const dc_aiq_sys_ctx_t* ctx);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see Error Code for details.

【Requirement】

Header file：dcaiq_api.h

Library file：libdcaiq.so

【Notice】

No.

2.2.6 dc_aiq_uapi_sysctl_getStaticMetas

【Description】

Query sensor corresponding static information, such as resolution, data format, etc.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_getStaticMetas (const char*sns_ent_name, dc_aiq_static_info_t* static_info);

【Parameter】

Parameter name	Description	Input/Output
sns_ent_name	sensor entity name	Input
static_info	Static information structure pointer	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see Error Code for details.

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

2.2.7 dc_aiq_uapi_sysctl_enumStaticMetas

【Description】

Enumerate the static information obtained by AIQ.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_enumStaticMetas (int index, dc_aiq_static_info_t* static_info);

【Parameter】

Parameter name	Description	Input/Output
index	Index number, starting from 0	Input
static_info	Static information structure pointer	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see Error Code for details.

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

2.2.8 dc_aiq_uapi_sysctl_setModuleCtl

【Description】

AIQ module switch settings.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_setModuleCtl (const dc_aiq_sys_ctx_t* ctx, dc_aiq_module_id_t **mId**, bool mod_en);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
mId	Module ID	Input
mod_en	true to enable, false to disable	Input

【Return value】

Return value	Description
0	Success.

Not 0	Failure, see Error Code for details.

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

2.2.9 dc_aiq_uapi_sysctl_getModuleCtl

【Description】

AIQ module status query.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_getModuleCtl (const dc_aiq_sys_ctx_t* ctx, dc_aiq_module_id_t mId, bool* mod_en);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
mId	Module ID	Input
mod_en	Current status	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see Error Code for details.

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

2.2.10 dc_aiq_uapi_sysctl_regLib

【Description】

Register a custom algorithm library.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_regLib (const dc_aiq_sys_ctx_t* ctx, RkAiqAlgoDesComm* algo_lib_des);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
algo_lib_des	Algorithm description structure, the field id is the identification ID generated by AIQ	Input & Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see Error Code for details.

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

2.2.11 dc_aiq_uapi_sysctl_unRegLib

【Description】

Log out of the custom algorithm library.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_unRegLib (const dc_aiq_sys_ctx_t* ctx, const int algo_type, const int lib_id);

【Parameter】

Parameter name	Description	Input/Output

ctx	AIQ context pointer.	Input
algo_type	Type of algorithm module to be operated.	Input
lib_id	Algorithm library identification ID.	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see Error Code for details.

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

2.2.12 dc_aiq_uapi_sysctl_enableAxlib

【Description】

Set the running status of the custom algorithm library.

【Grammar】

```
XCamReturn dc_aiq_uapi_sysctl_enableAxlib (const dc_aiq_sys_ctx_t* ctx, const int algo_type,
                                           const int lib_id, bool enable);
```

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
algo_type	Type of algorithm module to be operated	Input
lib_id	Algorithm library identification ID	Input
enable	Status setting	Input

【Return value】

Return value	Description
0	Success.

Not 0	Failure, see Error Code for details.

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

-If lib_id is equal to the currently running algorithm library, this function can be called in any state except uninitialized.

-In other cases, it is only called in the prepared state, and the algorithm library identified by algo_type will be replaced by the new algorithm library identified by lib_id.

2.2.13 dc_aiq_uapi_sysctl_getAxlibStatus

【Description】

Get the state of the algorithm library.

【Grammar】

bool dc_aiq_uapi_sysctl_getAxlibStatus (const dc_aiq_sys_ctx_t* ctx, const int algo_type, const int lib_id);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
algo_type	Type of algorithm module to be operated	Input
lib_id	Algorithm library identification ID	Input

【Return value】

Return value	Description
false	Closed state
true	Enable state

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

2.2.14 dc_aiq_uapi_sysctl_getEnabledAxlibCtx

【Description】

Get the context structure of the enabled algorithm library.

【Grammar】

const RkAiqAlgoContext* dc_aiq_uapi_sysctl_getEnabledAxlibCtx (const dc_aiq_sys_ctx_t* ctx, const int algo_type);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
algo_type	Type of algorithm module to be operated	Input

【Return value】

Return value	Description
NULL	Get failed
Not NULL	Successfully obtained

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

-The returned algorithm context structure will be used by internal private functions. For userdefined algorithm libraries, this function should be called after dc_aiq_uapi_sysctl_enableAxlib, otherwise it will return NULL.

2.2.15 dc_aiq_uapi_sysctl_setCpsLtCfg

【Description】

Set the fill light control information.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_setCpsLtCfg (const dc_aiq_sys_ctx_t* ctx, dc_aiq_cpsl_cfg_t* cfg);

【Parameter】

Parameter name	Description	Input/Output

ctx	AIQ context pointer	Input
cfg	Fill light configuration structure pointer	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see Error Code for details.

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

2.2.16 dc_aiq_uapi_sysctl_getCpsLtInfo

【Description】

Obtain the fill light control information.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_getCpsLtInfo (const dc_aiq_sys_ctx_t* ctx, dc_aiq_cpsl_info_t* info);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
info	Fill light configuration structure pointer	Output

【Return value】

Return value	Description
0	Success
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

2.2.17 dc_aiq_uapi_sysctl_queryCpsLtCap

【Description】

Query the support capability of the fill light.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_queryCpsLtCap (const dc_aiq_sys_ctx_t* ctx, dc_aiq_cpsl_cap_t* cap);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
cap	Fill light support ability query structure pointer	Output

【Return value】

Return value	Description
0	Success
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

2.2.18 dc_aiq_uapi_sysctl_getBindedSnsEntNmByVd

【Description】

Query the sensor entity name corresponding to the video node.

