Rockchip RKADK Development Guide

ID: RK-KF-YF-904

Release Version: V2.2.3

Release Date: 2024-09-11

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

Rockchip Electronics Co., Ltd.

No.18 Building, A District, No.89, software Boulevard Fuzhou, Fujian,PRC

Website: www.rock-chips.com

Customer service Tel: +86-4007-700-590

Customer service Fax: +86-591-83951833

Customer service e-Mail: fae@rock-chips.com

Preface

Overview

This document is going to introduce development reference of Rockchip Rkadk component.

Product Version

Chipset	Kernel Version
RV1126/RV1109	Linux 4.19
RV1106/RV1103	Linux 5.10

Intended Audience

This document (this guide) is mainly intended for:

Technical support engineers

Software development engineers

Revision History

Version	Author	Date	Change Description
V1.0.0	CTF	2021-05-02	Initial version
V1.1.0	CTF	2021-08-23	1. Add API introduction: (1) RKADK_RECORD_GetAencChn (2) RKADK_GetThmInMp4 (3) RKADK_PHOTO_GetThmInJpg (4) RKADK_PARAM_GetVencChnId 2. New module introduction: (1) Local preview module `Display` (2) Live broadcast module `Live`: includes RTSP and RTMP sub-modules
V1.2.0	CTF	2021-09-26	1. Add thumbnail extension API: (1) RKADK_GetThmInMp4Ex (2) RKADK_ThmBufFree (3) RKADK_PHOTO_GetThmInJpgEx (4) RKADK_PHOTO_ThumbBufFree 2. Support multi sensor 3. Support setting INI file path 4. Add VI ISP module 5. Support setting VENC GOP
V1.3.0	GZC	2021-11-30	New storage module
V1.3.1	CTF	2021-12-20	1. Add API introduction: (1) RKADK_PHOTO_GetData (2) RKADK_PHOTO_FreeData (3) RKADK_PLAYER_GetDuration 2. Add INI configuration notes
V2.0.0	CTF	2023-01-29	1. Adapt to general Linux SDK 2. Add API introduction: (1) RKADK_OSD_Init (2) RKADK_OSD_Deinit (3) RKADK_OSD_UpdateOsdSize (4) RKADK_OSD_AttachToStream (5) RKADK_OSD_DettachFromStream (6) RKADK_OSD_UpdateDisplayAttr
V2.1.0	CTF	2023-04-27	Compatible with RV1109/RV1126 platforms

Version	Author	Date	Change Description
V2.2.0	CTF	2023-11-07	1. Added UI overlay module 2. Added JPEG Slice function 3. Added the following functions to Player: (1) Supported third-party demuxer library (2) Supports RTSP network stream playback (3) Supports screen snapshot function (4) Compatible with RV1106/RV1103, RK3308, RK3506 platforms 4. Added Record File Cache description 5. Add Record/Photo rotation, flip, mirror API description 6. Update ini description 7. Update test cases
V2.2.1	CTF	2024-01-04	1. Added Post AI ISP function
V2.2.2	CTF	2024-04-23	Added AOV(Always On Video) time-lapse recording function
V2.2.3	CTF	2024-09-11	1. Added Picture-In-Picture (PiP) recording function 2. Add API introduction: (1) RKADK_RECORD_Single_Start (2) RKADK_RECORD_Single_Stop (3) RKADK_RECORD_SetPipAttr (4) RKADK_RECORD_FileCacheSetMode (5) RKADK_PLAYER_GetSendFrameNum (6) RKADK_PLAYER_SetVdecWaterline (7) RKADK_PLAYER_SetAoVolume

Contents

1. Overview

Rkadk provides basic and universal components such as video recording, photo capture, playback, preview, etc., simplifying the difficulty of application development and supporting the rapid development of audio and video recording-related software.

This component only supports the implementation of single-process functionality. It does not support multiple processes used simultaneously unless noted otherwise.

1.1 Version Introduction

- Version 1.x.x: Further encapsulation based on rkmedia and rockit, corresponding to the master repository.
- Version 2.x.x: Further encapsulation based on rockit, corresponding to the develop repository.

2. Video Recording

2.1 Overview

Provides basic video recording functionality, offering the following features to product:

- Creation and destruction of recording tasks
- Start and stop of recording tasks
- Manual slice of recording files
- Time-lapse recording
- Pre-recording
- Picture-In-Picture (PiP) recording

Recording tasks obtain video and audio information through the parameter module, start and stop VENC (video encoding), start and stop AENC (audio encoding), and call the encapsulation module to create recording files and write frames to the files.

Each recording task corresponds to one or more recording files. Each file must correspond to a video encoding channel, and if audio recording is needed, an audio encoding channel must be added.

Multiple recording files under the same recording task have the same recording type, the same slicing conditions, and in time-lapse recording mode, they also have the same time-lapse interval. Different recording times can be configured.

Post AI ISP: RV1106/RV1126/RV1109 chip supports noise reduction in dim light environments and intelligent enhancement processing of VI output images through AI ISP, which can still present a clearer picture without dragging shadows, low noise, and even in dim light environments.

2.2 API Reference

2.2.1 RKADK_RECORD_Create

【Description】

Create a video recording task.

【Syntax】

RKADK_S32 RKADK_RECORD_Create(RKADK_RECORD_ATTR_S *pstRecAttr, RKADK_MW_PTR *ppRecorder);

【Parameters】

Parameter name	Description	Input/output
pstRecAttr	Recording task attributes	Input
ppRecorder	Created recording task pointer	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_record.h

Library file: librkadk.so

【Notice】

- Each recording task supports a maximum of 2 recording files simultaneously.
- Each recording file must have at least 1 video stream and supports encapsulating 1 video stream and 1 audio stream simultaneously at most.
- It does not support the creation of the same task repeatedly.
- After creating a recording task, RKADK_RECORD_Start must be called to start recording.
- To enable the Picture-in-Picture feature, it must be enabled through stPipAttr when RKADK_RECORD_Create, which will then create the Picture-in-Picture pathway.

【Example】

rkadk_record_test.

【See Also】

RKADK_RECORD_Destroy.

2.2.2 RKADK_RECORD_Destroy

【Description】

Destroy video tasks.

【Syntax】

RKADK_S32 RKADK_RECORD_Destroy(RKADK_MW_PTR pRecorder);

【Parameters】

Parameter name	Description	Input/output
pRecorder	Pointer to recording task	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_record.h

Library file: librkadk.so

【Notice】

- The RKADK_RECORD_Destroy interface can only be used after creating a recording task.
- Only created recording tasks can be destroyed, and repeated destruction of the same recording task is not supported.

【Example】

rkadk_record_test.

【See Also】

RKADK_RECORD_Create

2.2.3 RKADK_RECORD_Start

【Description】

Start recording task.

【Syntax】

RKADK_S32 RKADK_RECORD_Start(RKADK_MW_PTR pRecorder);

【Parameters】

Parameter name	Description	Input/output
pRecorder	Pointer to recording task	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_record.h

Library file: librkadk.so

【Notice】

- The RKADK_RECORD_Start interface can only be used after creating a recording task.
- It is possible to restart a recording task after stopping it.

【Example】

rkadk_record_test.

【See Also】

RKADK_RECORD_Stop

2.2.4 RKADK_RECORD_Stop

【Description】

Stop recording task.