【Grammar】

const char* dc_aiq_uapi_sysctl_getBindedSnsEntNmByVd (const char* vd);

【Parameter】

Parameter name	Description	Input/Output
vd	video path, such as /dev/video20	Input

【Return value】

Return value type	Description
sensor entity name	String pointer

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

The parameter must be the path of the ISPP scale node.

2.2.19 dc_aiq_uapi_sysctl_updateIq

【Description】

Dynamically update the currently used iq parameter file without stopping the data flow.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_updateIq (const dc_aiq_sys_ctx_t* sys_ctx, char* iqfile);

【Parameter】

Parameter name	Description	Input/Output
sys_ctx	AIQ context pointer	Input
iqfile	new iq file	Input

【Return value】

Return value	Description
0	Success
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h

Library file：libdcaiq.so

【Notice】

-iqfile needs to be a full path.

-Updating the iq parameter does not mean that the operating mode can be switched. If you need to switch between hdr and normal, it cannot be updated iq file implementation; but the switching of some functions can be achieved through different configurations of iq parameters, such as day and night switching Switching is achieved entirely through iq configuration.

-When switching iq, the configuration parameters in iq will override the user API settings. Such as AWB module, manual and automatic can be configured in iqMode, after executing this function, no matter what mode the current AWB is in, it will eventually be overwritten by the default configuration in the new iq.

2.2.20 dc_aiq_uapi_sysctl_setCrop

【Description】

Set crop parameters.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_setCrop (const dc_aiq_sys_ctx_t* sys_ctx, dc_aiq_rect_t rect);

【Parameter】

Parameter name	Description	Input/Output
sys_ctx	AIQ context pointer	Input
rect	crop parameter	Input

【Return value】

Return value	Description
0	Success
Not 0	Failure, see Error Code for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

-The minimum crop resolution is 64x64.

-The resolution to be set must exist in the IQ XML effect file.

-Must be called before rk_aiq_uapi_sysctl_prepare.

-rect.width must be an integer multiple of 4. For different data formats, there are different requirements for the horizontal offset rect.left: For raw8 and yuv422 formats, rect.left must be an integer multiple of 8. In raw10 and raw12 formats, rect.left must be an integer multiple of 4. In rgb888 format, rect.left must be an integer multiple of 24.

2.2.21 rk_aiq_uapi_sysctl_getCrop

【Description】

Get crop parameters.

【Grammar】

XCamReturn rk_aiq_uapi_sysctl_getCrop (const rk_aiq_sys_ctx_t* sys_ctx, rk_aiq_rect_t* rect);

【Parameter】

Parameter name	Description	Input/Output
sys_ctx	AIQ context pointer	Input
rect	crop parameter structure pointer	Output

【Return value】

Return value	Description
0	Success
Not 0	Failure, see Error Code for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

2.3 Data Type

2.3.1 dc_aiq_working_mode_t

【Description】

AIQ pipeline working mode

【Definiton】

```
typedef enum {
    DC_AIQ_WORKING_MODE_NORMAL,
    DC_AIQ_WORKING_MODE_ISP_HDR2 = 0x10,
    DC_AIQ_WORKING_MODE_ISP_HDR3 = 0x20,
} dc_aiq_working_mode_t;
```

【Members】

Member name	Description
DC_AIQ_WORKING_MODE_NORMAL	Normal mode
DC_AIQ_WORKING_MODE_ISP_HDR2	Two-frame HDR mode
DC_AIQ_WORKING_MODE_ISP_HDR3	Three-frame HDR mode

【Precaution】

-You need to query the modes supported by the sensor and AIQ first. If the set mode does not support the setting, the setting will be invalid.

2.3.2 dc_aiq_static_info_t

【Description】

AIQ static information.

【Definiton】

```
typedef struct {
    dc_aiq_sensor_info_t sensor_info;
    dc_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;
} dc_aiq_static_info_t;
```

【Members】

Member name	Description
sensor_info	Description of sensor name, supported resolution, etc.
lens_info	Lens Information
has_lens_vcm	Whether to bring vcm
has_fl	Whether with flash
fl_strth_adj_sup	Is it adjustable with flashlight
has_irc	With IR-CUT
fl_ir_strth_adj_sup	Is it adjustable with IR-CUT

2.3.3 dc_aiq_sensor_info_t

【Description】

sensor information

【Definiton】

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

【Members】

Member name	Description
sensor_name	The name of the sensor.
support_fmt	Supported formats
num	Number of supported formats
has_fl	Whether with flash
binded_strm_media_idx	Media node number mounted on the sensor

2.3.4 dc_aiq_module_id_t

【Description】

AIQ module ID.

【Definiton】

```
typedef enum {
    DC_MODULE_INVAL = 0,
    DC_MODULE_DPCC,
    DC_MODULE_BLS,
    DC_MODULE_LSC,
    DC_MODULE_AWB_GAIN,
    DC_MODULE_CTK,
    DC_MODULE_GOC,
    DC_MODULE_SHARP,
    DC_MODULE_AE,
    DC_MODULE_AWB,
    DC_MODULE_NR,
    DC_MODULE_GIC,
    DC_MODULE_3DLUT,
    DC_MODULE_LDCH,
    DC_MODULE_TNR,
    DC_MODULE_FEC,
    DC_MODULE_MAX
}dc_aiq_module_id_t;
```

【Members】

Member name	Description
DC_MODULE_DPCC	Bad pixel detection and correction
DC_MODULE_BLS	Black level
DC_MODULE_LSC	Lens shading correction
DC_MODULE_AWB_GAIN	White balance gain
DC_MODULE_CTK	Color correction
DC_MODULE_GOC	Gamma
DC_MODULE_SHARP	Sharpen
DC_MODULE_AE	Exposure
DC_MODULE_AWB	White balance
DC_MODULE_NR	Denoising

DC_MODULE_GIC	Green Balance
DC_MODULE_3DLUT	3DLUT
DC_MODULE_LDCH	LDCH
DC_MODULE_TNR	3D denoising
DC_MODULE_FEC	Fisheye correction

2.3.5 dc_aiq_cpsl_cfg_t

【Description】

Fill light setting information structure.

【Definiton】

```
typedef struct rk_aiq_cpsl_cfg_s {
    RKAiqOPMode_t mode;
    dc_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;
} dc_aiq_cpsl_cfg_t;
```

【Members】

Member name	Description
mode	Working mode
lght_src	Type of light source
gray_on	Whether to cut the screen to black and white after switching to night mode
sensitivity	Switching sensitivity in automatic mode, range [0,100]

sw_interval	Switching interval in automatic mode, in seconds
on	Whether to switch to night mode in manual mode
strength_led	LED light intensity in manual mode, range [0,100]
strength_ir	Infrared light intensity in manual mode, range [0,100]

2.4.6 dc_aiq_cpsl_info_t

【Description】

Fill light query information structure.

【Definiton】

```
typedef struct dc_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;
} dc_aiq_cpsl_info_t;
```

【Members】

Member name	Description
mode	Working mode
lght_src	Type of light source
gray	Whether to cut the screen to black and white after switching to night mode
sensitivity	Switching sensitivity in automatic mode, range [0,100]
sw_interval	Switching interval in automatic mode, in seconds
on	Whether to switch to night mode in manual mode
strength_led	LED light intensity in manual mode, range [0,100]
strength_ir	Infrared light intensity in manual mode, range [0,100]

2.4.7 dc_aiq_cpsl_cap_t

【Description】

Supplement light support capability structure.

【Definiton】

```
typedef struct dc_aiq_cpsl_cap_s {
    int32_t supported_modes[DC_AIQ_OP_MODE_MAX];
    uint8_t modes_num;
    int32_t supported_lght_src[DC_AIQ_CPSLS_MAX];
    uint8_t lght_src_num;
    dc_aiq_range_t strength_led;
    dc_aiq_range_t sensitivity;
    dc_aiq_range_t strength_ir;
} dc_aiq_cpsl_cap_t;
```

【Members】

Member name	Description
supported_modes	Supported working modes
modes_num	Number of supported modes
gray	Whether to cut the screen to black and white after switching to night mode
supported_lght_src	Supported light sources
lght_src_num	Number of supported light sources
strength_led	LED intensity range
sensitivity	Sensitivity range
strength_ir	Intensity range of infrared lamp

2.4.8 dc_aiq_rect_t

【Description】

Define crop parameter structure.

【Definiton】

```
typedef struct dc_aiq_rect_s {
    int left;
```

```
    int top;
    int width;
    int height;
} dc_aiq_rect_t;
```

【Members】

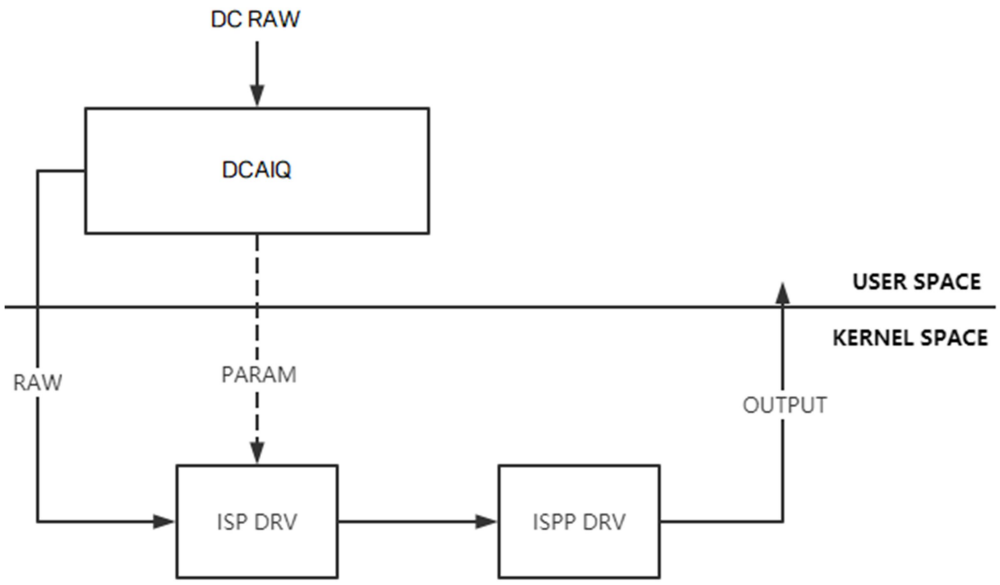
Member name	Description
left	horizontal output offset.
top	vertical output offset.
width	horizontal output size.
height	vertical output size.

3. Offline frame processing

3.1 Overview

DCAIQ provides offline RAW frame processing function, that is, DC customized RAW format files are parsed by DCAIQ and then sent into ISP for processing, and output as an image that can be normally displayed.

3.2 Functional block diagram



Offline frame processing block diagram

3.3 Function description

- Support RK-RAW format file input. Calling the file input interface, the calling process will be blocked until the file is successfully processed and output.
- Support RK-RAW format buffer input and asynchronous working mode. Calling the buffer input interface, the calling process will not be blocked, and the callback function will be called after the buffer processing is completed (if there is a registered callback function).
- Support RK-RAW format buffer input, synchronous working mode. Calling the buffer input interface, the calling process will be blocked until the buffer processing is successfully processed and output .

3.4 Supported RAW format

Support raw8/raw10/raw12, BGGR/GBRG/GRBG/RGGB.