【Syntax】

RKADK_S32 RKADK_RECORD_Stop(RKADK_MW_PTR pRecorder);

【Parameters】

Parameter name	Description	Input/output
pRecorder	Pointer to recording task	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_record.h

Library file: librkadk.so

【Notice】

- The RKADK_RECORD_Stop interface can only be used after creating a recording task.
- Repeatedly stopping the same recording task is not supported.

【Example】

rkadk_record_test.

【See Also】

RKADK_RECORD_Start

2.2.5 RKADK_RECORD_Single_Start

【Description】

Start the recording task for a specified stream.

【Syntax】

RKADK_S32 RKADK_RECORD_Single_Start(RKADK_MW_PTR pRecorder, RKADK_STREAM_TYPE_E enStrmType);

【Parameters】

Parameter Name	Description	Input/Output
pRecorder	Pointer to recording task	Input
enStrmType	Stream type	Input

【Return Value】

Return Value	Description
0	Success
Other value	Failure

【Requirements】

Header file: rkadk_record.h

Library file: librkadk.so

【Notes】

- The RKADK_RECORD_Single_Start interface can only be used after creating a recording task.
- Used in conjunction with RKADK_RECORD_Single_Stop.
- Supports restarting the recording task after stopping.

【Example】

rkadk_record_test.

【Related Topics】

RKADK_RECORD_Single_Stop.

2.2.6 RKADK_RECORD_Single_Stop

【Description】

Stop the recording task for a specified stream.

【Syntax】

RKADK_S32 RKADK_RECORD_Single_Stop(RKADK_MW_PTR pRecorder, RKADK_STREAM_TYPE_EenStrmType);

【Parameters】

Parameter Name	Description	Input/Output
pRecorder	Pointer to recording task	Input
enStrmType	Stream type	Input

【Return Values】

Return Value	Description
0	Success
Other value	Failure

【Requirements】

Header file: rkadk_record.h

Library file: librkadk.so

【Notes】

- The RKADK_RECORD_Stop interface can only be used after creating a recording task.
- Use in conjunction with RKADK_RECORD_Single_Start.
- Does not support stopping the same recording task repeatedly.

【Example】

rkadk_record_test.

【Related Topics】

RKADK_RECORD_Single_Start

2.2.7 RKADK_RECORD_Reset

【Description】

Reconfigure the recording task.

【Syntax】

RKADK_S32 RKADK_RECORD_Reset(RKADK_MW_PTR *ppRecorder);

【parameters】

Parameter name	Description	Input/output
ppRecorder	Pointer to recording task	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_record.h

Library file: librkadk.so

【Notice】

- The RKADK_RECORD_Reset interface can only be used after creating a recording task.
- When switching resolution, frame rate, bit rate, encoding type, or recording type, reset the parameters of the Record module in the ini file by using the RKADK_PARAM_SetCamParam API.
- RV1126/RV1109 does not support dynamic switching of resolution and encoding type. You need to RKADK_RECORD_Destroy first, then configure the new resolution or encoding type, and finally RKADK_RECORD_Create again.
- If Photo and Record share the VI channel, both the resolution of Record and Photo need to be switched simultaneously to avoid VI and VENC resolution mismatch.

【Example】

rkadk_record_test.

【See Also】

None

2.2.8 RKADK_RECORD_ManualSplit

【Description】

Manually split video files.

【Syntax】

RKADK_S32 RKADK_RECORD_ManualSplit(RKADK_MW_PTR pRecorder,
RKADK_REC_MANUAL_SPLIT_ATTR_S *pstSplitAttr);

【Parameters】

Parameter name	Description	Input/output
pRecorder	Pointer to recording task	Input
pstSplitAttr	Manual split attribute parameters	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_record.h

Library file: librkadk.so

【Notice】

- The RKADK_RECORD_ManualSplit interface can only be used after creating a recording task.
- Supports repeated manual splitting of video files when the manual splitting of video files has not ended.

【Example】

rkadk_record_test.

【See Also】

No

2.2.9 RKADK_RECORD_GetAencChn

【Description】

Get the video AENC channel ID.

【Syntax】

RKADK_S32 RKADK_RECORD_GetAencChn();

【Return value】

AENC channel ID used for recording.

【Requirement】

Header file: rkadk_record.h

Library file: librkadk.so

2.2.10 RKADK_GetThmInMp4

【Description】

Get thumbnail data from MP4 files.

【Syntax】

RKADK_S32 RKADK_GetThmInMp4(RKADK_U32 u32CamId, RKADK_CHAR *pszFileName, RKADK_U8 *pu8Buf,
RKADK_U32 *pu32Size);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
pszFileName	MP4 file path	Input
pu8Buf	Input: thumbnail data storage pointer, output: actual thumbnail data	Input/Output
pu32Size	Input: pu8Buf length, output: actual thumbnail data length	Input/Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_thumb.h

Library file: librkadk.so

【Example】

rkadk_thumb_test

【See Also】

None

2.2.11 RKADK_GetThmInMp4Ex

【Description】

MP4 obtains thumbnail extension interface, obtains thumbnail data from MP4 files, and supports specifying the type and resolution of output thumbnails. It must be used together with RKADK_ThmBufFree.

【Syntax】

RKADK_S32 RKADK_GetThmInMp4Ex(RKADK_U32 u32CamId, RKADK_CHAR *pszFileName, RKADK_THUMB_ATTR_S *pstThumbAttr);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
pszFileName	MP4 file path	Input
pstThumbAttr	Thumbnail attribute structure pointer	Input/output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_thumb.h

Library file: librkadk.so

【Example】

rkadk_thumb_test

【See Also】

RKADK_ThmBufFree

2.2.12 RKADK_ThmBufFree

【Description】

Release the memory requested by RKADK_GetThmInMp4Ex. It must be used together with RKADK_GetThmInMp4Ex.

【Syntax】

RKADK_S32 RKADK_ThmBufFree(RKADK_THUMB_ATTR_S *pstThumbAttr);

【Parameters】

Parameter name	Description	Input/output
pstThumbAttr	Thumbnail attribute structure pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_thumb.h

Library file: librkadk.so

【Example】

rkadk_thumb_test

【See Also】

RKADK_GetThmInMp4Ex

2.2.13 RKADK_RECORD_SetRotation

【Description】

Set Record rotation.

【Syntax】

RKADK_S32 RKADK_RECORD_SetRotation(RKADK_MW_PTR pRecorder, ROTATION_E enRotation, RKADK_STREAM_TYPE_E enStreamType);

【Parameters】

Parameter name	Description	Input/output
pRecorder	Pointer to recording task	Input
enRotation	Rotation angle	Input
enStreamType	Stream type	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_record.h

Library file: librkadk.so

【Example】

rkadk_record_test

2.2.14 RKADK_RECORD_ToggleMirror

【Description】

Set Record Mirror.

【Syntax】

RKADK_S32 RKADK_RECORD_ToggleMirror(RKADK_MW_PTR pRecorder, RKADK_STREAM_TYPE_E enStrmType, bool mirror);

【Parameters】

Parameter name	Description	Input/output
pRecorder	Pointer to recording task	Input
enStreamType	Stream type	Input
mirror	Whether to enable mirror	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_record.h

Library file: librkadk.so

2.2.15 RKADK_RECORD_ToggleFlip

【Description】

Set Record Flip.