3.5 API Reference

3.5.1 dc_aiq_uapi_sysctl_prepareRkRaw

【Description】

Prepare the environment for RK Raw format data processing.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_prepareRkRaw (const dc_aiq_sys_ctx_t* ctx, dc_aiq_raw_prop_t prop);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
prop	RK Raw format property parameters	Input

【Return value】

Return value	Description
0	Success
Not 0	Failure, see Error Code for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

This interface must be called before dc_aiq_uapi_sysctl_prepare.

3.5.2 dc_aiq_uapi_sysctl_enqueueRkRawBuf

【Description】

input RK Raw format buffer.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_enqueueRkRawBuf (const dc_aiq_sys_ctx_t* ctx, void* rawdata, bool sync);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
rawdata	RK Raw format data buffer	Input
sync	true: Synchronous mode. false: Asynchronous mode	Input

【Return value】

Return value	Description
0	Success
Not 0	Failure, see Error Code for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

If you need extra operation on the rawdata that has been processed in asynchronous mode, you can use the rk_aiq_uapi_sysctl_registRkRawCb interface to register the callback function, and the rawdata buffer will be passed into the callback function.

3.5.3 dc_aiq_uapi_sysctl_enqueueRkRawFile

【Description】

Input RK Raw format file.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_enqueueRkRawFile (const dc_aiq_sys_ctx_t* ctx, const char* path);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
path	RK Raw format file path	Input

【Return value】

Return value	Description
0	Success
Not 0	Failure, see Error Code for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

The interface is synchronous..

3.5.4 dc_aiq_uapi_sysctl_registRkRawCb

【Description】

Register callback function.

【Grammar】

XCamReturn dc_aiq_uapi_sysctl_registRkRawCb (const dc_aiq_sys_ctx_t* ctx, void (*callback)(void*));

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
callback	Callback function pointer	Input

【Return value】

Return value	Description

0	Success
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

-This interface is not necessary.

-If a callback is registered, the callback will only be called when the dc_aiq_uapi_sysctl_enqueueRkRawBuf interface is called in asynchronous mode and the Raw data processing is completed.

3.6 Type of data

3.6.1 dc_aiq_raw_prop_t

【Description】

RK Raw parameter structure

【Definiton】

```
typedef struct dc_aiq_raw_prop_s {
    uint32_t frame_width;
    uint32_t frame_height;
    dc_aiq_format_t format;
    dc_aiq_rawbuf_type_t rawbuf_type;
}dc_aiq_raw_prop_t;
```

【Members】

Member name	Description
frame_width	RK Raw image width.
frame_height	RK Raw image height.
format	RK Raw bayer pattern.
rawbuf_type	RK Raw type.

3.6.2 dc_aiq_rawbuf_type_t

【Description】

RK Raw type structure.

【Definiton】

```
typedef enum rk_aiq_rawbuf_type_s {
    DC_AIQ_RAW_ADDR,
    DC_AIQ_RAW_FD,
    DC_AIQ_RAW_DATA,
    DC_AIQ_RAW_FILE
}rk_aiq_rawbuf_type_t;
```

【Members】

Member name	Description
DC_AIQ_RAW_ADDR	Indicates that the 'Raw data' section of the input RK Raw format buffer stores the virtual address of the DMA BUF in this process, not the raw data itself.
DC_AIQ_RAW_FD	Indicates that the 'Raw data' section in the input RK Raw format buffer stores the BUF fd in this process, not the RAW data itself.
DC_AIQ_RAW_DATA	Indicates that the raw image data is stored in the 'Raw data' section of the input RK Raw format buffer.
DC_AIQ_RAW_FILE	Indicates that the input is a RK Raw format file.

【Notice】

No.

4. IMGPROC

4.1 Overview

imgproc refers to the module that affects the image effect.

4.2 FEC

4.2.1 Function description

The squint distortion, pincushion, barrel distortion, etc. caused by the distortion of the optical system and the electronic scanning system may cause the geometric characteristics of the image to be distorted. Image distortion correction is a process of transforming a distorted image into an ideal image in a certain transformation manner.

This module corrects image distortion in the x and y directions.

4.2.2 Important concepts

Distortion actually refers to the distortion of the photographed object relative to the object itself.

4.2.3 Functional API Reference

4.2.3.1 dc_aiq_uapi_setFecEn

【Description】

Enable fisheye distortion correction function.

【Grammar】

XCamReturn dc_aiq_uapi_setFecEn (const dc_aiq_sys_ctx_t* ctx, bool en);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
en	enable	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

-This interface can only be called before dc_aiq_uapi_sysctl_prepare, that is, it cannot support dynamic switch while AIQ is running. If you need dynamic switch distortion correction effect, you can use the dc_aiq_uapi_setFecBypass interface.

-After enabling fisheye distortion correction, DDR bandwidth and CPU load will increase, which may affect the camera's capture frame rate.

4.2.3.2 dc_aiq_uapi_setFecCorrectDirection

【Description】

Set the direction of fisheye distortion correction.

【Grammar】

XCamReturn dc_aiq_uapi_setFecCorrectDirection (const dc_aiq_sys_ctx_t* ctx,
const fec_correct_direction_t direction);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
direction	Correction direction	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

This interface can only be called before dc_aiq_uapi_sysctl_prepare, that is, it cannot be call while AIQ is running.

4.2.3.3 dc_aiq_uapi_setFecBypass

【Description】

Bypass fisheye distortion correction function. The data also be processd by FEC, but correction strength is equivalent to no correction.

【Grammar】

XCamReturn dc_aiq_uapi_setFecBypass (const dc_aiq_sys_ctx_t* ctx, bool bypass);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
bypass	Correction effect switch	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

4.2.3.4 dc_aiq_uapi_setFecCorrectLevel

【Description】

Set the fisheye distortion correction level.