【Syntax】

RKADK_S32 RKADK_RECORD_ToggleFlip(RKADK_MW_PTR pRecorder, RKADK_STREAM_TYPE_E enStrmType, bool flip);

【Parameters】

Parameter name	Description	Input/output
pRecorder	Pointer to recording task	Input
enStreamType	Stream type	Input
flip	Whether to enable flip	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_record.h

Library file: librkadk.so

2.2.16 RKADK_RECORD_FileCacheInit

【Description】

File Cache initialization is aimed at making file writing smoother. Enabling File Cache means that files are written to the storage device through DirectIO, and the kernel's cache mechanism is not effective.

【Syntax】

RKADK_S32 RKADK_RECORD_FileCacheInit(FILE_CACHE_ARG *pstFileCacheAttr)

【Parameters】

Parameter name	Description	Input/output
pstFileCacheAttr	File Cache attribute pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_record.h

Library file: librkadk.so

【Notice】

- File Cache is not enabled by default.
- To enable File Cache, the file_cache_env environment variable must be set

```
export file_cache_env=1
```

- If File Cache is enabled, this interface must be called before RKADK_RECORD_Create, and only needs to be called once, otherwise there is no need to call it.
- Enabling File Cache will need additional memory, and the memory size is determined by u32TotalCache.
- This interface must be used in pair with RKADK_RECORD_FileCacheDeInit

【Example】

rkadk_record_test

2.2.17 RKADK_RECORD_FileCacheDeInit

【Description】

De-initialization File Cache

【Syntax】

RKADK_S32 RKADK_RECORD_FileCacheDeInit();

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_record.h

Library file: librkadk.so

【Notice】

- It must be used together with RKADK_RECORD_FileCacheInit.
- If File Cache is enabled, this interface needs to be called after RKADK_RECORD_Destroy, otherwise there is no need to call it.

rkadk_record_test

2.2.18 RKADK_RECORD_FileCacheSetMode

【Description】

Set the recording type for File Cache.

【Syntax】

void RKADK_RECORD_FileCacheSetMode(RKADK_REC_TYPE_E enRecType);

【Parameters】

Parameter Name	Description	Input/Output
enRecType	Recording type	Input

【Requirements】

Header file: rkadk_record.h

Library file: librkadk.so

【Notes】

- After enabling File Cache, when switching recording types, this API must be called to set a new recording type for File Cache.

rkadk_aov_record_test

2.2.19 RKADK_MEDIA_EnablePostIsp

【Description】

Enable post AI ISP, in addition to the interface, you can also enable post AI ISP by configuring post_aiisp in sensor ini.

【Syntax】

RKADK_S32 RKADK_MEDIA_EnablePostIsp(RKADK_U32 u32CamId, RKADK_STREAM_TYPE_E enStrmType, RKADK_POST_ISP_ATTR_S *pstPostIspAttr);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
enStrmType	Stream type	Input
pstPostIspAttr	Post AI ISP attribute pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_media_comm.h

Library file: librkadk.so

2.2.20 RKADK_MEDIA_SetPostIspAttr

【Description】

Dynamically set post AI ISP attributes.

【Syntax】

RKADK_S32 RKADK_MEDIA_SetPostIspAttr(RKADK_U32 u32CamId, RKADK_STREAM_TYPE_E enStrmType, bool bEnable, RKADK_POST_ISP_ATTR_S *pstPostIspAttr);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
enStrmType	Stream type	Input
bEnable	Post AI ISP function switch, dynamic attribute	Input
pstPostIspAttr	Post AI ISP attribute pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_media_comm.h

Library file: librkadk.so

2.2.21 RKADK_RECORD_SetPipAttr

【Description】

Dynamically set picture-in-picture (PiP) attributes.

【Syntax】

RKADK_S32 RKADK_RECORD_SetPipAttr(RKADK_MW_PTR pRecorder, RKADK_PIP_ATTR_S *pstPipAttr);

【Parameters】

Parameter Name	Description	Input/Output
pRecorder	Pointer to recording task	Input
pstPipAttr	Picture-in-picture attributes	Input

【Return Value】

Return Value	Description
0	Success
Other value	Failure

【Requirements】

Header file: rkadk_record.h

Library file: librkadk.so

【Note】

- To enable the Picture-in-Picture feature, it must be enabled through stPipAttr when RKADK_RECORD_Create, which will then create the Picture-in-Picture pathway.

2.3 Type of Data

The video recording module mainly provides the following data types:

RKADK_MW_PTR: Pointer to recording task

RECORD_FILE_NUM_MAX: The maximum number of files recorded simultaneously by a single recording task

RKADK_MUXER_EVENT_E: Recording event enumeration type

RKADK_MUXER_FILE_EVENT_INFO_S: File-related event information structure

RKADK_MUXER_EVENT_INFO_S: Recording event information structure

RKADK_REC_EVENT_CALLBACK_FN: Event callback function pointer

RKADK_REC_TYPE_E: Recording type enumeration

RKADK_REC_REQUEST_FILE_NAMES_FN: Request video file name function pointer

RKADK_RECORD_ATTR_S: Recording task attribute structure

RKADK_MUXER_MANUAL_SPLIT_TYPE_E: Manually split enumeration types

RKADK_REC_MANUAL_SPLIT_ATTR_S: Manually split attribute structure

FILE_CACHE_ARG: File Cache attribute structure

RKADK_POST_ISP_ATTR_S: Post AI ISP attribute structure

RKADK_MOUNT_SDCARD_FN: SD card mounting function pointer

RKADK_AOV_ATTR_S: AOV attribute structure

RKADK_PIP_ATTR_S: Picture-in-Picture attribute structure

2.3.1 Public Data Types

【Description】

Definition of basic data types.

【Definition】

```
typedef unsigned char RKADK_U8;
typedef unsigned short RKADK_U16;
typedef unsigned int RKADK_U32;

typedef signed char RKADK_S8;
typedef short RKADK_S16;
typedef int RKADK_S32;

typedef unsigned long RKADK_UL;
typedef signed long RKADK_SL;

typedef float RKADK_FLOAT;
typedef double RKADK_DOUBLE;

#ifndef _M_IX86
typedef unsigned long long RKADK_U64;
typedef long long RKADK_S64;
#else
typedef unsigned __int64 RKADK_U64;
typedef __int64 RKADK_S64;
#endif

typedef char RKADK_CHAR;
#define RKADK_VOID void

typedef unsigned int RKADK_HANDLE;

typedef RKADK_VOID *RKADK_MW_PTR;

typedef char (*ARRAY_FILE_NAME)[RKADK_MAX_FILE_PATH_LEN];

typedef enum {
  RKADK_FALSE = 0,
  RKADK_TRUE = 1,
} RKADK_BOOL;

#ifndef NULL
#define NULL 0L
#endif

#define RKADK_NULL 0L
#define RKADK_SUCCESS 0
#define RKADK_FAILURE (-1)
```

2.3.2 RKADK_MW_PTR

【Description】

Define task pointer

【Definition】

```
typedef RKADK_VOID *RKADK_MW_PTR;
```

2.3.3 RKADK_MAX_SENSOR_CNT

【Description】

Define the maximum number of sensors supported, which can be adjusted according to the actual situation

【Definition】

```
#define RKADK_MAX_SENSOR_CNT 3
```

2.3.4 RECORD_FILE_NUM_MAX

【Description】

Define the maximum number of files recorded simultaneously by a single recording task.

【Definition】

```
#define RECORD_FILE_NUM_MAX 2
```

2.3.5 RKADK_MUXER_EVENT_E

【Description】

Define the recording event enumeration type.

【Definition】

```
typedef enum rkMUXER_EVENT_E {
  RKADK_MUXER_EVENT_STREAM_START = 0,
  RKADK_MUXER_EVENT_STREAM_STOP,
  RKADK_MUXER_EVENT_FILE_BEGIN,
  RKADK_MUXER_EVENT_FILE_END,
  RKADK_MUXER_EVENT_MANUAL_SPLIT_END,
  RKADK_MUXER_EVENT_ERR_CREATE_FILE_FAIL,
  RKADK_MUXER_EVENT_ERR_WRITE_FILE_FAIL,
  RKADK_MUXER_EVENT_FILE_WRITING_SLOW,
  RKADK_MUXER_EVENT_ERR_CARD_NONEXIST,
  RKADK_MUXER_EVENT_BUTT
} RKADK_MUXER_EVENT_E;
```

【Members】

Member Name	Description
RKADK_MUXER_EVENT_STREAM_START	Start recording
RKADK_MUXER_EVENT_STREAM_STOP	Stop recording
RKADK_MUXER_EVENT_FILE_BEGIN	Start recording a new file
RKADK_MUXER_EVENT_FILE_END	End of file recording
RKADK_MUXER_EVENT_MANUAL_SPLIT_END	End of manual split file recording
RKADK_MUXER_EVENT_ERR_CREATE_FILE_FAIL	Reserved
RKADK_MUXER_EVENT_ERR_WRITE_FILE_FAIL	Failed to write file
RKADK_MUXER_EVENT_FILE_WRITING_SLOW	Writing files is slow
RKADK_MUXER_EVENT_ERR_CARD_NONEXIST	SD card does not exist

【Related data types and interfaces】

RKADK_MUXER_EVENT_INFO_S

2.3.6 RKADK_MUXER_FILE_EVENT_INFO_S

【Description】

Define file-related event information structure.

【Definition】

```
typedef struct {
  RK_CHAR asFileName[RKADK_MUXER_FILE_NAME_LEN];
  RK_U32 u32Duration; // ms
} RKADK_MUXER_FILE_EVENT_INFO_S;
```

【Members】

Member Name	Description
asFileName	File name
u32Duration	Actual recorded file duration

【Related data types and interfaces】

RKADK_MUXER_EVENT_INFO_S

2.3.7 RKADK_MUXER_EVENT_INFO_S

【Description】

Define the recording event information structure.

【Definition】

```
typedef struct {
  RKADK_MUXER_EVENT_E enEvent;
  union {
    RKADK_MUXER_FILE_EVENT_INFO_S stFileInfo;
    RKADK_MUXER_ERROR_EVENT_INFO_S stErrorInfo;
  } unEventInfo;
} RKADK_MUXER_EVENT_INFO_S;
```

【Members】

Member Name	Description
enEvent	Recording event type
stFileInfo	File event information
stErrorInfo	Error event information (Reserved)

【Related data types and interfaces】

RKADK_MUXER_EVENT_E

RKADK_MUXER_FILE_EVENT_INFO_S

2.3.8 RKADK_REC_EVENT_CALLBACK_FN

【Description】

Define the video event callback function pointer.

【Definition】

```
typedef RKADK_MUXER_EVENT_CALLBACK_FN RKADK_REC_EVENT_CALLBACK_FN;

typedef RKADK_VOID (*RKADK_MUXER_EVENT_CALLBACK_FN)(RKADK_MW_PTR pRecorder, const
RKADK_MUXER_EVENT_INFO_S *pstEventInfo);
```

【Related data types and interfaces】

RKADK_MW_PTR

RKADK_MUXER_EVENT_INFO_S

2.3.9 RKADK_REC_TYPE_E

【Description】

Define the enumeration of recording types.

【Definition】

```
typedef enum {
  RKADK_REC_TYPE_NORMAL = 0, /* normal record */
  RKADK_REC_TYPE_LAPSE,      /* time lapse record */
  RKADK_REC_TYPE_BUTT
} RKADK_REC_TYPE_E;
```

【Members】

Member Name	Description
RKADK_REC_TYPE_NORMAL	Normal recording
RKADK_REC_TYPE_LAPSE	Time-lapse video

【Related data types and interfaces】

None

2.3.10 RKADK_REC_REQUEST_FILE_NAMES_FN

【Description】

Define the callback function pointer to request the video file name.

【Definition】

```
typedef RKADK_S32 (*RKADK_REC_REQUEST_FILE_NAMES_FN)(RKADK_MW_PTR pRecorder,
RKADK_U32 u32FileCnt, RKADK_CHAR(*paszFilename)[RKADK_MAX_FILE_PATH_LEN]);
```

【Members】

Member Name	Description
pRecorder	Pointer to recording task
u32FileCnt	Number of requested file names
paszFilename	Store file name buffer

【Related data types and interfaces】

RKADK_MW_PTR

RKADK_RECORD_ATTR_S

2.3.11 RKADK_MOUNT_SDCARD_FN

【Description】

Define the SD card mounting function pointer.

【Definition】

```
typedef int (*RKADK_MOUNT_SDCARD_FN)(void);
```

【Related Data Types and Interfaces】

RKADK_RECORD_ATTR_S

2.3.12 RKADK_AOV_ATTR_S

2.3.13 RKADK_RECORD_ATTR_S

【Description】

Define the recording task attribute structure.

【Definition】

```
typedef struct {
  RKADK_S32 s32CamID;                                  /* Camera ID */
  RKADK_U32 u32FragKeyFrame;
  RKADK_REC_REQUEST_FILE_NAMES_FN pfnRequestFileNames; /* rec callbak */
  RKADK_REC_EVENT_CALLBACK_FN pfnEventCallback;        /* event callbak */
  RKADK_AOV_ATTR_S stAovAttr;
  RKADK_POST_ISP_ATTR_S *pstPostIspAttr;
  RKADK_MOUMNT_SDCARD_FN pfnMountSdcard;
  RKADK_PIP_ATTR_S stPipAttr[RECORD_FILE_NUM_MAX];
} RKADK_RECORD_ATTR_S;
```

【Members】

Member Name	Description
s32CamID	Camera ID
u32FragKeyFrame	Whether the video file is sliced into I frames
pfnRequestFileNames	Request file name function pointer
pfnEventCallback	Recording event callback function pointer
stAovAttr	AOV (Always On Video) attributes
pstPostIspAttr	Post AI ISP attributes
pfnMountSdcard	Sdcard mount function pointer
stPipAttr	Picture-in-picture attributes

【Related data types and interfaces】

RKADK_REC_EVENT_CALLBACK_FN

RKADK_REC_REQUEST_FILE_NAMES_FN

RKADK_MOUMNT_SDCARD_FN

RKADK_AOV_ATTR_S

RKADK_POST_ISP_ATTR_S

RKADK_PIP_ATTR_S

RKADK_RECORD_Create

2.3.14 RKADK_MUXER_MANUAL_SPLIT_TYPE_E

【Description】

Define manual split type.

【Definition】

```
typedef enum {
  MUXER_PRE_MANUAL_SPLIT,      /* pre manual split type */
  MUXER_NORMAL_MANUAL_SPLIT,   /* normal manual split type */
} RKADK_MUXER_MANUAL_SPLIT_ATTR_S;
```

【Members】

Member Name	Description
MUXER_PRE_MANUAL_SPLIT	Manually split video files and pre-record
MUXER_NORMAL_MANUAL_SPLIT	Manually split video files

【Related data types and interfaces】

RKADK_REC_MANUAL_SPLIT_ATTR_S

2.3.15 RKADK_REC_MANUAL_SPLIT_ATTR_S

【Description】

Define manually split attribute structure.