【Grammar】

```
XCamReturn dc_aiq_uapi_setFecCorrectLevel (const dc_aiq_sys_ctx_t* ctx, int correctLevel);
```

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
correctLevel	Correction level	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

4.2.4 Type of data

4.2.4.1 fec_correct_direction_t

【Description】

fec correction direction

【Definiton】

```
typedef enum fec_correct_direction_e {
    FEC_CORRECT_DIRECTION_X = 0x1,
    FEC_CORRECT_DIRECTION_Y,
    FEC_CORRECT_DIRECTION_XY
} fec_correct_direction_t;
```

【Members】

Member name	Description
FEC_CORRECT_DIRECTION_X	Only correct the X direction.
FEC_CORRECT_DIRECTION_Y	Only correct the Y direction.
FEC_CORRECT_DIRECTION_XY	Correction in both XY directions.

4.2.4.2 dc_aiq_fec_attrib_t

【Description】

fec correction direction

【Definiton】

```
typedef struct dc_aiq_fec_cfg_s {
    unsigned int en;
    int bypass;
    int correct_level;
```

```
    fec_correct_direction_t direction;
} dc_aiq_fec_cfg_t;
```

【Members】

Member name	Description
en	Enable/Disable fec
bypass	Bypass fec
correct_level	Set fec correction level (0~255)
direction	Set fec correction direction

4.3 LDCH

4.3.1 Function description

The squint distortion, pincushion, barrel distortion, etc. caused by the distortion of the optical system and the electronic scanning system may cause the geometric characteristics of the image to be distorted. Image distortion correction is a process of transforming a distorted image into an ideal image in a certain transformation manner. This module only corrects the image distortion in the x direction.

4.3.2 Functional API Reference

4.3.2.1 dc_aiq_uapi_setLdchEn

【Description】

Enable horizontal distortion correction function.

【Grammar】

XCamReturn dc_aiq_uapi_setLdchEn (const dc_aiq_sys_ctx_t* ctx, bool en);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
en	enable	Input

【Return value】

Return value	Description

0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

4.3.2.2 dc_aiq_uapi_setLdchCorrectLevel

【Description】

Enable horizontal distortion correction function.

【Grammar】

XCamReturn dc_aiq_uapi_setLdchCorrectLevel (const dc_aiq_sys_ctx_t* ctx, int correctLevel);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
correctLevel	Correction level	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

4.3.3 Type of data

4.3.3.1 rk_aiq_ldch_attrib_t

【Description】

ldch attribute configuration

【Definiton】

```
typedef struct dc_aiq_ldch_cfg_s {
    unsigned int en;
    int correct_level;
} dc_aiq_ldch_cfg_t;
```

【Members】

Member name	Description
en	Enable/Disable ldch
correct_level	Set ldch correction level (0~255)

4.4 HDR

4.4.1 Function description

HDR (High Dynamic Range Imaging, HDRI or HDR), in computer graphics and photography, is achieved through calculation using existing equipment to obtain a larger dynamic range than ordinary digital imaging technology (that is, larger The difference between light and shade) is a technique for images. The purpose of HDR is to correctly restore the luminance ratio of the real scene that exceeds the dynamic range of the existing equipment.

4.4.2 Important concepts

The module contains two parts: Merge and Tmo, Tmo can be used alone, Merge needs to be used together with Tmo.

4.4.3 Functional API Reference

4.4.2.1 dc_aiq_uapi_setHDRMode

【Description】

Set HDR working mode.

【Grammar】

XCamReturn dc_aiq_uapi_setHDRMode (const dc_aiq_sys_ctx_t* ctx, opMode_t mode);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
mode	Working mode	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

4.4.2.2 dc_aiq_uapi_getHDRMode

【Description】

Set HDR working mode.

【Grammar】

XCamReturn dc_aiq_uapi_getHDRMode (const dc_aiq_sys_ctx_t* ctx, opMode_t mode);

【Parameter】

Parameter name	Description	Input/Output

ctx	AIQ context pointer	Input
mode	Working mode	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

4.4.2.3 dc_aiq_uapi_setMHDRStrth

【Description】

Set the HDR strength in manual mode.

【Grammar】

XCamReturn dc_aiq_uapi_setMHDRStrth (const dc_aiq_sys_ctx_t* ctx, bool on, unsigned int level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
on	Working mode	Input
level	Strength. Value range: [1,100]	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

4.4.2.4 dc_aiq_uapi_getMHDRStrth

【Description】

Get the HDR strength in manual mode.

【Grammar】

XCamReturn dc_aiq_uapi_getMHDRStrth (const dc_aiq_sys_ctx_t* ctx, bool* on, unsigned int* level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
on	Working mode	Output
level	Strength. Value range: [1,100]	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Notice】

No.

4.4.4 Type of data

4.4.4.1 hdr_OpMode_t

【Description】

Define white balance working mode

【Definiton】

```
typedef enum hdr_OpMode_s {
    HDR_OpMode_Api_OFF = 0,
    HDR_OpMode_Auto = 1,
    HDR_OpMode_MANU = 2,
    HDR_OpMode_SET_LEVEL = 3,
    HDR_OpMode_DarkArea = 4,
    HDR_OpMode_Tool = 5,
} hdr_OpMode_t;
```

【Members】

Member name	Description
HDR_OpMode_Api_OFF	Api off mode
HDR_OpMode_Auto	Automatic mode
HDR_OpMode_MANU	Manual mode
HDR_OpMode_SET_LEVEL	Fast mode, adjust the HDR effect by setting the level
HDR_OpMode_DarkArea	Dark area boost mode, can be used in both linear and HDR
HDR_OpMode_Tool	Tool Mode

4.4.4.2 FastMode_t

【Description】

Define HDR fast mode attributes

【Definiton】

```
typedef struct FastMode_s {
  int level;
} FastMode_t;
```

【Members】

Member name	Description
level	Fast mode level

4.4.4.3 DarkArea_t

【Description】

Define HDR dark area boost mode attributes

【Definiton】

```
typedef struct DarkArea_s {
  int level;
} FastMode_t;
```

【Members】

Member name	Description
level	Fast mode level

4.5 Noise Removal

4.5.1 Function description

Image noise refers to unnecessary or redundant interference information existing in image data. Image denoising is the process of reducing noise in digital images.