【Definition】

```
typedef MUXER_MANUAL_SPLIT_ATTR_S RKADK_REC_MANUAL_SPLIT_ATTR_S;

typedef struct {
  RKADK_MUXER_MANUAL_SPLIT_TYPE_E enManualType;               /* maual split type
*/
  RKADK_U32 u32DurationSec; /* file duration of manual split file */
} MUXER_MANUAL_SPLIT_ATTR_S;
```

【Members】

Member Name	Description
enManualType	Manually split type
u32DurationSec	Manually split video file duration
【Related data types and interfaces】	

RKADK_MUXER_MANUAL_SPLIT_TYPE_E

RKADK_RECORD_ManualSplit

2.3.16 FILE_CACHE_ARG

【Description】

Define the File Cache attribute structure.

【Definition】

```
typedef struct _FILE_CACHE_ARG {
  const char *sdcard_path;
  int write_cache;  /* write cache size(byte), default 1M */
  int total_cache;  /* total cache size(byte), default 10M */
  FILE_WRITE_THREAD_ARG write_thread_arg;
  FILE_SDCARD_ARG sdcard_arg;
} FILE_CACHE_ARG;
```

【Members】

Member Name	Description
sdcard_path	SD card mounting path
write_cache	Cache size for each write file
total_cache	Total Cache size
write_thread_arg	Write thread attribute
sdcard_arg	sdcard attribute

【Related data types and interfaces】

RKADK_RECORD_FileCacheInit

FILE_WRITE_THREAD_ARG

FILE_SDCARD_ARG

2.3.17 FILE_WRITE_THREAD_ARG

【Description】

Define the write thread attribute structure.

【Definition】

```
typedef enum _FILE_SCHED_POLICY {
  FILE_SCHED_OTHER = 0,
  FILE_SCHED_BATCH,
  FILE_SCHED_IDLE,
  FILE_SCHED_FIFO,      /* sched_priority[1, 99] */
  FILE_SCHED_RR,        /* sched_priority[1, 99] */
} FILE_SCHED_POLICY;

typedef struct _FILE_WRITE_THREAD_ARG {
  FILE_SCHED_POLICY sched_policy;
  int priority; /* SCHED_OTHER/SCHED_IDLE/SCHED_BATCH inoperative */
} FILE_WRITE_THREAD_ARG;
```

【Members】

Member Name	Description
sched_policy	Thread schedule policy
priority	Thread priority

【Related data types and interfaces】

RKADK_RECORD_FileCacheInit

FILE_CACHE_ARG

2.3.18 FILE_SDCARD_ARG

【Description】

Define the sdcard attribute structure.

【Definition】

```
typedef struct _FILE_SDCARD_ARG {
    FILE_CACHE_SDCARD_LOCK lock;
    FILE_CACHE_SDCARD_UNLOCK unlock;
    FILE_CACHE_MOUMNT_SDCARD mount_sdcard;
    FILE_CACHE_UMOUMNT_SDCARD umount_sdcard;
} FILE_SDCARD_ARG;
```

【Members】

Member Name	Description
lock	sdcard operation lock function
unlock	sdcard operation unlock function
mount_sdcard	sdcard mount function
umount_sdcard	sdcard unmount function

【Related data types and interfaces】

FILE_CACHE_ARG

2.3.19 RKADK_POST_ISP_ATTR_S

【Description】

Define the post AI ISP attribute structure.

【Definition】

```
typedef struct {
  AIISP_CALLBACK_FUNC_S stAiIspCallback;      /* post isp callback function */
  const RK_CHAR        *pModelFilePath;       /* post isp model file path   */
  RK_U32                u32FrameBufCnt;       /* RW; frame buffer cnt    */
} RKADK_POST_ISP_ATTR_S;
```

【Members】

Member Name	Description
stAiIspCallback	AI NR parameter update callback function
pModelFilePath	Post AI ISP model file path
u32FrameBufCnt	frame buffer cnt, default 1

【Related data types and interfaces】

RKADK_MEDIA_EnablePostIsp

RKADK_MEDIA_SetPostIspAttr

2.3.20 RKADK_AOV_ATTR_S

【Description】

Define the AOV attribute structure.

【Definition】

```
typedef struct {
  RKADK_ISP_WAKE_UP_PAUSE_FN pfnSingleFrame;
  RKADK_ISP_WAKE_UP_RESUME_FN pfnMultiFrame;
} RKADK_AOV_ATTR_S;
```

【Members】

Member Name	Description
pfnSingleFrame	Pointer to the single-frame mode function
pfnMultiFrame	Pointer to the multi-frame mode function

【Related Data Types and Interfaces】

RKADK_RECORD_Create

2.3.21 RKADK_PIP_ATTR_S

【Description】

Define the structure for picture-in-picture attributes.

【Definition】

```
typedef struct {
  RKADK_BOOL bEnablePip;                    /* enable picture-in-picture */
  RKADK_U32 u32AvsGrpId;                    /* AVS group id [0,
AVS_MAX_GRP_NUM)] */
  RKADK_U32 u32AvsBufCnt;                   /* default 2, min value 2 */
  RKADK_U32 u32SubCamId;                    /* subwindow camera id */
  RKADK_STREAM_TYPE_E enSubStreamType;      /* subwindow stream type */
  RKADK_RECT_S stSubRect;                   /* The subwindow is based on the
display area of the main window */
} RKADK_PIP_ATTR_S;
```

【Members】

Member Name	Description
bEnablePip	Enable picture-in-picture
u32AvsGrpId	AVS Group Id used for picture-in-picture
u32AvsBufCnt	Maximum number of target image buffers for AVS
u32SubCamId	Subwindow Camera Id
enSubStreamType	Subwindow stream type
stSubRect	Display area of the subwindow within the main window

【Related Data Types and Interfaces】

RKADK_RECORD_Create

RKADK_RECORD_SetPipAttr

3. AOV(Always On Video)

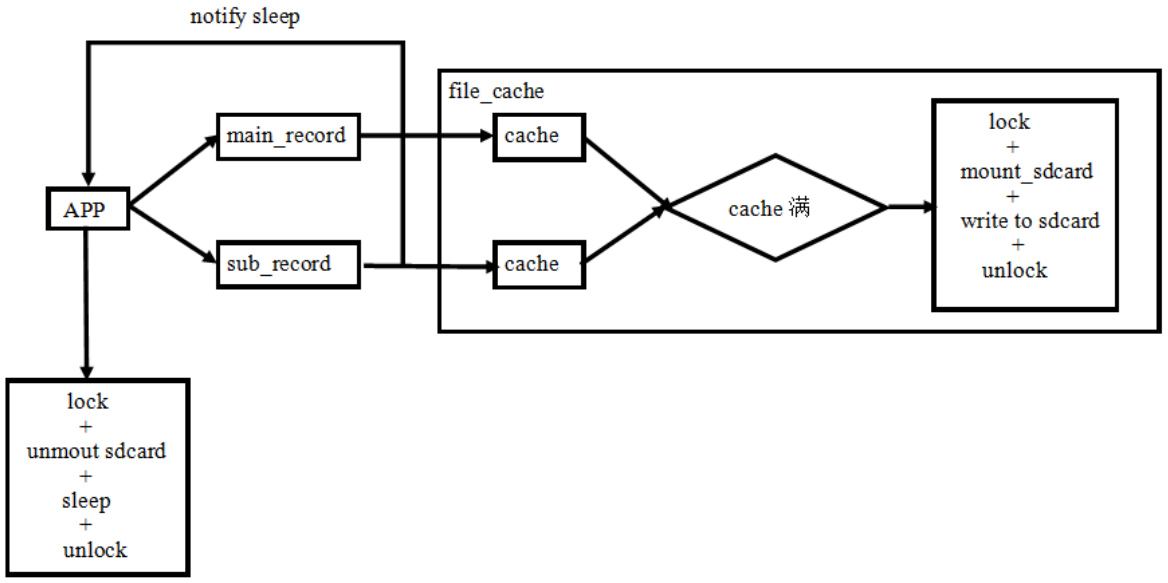
3.1 Overview

AOV (Always On Video) is a low frame rate continuous recording based on sleep&wake-up. It can switch back to the normal frame rate mode when there is an event, such as AI detection, PIR trigger, WIFI wake-up, etc.

It is recommended to enable the file cache function when using AOV. For file cache usage, please refer to the RKADK_RECORD_FileCacheInit and RKADK_RECORD_FileCacheDeInit APIs in the recording chapter of this document.

In order to reduce power consumption in the AOV time-lapse recording mode, the sdcard needs to be powered off after sleeping, and the sdcard must be unmounted before the application goes to sleep. When the file cache is enabled, the file cache will try to mount the sdcard when writing files, and perform file write protection, to ensure that the sdcard will not be unmounted during the file writing process.

The AOV process is shown in the figure below:



3.2 API Reference

3.2.1 RKADK_AOV_Init

【Description】

AOV initialization.

【Grammar】

int RKADK_AOV_Init(RKADK_AOV_ARG_S *pstAovAttr);

【Parameter】

Parameter name	Description	Input/output
pstAovAttr	AOV attribute pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_aov.h

Library file: librkadk.so

【Notice】

- It Must be used in conjunction with RKADK_AOV_DeInit.
- Repeated initialization is not supported.

【Example】

rkadk_aov_record_test

【Related topic】

RKADK_AOV_DeInit

3.2.2 RKADK_AOV_DeInit

【Description】

AOV deinitialization.

【Grammar】

int RKADK_AOV_DeInit();

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_aov.h

Library file: librkadk.so

【Notice】

- It Must be used in conjunction with RKADK_AOV_Init.
- Repeated deinitialization is not supported.

【Example】

rkadk_aov_record_test

【Related topic】

RKADK_AOV_Init

3.2.3 RKADK_AOV_SetSuspendTime

【Description】

Set the AOV scheduled sleep time in ms.

【Grammar】

int RKADK_AOV_SetSuspendTime(int u32WakeupSuspendTime);

【Parameter】

Parameter name	Description	Input/output
u32WakeupSuspendTime	Sleep time	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_aov.h

Library file: librkadk.so

【Example】

rkadk_aov_record_test

3.2.4 RKADK_AOV_EnterSleep

【Description】

Enter AOV.

【Grammar】

int RKADK_AOV_EnterSleep();

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_aov.h

Library file: librkadk.so

【Example】

rkadk_aov_record_test

3.2.5 RKADK_AOV_WakeupLock

【Description】

Lock AOV related operations.

【Grammar】

void RKADK_AOV_WakeupLock();

【Requirement】

Header file: rkadk_aov.h

Library file: librkadk.so

【Notice】

- Must be used in conjunction with RKADK_AOV_WakeupUnlock.

【Example】

rkadk_aov_record_test

【Related topic】

RKADK_AOV_WakeupUnlock

3.2.6 RKADK_AOV_WakeupUnlock

【Description】

Unlocked AOV related operations.

【Grammar】

void RKADK_AOV_WakeupUnlock();

【Requirement】

Header file: rkadk_aov.h

Library file: librkadk.so

【Notice】

- Must be used in conjunction with RKADK_AOV_WakeupLock.

【Example】

rkadk_aov_record_test

【Related topic】

RKADK_AOV_WakeupLock

3.3 Type of data

The AOV module mainly provides the following data types:

RKADK_AOV_ARG_S: AOV attribute structure

RKADK_AOV_NOTIFY_CALLBACK: AOV notification callback function pointer

3.3.1 RKADK_AOV_ARG_S

【Definition】

Define the AOV attribute structure.

【definition】

```
typedef struct {
   RKADK_AOV_NOTIFY_CALLBACK pfnNotifyCallback;
} RKADK_AOV_ARG_S;
```

【Members】

Member Name	Description
pfnNotifyCallback	AOV notification callback function pointer

[Related data types and interfaces]

RKADK_AOV_Init

RKADK_AOV_NOTIFY_CALLBACK

3.3.2 RKADK_AOV_NOTIFY_CALLBACK

【Definition】

Define the AOV notification callback function pointer.

【Definition】

```
typedef void (*RKADK_AOV_NOTIFY_CALLBACK)(RKADK_AOV_EVENT_E enEvent, void *msg);
```

【Members】

Member Name	Description
enEvent	AOV notification event enumeration
msg	Reserve

[Related data types and interfaces]

RKADK_AOV_ARG_S

RKADK_AOV_EVENT_E

3.3.3 RKADK_AOV_EVENT_E

【Definition】

Defines the AOV notification event enumeration.

【definition】

```
typedef enum {
   RKADK_AOV_ENTER_SLEEP = 0,
   RKADK_AOV_EVENT_BUTT
} RKADK_AOV_EVENT_E;
```

【Members】

Member Name	Description
RKADK_AOV_ENTER_SLEEP	Enter AOV sleep

[Related data types and interfaces]

RKADK_AOV_NOTIFY_CALLBACK

4. Taking Photos

4.1 Overview

Provides basic snapshot functionality, offering JPEG encapsulation for capturing photos. The features include:

- Single snapshot
- Multiple snapshots
- JPEG Slice: Encodes VI data into JPEG in segments, which can save memory usage. When enlarging the JPEG, use interpolation algorithms to enlarge VI data in segments, and then encode it into JPEG in segments.

Notice about JPEG Slice:

- Segment height requires alignment to 16 pixels.
- Considering the stitching effect of final images, it is recommended to keep the aspect ratio of the segmented image as consistent as possible with the original images.
- The more segments, the less memory required, but the image generation time increases. Conversely, when the number of segments decreases, the required memory increases, but the image generation speed accelerates; the actual number of segments depends on the actual situation.

4.2 API Reference

4.2.1 RKADK_PHOTO_Init

【Description】

The photo taking task is initialized.

【Syntax】

RKADK_S32 RKADK_PHOTO_Init(RKADK_PHOTO_ATTR_S *pstPhotoAttr, RKADK_MW_PTR *ppHandle);

【Parameters】

Parameter name	Description	Input/output
pstPhotoAttr	Photo task attribute pointer	Input
ppHandle	Created photo task pointer	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_photo.h

Library file: librkadk.so

【Notice】

- Repeated initialization is not supported.

【Example】

rkadk_photo_test

【See Also】

RKADK_PHOTO_DeInit

4.2.2 RKADK_PHOTO_DeInit

【Description】

Deinitialize the photo taking task.

【Syntax】

RKADK_S32 RKADK_PHOTO_DeInit(RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Pointer to photo taking task	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_photo.h

Library file: librkadk.so

【Notice】

- Repeated deinitialization is not supported.

【Example】

rkadk_photo_test

【See Also】

RKADK_PHOTO_DeInit

4.2.3 RKADK_PHOTO_TakePhoto

【Description】

Take Photos.