4.5.2 Functional API Reference

4.5.2.1 dc_aiq_uapi_setNRMode

【Description】

Set the denoising mode.

【Grammar】

XCamReturn dc_aiq_uapi_setNRMode (const dc_aiq_sys_ctx_t* ctx, opMode_t mode);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
mode	Working mode	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.5.2.2 dc_aiq_uapi_getNRMode

【Description】

Get the current denoising mode.

【Grammar】

XCamReturn dc_aiq_uapi_getNRMode (const dc_aiq_sys_ctx_t* ctx, opMode_t* mode);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
mode	Working mode	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.5.2.3 dc_aiq_uapi_setANRStrth

【Description】

Set the normal denoising intensity.

【Grammar】

XCamReturn dc_aiq_uapi_setANRStrth (const dc_aiq_sys_ctx_t* ctx, unsigned int level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
level	Denoising intensity	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.5.2.4 dc_aiq_uapi_getANRStrth

【Description】

Get the normal denoising intensity.

【Grammar】

XCamReturn dc_aiq_uapi_getANRStrth (const dc_aiq_sys_ctx_t* ctx, unsigned int* level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
level	Denoising intensity	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.5.2.5 dc_aiq_uapi_setMSpaNRStrth

【Description】

Set the airspace denoising intensity.

【Grammar】

XCamReturn dc_aiq_uapi_setMSpaNRStrth (const dc_aiq_sys_ctx_t* ctx, bool on , unsigned int level);

【Parameter】

Parameter name	Description	Input/Output

ctx	AIQ context pointer	Input
on	switch	Input
level	Denoising intensity	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.5.2.6 dc_aiq_uapi_getMSpaNRStrth

【Description】

Get the airspace denoising intensity.

【Grammar】

XCamReturn dc_aiq_uapi_getMSpaNRStrth (const dc_aiq_sys_ctx_t* ctx, bool* on , unsigned int* level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
on	switch	Output
level	Denoising intensity	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.5.2.7 dc_aiq_uapi_setMTNRStrth

【Description】

Set the time domain denoising intensity.

【Grammar】

XCamReturn dc_aiq_uapi_setMTNRStrth (const dc_aiq_sys_ctx_t* ctx, bool on , unsigned int level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
on	switch	Input
level	Denoising intensity	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.5.2.8 dc_aiq_uapi_getMTNRStrth

【Description】

Set the time domain denoising intensity.

【Grammar】

XCamReturn dc_aiq_uapi_getMTNRStrth (const dc_aiq_sys_ctx_t* ctx, bool* on , unsigned int* level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
on	switch	Output
level	Denoising intensity	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.6 Defog

4.6.1 Function description

Defog is a defogging enhancement by dynamically changing the contrast and brightness of the image.

4.6.2 Functional API Reference

4.6.2.1 dc_aiq_uapi_setDhzMode

【Description】

Set the denoising mode.

【Grammar】

XCamReturn dc_aiq_uapi_setDhzMode (const dc_aiq_sys_ctx_t* ctx, opMode_t mode);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
mode	Working mode	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.6.2.2 dc_aiq_uapi_getDhzMode

【Description】

Get the current defogging working mode.

【Grammar】

XCamReturn dc_aiq_uapi_getDhzMode (const dc_aiq_sys_ctx_t* ctx, opMode_t* mode);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
mode	Working mode	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.6.2.3 dc_aiq_uapi_setMDhzStrth

【Description】

Set the intensity of defogging work.

【Grammar】

XCamReturn dc_aiq_uapi_setMDhzStrth (const dc_aiq_sys_ctx_t* ctx, bool on, unsigned int level);

【Parameter】

Parameter name	Description	Input/Output

ctx	AIQ context pointer	Input
on	Switch	Input
level	Intensity	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.6.2.4 dc_aiq_uapi_getMDhzStrth

【Description】

Get the intensity of dehazing work.

【Grammar】

```
XCamReturn dc_aiq_uapi_getMDhzStrth (const dc_aiq_sys_ctx_t* ctx, bool* on, unsigned int* level);
```

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
mode	Working mode	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.6.2.5 dc_aiq_uapi_enableDhz

【Description】

Turn on the defogging function.

【Grammar】

XCamReturn dc_aiq_uapi_enableDhz (const dc_aiq_sys_ctx_t* ctx);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.6.2.6 dc_aiq_uapi_disableDhz

【Description】

Turn off the defogging function.

【Grammar】

XCamReturn dc_aiq_uapi_disableDhz (const dc_aiq_sys_ctx_t* ctx);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input

【Return value】

Return value	Description
0	Success.

Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.7 ACM

4.7.1 Function description

ACM (Auto Color Managment) provides basic preference color adjustment function, Adjust the degree, contrast, saturation and chroma to achieve the adjustment of the color preference.

4.7.2 Functional API Reference

4.7.2.1 dc_aiq_uapi_setBrightness

【Description】

Set the brightness level.

【Grammar】

XCamReturn dc_aiq_uapi_setBrightness (const dc_aiq_sys_ctx_t* ctx, unsigned int level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
level	Brightness value level. Value range: [0,255] The default value is 128	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.7.2.2 dc_aiq_uapi_getBrightness

【Description】

Get the brightness level.

【Grammar】

XCamReturn dc_aiq_uapi_getBrightness (const dc_aiq_sys_ctx_t* ctx, unsigned int* level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
level	Current brightness level	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.7.2.3 dc_aiq_uapi_setContrast

【Description】

Set the contrast level.