【Syntax】

RKADK_S32 RKADK_PHOTO_TakePhoto(RKADK_MW_PTR pHandle, RKADK_TAKE_PHOTO_ATTR_S *pstAttr);

【Parameters】

Parameter name	Description	Input/output
PHandle	Pointer to photo taking task	Input
pstAttr	Photo attributes	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_photo.h

Library file: librkadk.so

【Notice】

- The RKADK_PHOTO_TakePhoto interface can only be used after the recording task is initialized.

【Example】

rkadk_photo_test

【See Also】

No

4.2.4 RKADK_PHOTO_Reset

【Description】

Reconfigure the photo taking task.

【Syntax】

RKADK_S32 RKADK_PHOTO_Reset(RKADK_MW_PTR *ppHandle);

【Parameters】

Parameter name	Description	Input/output
ppHandle	Pointer to photo taking task	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_photo.h

Library file: librkadk.so

【Notice】

- The RKADK_PHOTO_Reset interface can only be used after initializing the recording task.
- When switching resolution, reset the resolution of the Photo module in the ini file by using the RKADK_PARAM_SetCamParam API.
- RV1126/RV1109 does not support dynamic resolution switching. You need to RKADK_PHOTO_DeInit first, then configure the new resolution, and finally RKADK_PHOTO_Init again.
- If Photo and Record share the VI channel, both the resolution of Record and Photo need to be switched simultaneously to avoid VI and VENC resolution mismatch.

【Example】

rkadk_photo_test

【See Also】

4.2.5 RKADK_PHOTO_GetThmInJpg

【Description】

Get thumbnail data from a JPG file.

【Syntax】

RKADK_S32 RKADK_PHOTO_GetThmInJpg(RKADK_U32 u32CamId, RKADK_CHAR *pszFileName,
RKADK_JPG_THUMB_TYPE_E eThmType,
RKADK_U8 *pu8Buf, RKADK_U32 *pu32Size);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
pszFileName	JPG file path	Input
eThmType	Thumbnail Type	Input
pu8Buf	Input: thumbnail data storage pointer, output: actual thumbnail data	Input/Output
pu32Size	Input: length of pu8Buf , output: length of actual thumbnail data	Input/Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_photo.h

Library file: librkadk.so

【Example】

rkadk_thumb_test

【See Also】

No.

4.2.6 RKADK_PHOTO_GetThmInJpgEx

【Description】

JPG Thumbnail obtaining extended Interface: get thumbnail data from a JPG file, supporting the specification of the output thumbnail type and resolution. It must be used together with RKADK_PHOTO_ThumbBufFree.

【Syntax】

RKADK_S32 RKADK_PHOTO_GetThmInJpgEx(RKADK_U32 u32CamId, RKADK_CHAR *pszFileName,
RKADK_JPG_THUMB_TYPE_E eThmType,
RKADK_THUMB_ATTR_S *pstThumbAttr);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
pszFileName	JPG file path	Input
eThmType	Thumbnail Type	Input
pstThumbAttr	Thumbnail attribute structure pointer	Input/output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_photo.h

Library file: librkadk.so

【Example】

rkadk_thumb_test

【See Also】

RKADK_PHOTO_ThumbBufFree

4.2.7 RKADK_PHOTO_ThumbBufFree

【Description】

Release the memory requested by RKADK_PHOTO_GetThmInJpgEx. It must be used together with RKADK_PHOTO_GetThmInJpgEx.

【Syntax】

RKADK_S32 RKADK_PHOTO_ThumbBufFree(RKADK_THUMB_ATTR_S *pstThumbAttr);

【Parameters】

Parameter name	Description	Input/output
pstThumbAttr	Thumbnail attribute structure pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_photo.h

Library file: librkadk.so

【Example】

rkadk_thumb_test

【See Also】

RKADK_PHOTO_GetThmInJpgEx

4.2.8 RKADK_MEDIA_SetVencRotation

【Description】

Set VENC channel rotation.

【Syntax】

RKADK_S32 RKADK_MEDIA_SetVencRotation(RKADK_U32 u32CamId, ROTATION_E enRotation, RKADK_STREAM_TYPE_E enStreamType);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
enRotation	Rotation angle	Input
enStreamType	Stream type	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_media_comm.h

Library file: librkadk.so

4.2.9 RKADK_MEDIA_ToggleVencMirror

【Description】

Set VENC channel Mirror.

【Syntax】

RKADK_S32 RKADK_MEDIA_ToggleVencMirror(RKADK_U32 u32CamId, RKADK_STREAM_TYPE_E enStrmType, bool mirror);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
enStreamType	Stream type	Input
mirror	Whether to enable mirror	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_media_comm.h

Library file: librkadk.so

4.2.10 RKADK_MEDIA_ToggleVencFlip

【Description】

Set VENC channel Flip.

【Syntax】

RKADK_S32 RKADK_MEDIA_ToggleVencFlip(RKADK_U32 u32CamId, RKADK_STREAM_TYPE_E enStrmType, bool flip);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
enStreamType	Stream type	Input
flip	Whether to enable flip	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_media_comm.h

Library file: librkadk.so

4.3 Type of Data

The camera module mainly provides the following data types:

RKADK_PHOTO_TYPE_E: Photo type enumeration

RKADK_PHOTO_SINGLE_ATTR_S: Single snapshot attribute structure

RKADK_PHOTO_MULTIPLE_ATTR_S: Multi snapshots attribute structure

RKADK_PHOTO_RECV_DATA_S: Photo data structure

RKADK_PHOTO_DATA_RECV_FN_PTR: Photo data receiving function pointer

RKADK_TAKE_PHOTO_ATTR_S: Photo attribute structure

RKADK_PHOTO_ATTR_S: Photo task attribute structure

RKADK_JPG_THUMB_TYPE_E: JPG thumbnail type enumeration

RKADK_THUMB_TYPE_E: Output thumbnail type enumeration

RKADK_THUMB_ATTR_S: Thumbnail attribute structure

ROTATION_E: Rotation type enumeration

RKADK_PHOTO_FMT_CHANGE_S: Format conversion parameter structure, used for FBC0/NV16 format conversion

4.3.1 RKADK_PHOTO_TYPE_E

【Description】

Define the photo type enumeration.

【Definition】

```
typedef enum {
  RKADK_PHOTO_TYPE_SINGLE = 0,
  RKADK_PHOTO_TYPE_MULTIPLE,
  RKADK_PHOTO_TYPE_LAPSE, // TODO
  RKADK_PHOTO_TYPE_BUTT
} RKADK_PHOTO_TYPE_E;
```

【Members】

Member Name	Description
RKADK_PHOTO_TYPE_SINGLE	Single snapshot mode
RKADK_PHOTO_TYPE_MULTIPLE	Multi snapshots mode
RKADK_PHOTO_TYPE_LAPSE	Time-lapse photography (Reserved)

【Related data types and interfaces】

RKADK_TAKE_PHOTO_ATTR_S

4.3.2 RKADK_PHOTO_SINGLE_ATTR_S

【Description】

Define the single photo attribute structure.

【Definition】

```
typedef struct {
  // TODO
  RKADK_S32 s32Time_sec;
} RKADK_PHOTO_SINGLE_ATTR_S;
```

【Members】

Member Name	Description
s32Time_sec	Reserved

【Related data types and interfaces】

RKADK_TAKE_PHOTO_ATTR_S

4.3.3 RKADK_PHOTO_MULTIPLE_ATTR_S

【Description】

Define the multi photos attribute structure.