【Grammar】

XCamReturn dc_aiq_uapi_setContrast (const dc_aiq_sys_ctx_t* ctx, unsigned int level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
level	Contrast level. Value range: [0,255] The default value is 128	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.7.2.4 dc_aiq_uapi_getContrast

【Description】

Set the contrast level.

【Grammar】

XCamReturn dc_aiq_uapi_getContrast (const dc_aiq_sys_ctx_t* ctx, unsigned int* level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
level	Current contrast level	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.7.2.5 dc_aiq_uapi_setSaturation

【Description】

Set the saturation level.

【Grammar】

XCamReturn dc_aiq_uapi_setSaturation (const dc_aiq_sys_ctx_t* ctx, unsigned int level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
level	Saturation level. Value range: [0,255] The default value is 128	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.7.2.6 dc_aiq_uapi_getSaturation

【Description】

Get the saturation level.

【Grammar】

XCamReturn dc_aiq_uapi_getSaturation (const dc_aiq_sys_ctx_t* ctx, unsigned int* level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
level	Current saturation level	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.7.2.7 dc_aiq_uapi_setHue

【Description】

Set the chroma level.

【Grammar】

XCamReturn dc_aiq_uapi_setHue (const dc_aiq_sys_ctx_t* ctx, unsigned int level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
level	Chroma level. Value range: [0,255] The default value is 128	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.7.2.8 dc_aiq_uapi_getHue

【Description】

Get the chroma level.

【Grammar】

XCamReturn dc_aiq_uapi_getHue (const dc_aiq_sys_ctx_t* ctx, unsigned int* level);

【Parameter】

Parameter name	Description	Input/Output

ctx	AIQ context pointer	Input
level	Current chroma level	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.8 Sharpen

4.8.1 Function description

The Sharpen module is used to enhance the sharpness of the image, including adjusting the sharpening properties of the edge of the image and enhancing the details of the image And texture.

4.8.2 Functional API Reference

4.8.2.1 dc_aiq_uapi_setSharpness

【Description】

Set the sharpening level.

【Grammar】

XCamReturn dc_aiq_uapi_setSharpness (const dc_aiq_sys_ctx_t* ctx, unsigned int level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
level	Sharpening level. Value range: [0,100] The default value is 50	Input

【Return value】

Return value	Description

0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.8.2.2 dc_aiq_uapi_getSharpness

【Description】

Get the sharpening level.

【Grammar】

XCamReturn dc_aiq_uapi_getSharpness (const dc_aiq_sys_ctx_t* ctx, unsigned int* level);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
level	sharpening level	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.9 Gamma

4.9.1 Function description

Gamma performs a non-linear conversion of the luminance space of the image to adapt to the output device.

4.9.2 Functional API Reference

4.9.2.1 dc_aiq_uapi_setGammaCoef

【Description】

Set gamma.

【Grammar】

XCamReturn dc_aiq_uapi_setGammaCoef (const dc_aiq_sys_ctx_t* ctx, dc_aiq_gamma_attrib_t gammaAttr);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
gammaAttr	Gamma software attribute structure	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

【Description】

The gamma curve in Api is not switched according to the scene. If the scene changes, please set the gamma curve through the api again.

4.9.3 Type of data

4.9.3.1 dc_aiq_gamma_op_mode_t

【Description】

Define Gamma working mode

【Definiton】

```
typedef enum dc_aiq_gamma_op_mode_s {
    DC_AIQ_GAMMA_MODE_OFF = 0,
    DC_AIQ_GAMMA_MODE_MANUAL = 1,
    DC_AIQ_GAMMA_MODE_TOOL = 2,
} dc_aiq_gamma_op_mode_t;
```

【Members】

Member name	Description
DC_AIQ_GAMMA_MODE_OFF	Api off mode.
DC_AIQ_GAMMA_MODE_MANUAL	Api automatic mode.
DC_AIQ_GAMMA_MODE_TOOL	Api tool mode.

4.9.3.2 dc_gamma_curve_type_t

【Description】

Define Gamma working mode

【Definiton】

```
typedef enum dc_gamma_curve_type_s {
    DC_GAMMA_CURVE_TYPE_DEFUALT = 0,
    DC_GAMMA_CURVE_TYPE_SRGB = 1,
    DC_GAMMA_CURVE_TYPE_HDR = 2,
    DC_GAMMA_CURVE_TYPE_USER_DEFINE1 = 3,
    DC_GAMMA_CURVE_TYPE_USER_DEFINE2 = 4,
} dc_gamma_curve_type_t;
```

【Members】

Member name	Description
DC_GAMMA_CURVE_TYPE_DEFUALT	Use Gamma curve in IQ file.
DC_GAMMA_CURVE_TYPE_SRGB	Use sRGB standard Gamma 2.2 curve.
DC_GAMMA_CURVE_TYPE_HDR	Use HDR mode Gamma curve.
DC_GAMMA_CURVE_TYPE_USER_DEFINE1	Use user-defined Gamma curve 1.
DC_GAMMA_CURVE_TYPE_USER_DEFINE2	Use user-defined Gamma curve 2.

4.9.3.3 dc_gamma_curve_usr_define1_para_t

【Description】

Define user-defined Gamma curve 1 attributes in manual mode.

【Definiton】

```
typedef struct dc_gamma_curve_usr_define1_para_s {
    float coef1;
    float coef2;
} dc_gamma_curve_usr_d
```

【Members】

Member name	Description
coef1	The Gamma curve slope.
coef2	The zero slope of the Gamma curve.

4.9.3.4 dc_gamma_curve_usr_define2_para_t

【Description】

Define user-defined Gamma curve 2 attributes in manual mode.

【Definiton】

```
typedef struct dc_gamma_curve_usr_define2_para_s {
    int gamma_out_segnum;
    int gamma_out_offset;
    int gamma_table[45];
} dc_gamma_curve_usr_define2_para_t;
```

【Members】

Member name	Description
gamma_out_segnum	Define the X-axis spacing of the Gamma curve, 0: Log space, 1: Linear space.
gamma_out_offset	Gamma curve offset.
gamma_table	Gamma curve.

4.9.3.5 Agamma_api_manual_t

【Description】

Define manual Gamma attributes.

【Definiton】

```
typedef struct Agamma_api_manual_s {
    bool en;
    dc_gamma_curve_type_t CurveType;
    dc_gamma_curve_usr_define1_para_t user1;
    dc_gamma_curve_usr_define2_para_t user2;
} Agamma_api_manual_t
```

【Members】

Member name	Description
en	Function switch.
CurveType	Curve Type.
user1	User-defined Gamma Curve 1.
user2	User-defined Gamma Curve 2.

4.9.3.6 CalibDb_Gamma_t

【Description】

Define Gamma attributes in tool mode.

【Definiton】

```
typedef struct CalibDb_Gamma_s {
    unsigned char gamma_en;
    unsigned char gamma_out_segnum;
    unsigned char gamma_out_offset;
    float curve_normal[45];
    float curve_hdr[45];
    float curve_night[45];
} CalibDb_Gamma_t;
```

【Members】

Member name	Description
gamma_en	Function switch

gamma_out_segnum	Define the X-axis spacing of the Gamma curve, 0: Log Space, 1: Linear space
gamma_out_offset	Gamma curve offset
curve_normal	Gamma curve in linear mode
curve_hdr	Gamma curve in HDR mode
curve_night	Gamma curve in night mode

4.9.3.7 dc_aiq_gamma_attr_t

【Description】

Define Gamma attributes.

【Definiton】

```
typedef struct dc_aiq_gamma_attr_s {
    dc_aiq_gamma_op_mode_t mode;
    Agamma_api_manual_t stManual;
    CalibDb_Gamma_t stTool;
    Int Scene_mode;
} dc_aiq_gamma_attr_t;
```

【Members】

Member name	Description
mod	Api mode
stManual	Manual Gamma parameter
stTool	Tool Gamma Parameters
Scene_mode	Scene Mode

4.10 Other

4.10.1 Functional API Reference

4.10.1.1 dc_aiq_uapi_setGrayMode

【Description】

Set how the black and white image mode works.

【Grammar】

XCamReturn dc_aiq_uapi_setGrayMode (const dc_aiq_sys_ctx_t* ctx, dc_aiq_gray_mode_t mode);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
mode	Working mode	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.10.1.2 dc_aiq_uapi_getGrayMode

【Description】

Get how the black and white image mode works.

【Grammar】

rk_aiq_gray_mode_t dc_aiq_uapi_getGrayMode (const dc_aiq_sys_ctx_t* ctx);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.10.1.3 dc_aiq_uapi_setFrameRate

【Description】

Set the image output frame rate.

【Grammar】

XCamReturn dc_aiq_uapi_setFrameRate (const dc_aiq_sys_ctx_t* ctx, frameRateInfo_t info);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
frameRateInfo_t	Frame Rate Information Structure	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.10.1.4 dc_aiq_uapi_getFrameRate

【Description】

Get image output frame rate information.

【Grammar】

XCamReturn dc_aiq_uapi_setFrameRate (const dc_aiq_sys_ctx_t* ctx, frameRateInfo_t* info);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
frameRateInfo_t	Frame Rate Information Structure	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.10.1.5 dc_aiq_uapi_setMirroFlip

【Description】

Set image mirroring and flipping.

【Grammar】

XCamReturn dc_aiq_uapi_setMirroFlip (const dc_aiq_sys_ctx_t* ctx, bool mirror, bool flip);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
mirror	Frame Rate Information Structure	Input
flip	Whether to flip	Input

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.10.1.6 dc_aiq_uapi_getMirroFlip

【Description】

Obtain image mirroring and flipping information.

【Grammar】

XCamReturn dc_aiq_uapi_getMirrorFlip (const dc_aiq_sys_ctx_t* ctx, bool* mirror, bool* flip);

【Parameter】

Parameter name	Description	Input/Output
ctx	AIQ context pointer	Input
mirror	Frame Rate Information Structure	Output
flip	Whether to flip	Output

【Return value】

Return value	Description
0	Success.
Not 0	Failure, see error code table for details

【Requirement】

Header file：dcaiq_api.h
Library file：libdcaiq.so

4.10.2 Type of data

4.10.2.1 dc_aiq_gray_mode_t

【Description】

Black and white switching working mode

【Definiton】

```
typedef enum dc_aiq_gray_mode_e {
    DC_AIQ_GRAY_MODE_CPSL,
    DC_AIQ_GRAY_MODE_OFF,
    DC_AIQ_GRAY_MODE_ON,
} dc_aiq_gray_mode_t;
```

【Members】

Member name	Description
DC_AIQ_GRAY_MODE_CPSL	RK Raw image width.

DC_AIQ_GRAY_MODE_OFF	RK Raw image height.
DC_AIQ_GRAY_MODE_ON	RK Raw bayer pattern.

5. Error code

Error code	Description
0	Success
-1	Failure
-2	Invalid parameter
-3	Insufficient memory
-4	File operation failed
-5	ANALYZER module error
-6	ISP module error
-7	Sensor driver error
-8	Thread operation error
-9	IOCTL operation error
-10	Timing error
-20	Timeout
-21	Out of range
-255	Unknown error

6. Acronyms

Abbreviation	Full name
CIS	Camera Image Sensor
DcAiq	DC100 Automatical Image Quality
ISP	Image Signal Process
IQ Tuning	Image Quality Tuning