【Definition】

```
typedef struct {
  /* s32Count is -1 that means continuous photo, larger than 0 that meas photo
   * number */
  RKADK_S32 s32Count;
} RKADK_PHOTO_MULTIPLE_ATTR_S;
```

【Members】

Member Name	Description
s32Count	Number of continuous photo taking, -1 means continuous photo taking until RKADK_PHOTO_DeInit is called

【Related data types and interfaces】

RKADK_TAKE_PHOTO_ATTR_SS

4.3.4 RKADK_PHOTO_THUMB_ATTR_S

【Description】

Define the thumbnail parameter attribute structure.

【Definition】

```
#define RKADK_MPF_LARGE_THUMB_NUM_MAX 2  /* Supports the maximum number of MPF
thumbnails to be generated simultaneously */

typedef struct rkSIZE_S {
  RK_U32 u32Width;              /* Thumbnail width*/
  RK_U32 u32Height;             /* Thumbnail height */
} SIZE_S;

typedef enum {
  RKADK_PHOTO_MPF_SINGLE = 0,    /* Single MPF thumbnail */
  RKADK_PHOTO_MPF_MULTI,         /* Multiple MPF thumbnails */
  RKADK_PHOTO_MPF_BUTT
} RKADK_PHOTO_MPF_MODE_E;

typedef struct {
  RKADK_U8 u8LargeThumbNum;      /* Number of generated MPF thumbnails*/
  SIZE_S astLargeThumbSize[RKADK_MPF_LARGE_THUMB_NUM_MAX];  /* Each thumbnail
resolution */
} RKADK_PHOTO_MPF_CFG_S;

typedef struct {
  RKADK_PHOTO_MPF_MODE_E eMode;  /* MPF thumbnail mode */
  RKADK_PHOTO_MPF_CFG_S sCfg;    /* MPF thumbnail configuration */
} RKADK_PHOTO_MPF_ATTR_S;

typedef struct {
```

```
  RKADK_BOOL bSupportDCF; /* Fixed resolution: 160 * 120 jpg */
  RKADK_PHOTO_MPF_ATTR_S stMPFAttr;
} RKADK_PHOTO_THUMB_ATTR_S;
```

【Members】

Member Name	Description
bSupportDCF	Whether to generate DCF thumbnails (fixed resolution 160*120)
stMPFAttr	MPF thumbnail attribute parameters

【Related data types and interfaces】

RKADK_PHOTO_ATTR_S

4.3.5 RKADK_PHOTO_RECV_DATA_S

【Description】

Define the photo data structure.

【Definition】

```
typedef struct {
  RKADK_U8 *pu8DataBuf;
  RKADK_U32 u32DataLen;
  RKADK_U32 u32CamId;
  bool bStreamEnd;
  void *userdata;
} RKADK_PHOTO_RECV_DATA_S;
```

【Members】

Member Name	Description
pu8DataBuf	Data pointer
u32DataLen	Data length
u32CamId	Camera ID
bStreamEnd	Whether the data stream ends, mainly used for JPEG Slice
userdata	User data pointer

【Related data types and interfaces】

RKADK_PHOTO_DATA_RECV_FN_PTR

4.3.6 RKADK_PHOTO_DATA_RECV_FN_PTR

【Description】

Define the camera data receiving function pointer.

【Definition】

```
typedef void (*RKADK_PHOTO_DATA_RECV_FN_PTR)(RKADK_PHOTO_RECV_DATA_S *pstData);
```

【Members】

Member Name	Description
pstData	Data pointer

【Related data types and interfaces】

RKADK_PHOTO_RECV_DATA_S

RKADK_PHOTO_ATTR_S

4.3.7 RKADK_TAKE_PHOTO_ATTR_S

【Description】

Define the photo attribute structure.

【Definition】

```
typedef struct {
  RKADK_PHOTO_TYPE_E enPhotoType;
  union tagPhotoTypeAttr {
    RKADK_PHOTO_SINGLE_ATTR_S stSingleAttr;
    RKADK_PHOTO_LAPSE_ATTR_S stLapseAttr; // TODO
    RKADK_PHOTO_MULTIPLE_ATTR_S stMultipleAttr;
  } unPhotoTypeAttr;
} RKADK_TAKE_PHOTO_ATTR_S;
```

【Members】

Member Name	Description
enPhotoType	Photo type
stSingleAttr	Single photo parameter attribute
stMultipleAttr	Multi photos parameter attribute
stLapseAttr	Time-lapse photo parameter attribute (Reserved)

【Related data types and interfaces】

RKADK_PHOTO_TYPE_E

RKADK_PHOTO_SINGLE_ATTR_S

RKADK_PHOTO_MULTIPLE_ATTR_S

RKADK_PHOTO_TakePhoto

4.3.8 RKADK_PHOTO_FMT_CHANGE_S

【Description】

Define the format conversion parameter structure, used for FBC0/NV16 format conversion

【Definition】

```
typedef struct {
  RKADK_U32 u32VencChn;
  RKADK_U32 u32VdecChn;

  //format change venc parameter
  RKADK_U32 u32BitRate;   //default 4M
  RKADK_U32 u32Profile;   //default 77
} RKADK_PHOTO_FMT_CHANGE_S;
```

【Members】

Member Name	Description
u32VencChn	VENC channel ID used for format conversion
u32VdecChn	VDEC channel ID used for format conversion
u32BitRate	Encoding bitrate, default 4M
u32Profile	Encoding level, default 77

【Related Data Types and Interfaces】

RKADK_PHOTO_ATTR_S

RKADK_PHOTO_Init

4.3.9 RKADK_PHOTO_ATTR_S

【Description】

Define the photo task attribute structure.

【Definition】

```
typedef struct {
  RKADK_U32 u32CamId;
  RKADK_PHOTO_THUMB_ATTR_S stThumbAttr;
  RKADK_PHOTO_DATA_RECV_FN_PTR pfnPhotoDataProc;
  void *userdata;
  RKADK_POST_ISP_ATTR_S *pstPostIspAttr;
  RKADK_PHOTO_FMT_CHANGE_S stFmtChange;
} RKADK_PHOTO_ATTR_S;
```

【Members】

Member Name	Description
u32CamId	Camera ID
stThumbAttr	Thumbnail parameter attribute
pfnPhotoDataProc	Photo data receiving callback function pointer
userdata	User data pointer
pstPostIspAttr	Post AI ISP attributes
stFmtChange	Format conversion parameters for FBC0/NV16 format conversion

【Related data types and interfaces】

RKADK_PHOTO_THUMB_ATTR_S

RKADK_PHOTO_DATA_RECV_FN_PTR

RKADK_POST_ISP_ATTR_S

RKADK_PHOTO_FMT_CHANGE_S

RKADK_PHOTO_Init

4.3.10 RKADK_JPG_THUMB_TYPE_E

【Description】

Defines the JPG thumbnail type enumeration.

【Definition】

```
typedef enum {
  RKADK_JPG_THUMB_TYPE_DCF,
  RKADK_JPG_THUMB_TYPE_MFP1,
  RKADK_JPG_THUMB_TYPE_MFP2,
  RKADK_JPG_THUMB_TYPE_BUTT
} RKADK_JPG_THUMB_TYPE_E;
```

【Members】

Member Name	Description
RKADK_JPG_THUMB_TYPE_DCF	DCF thumbnail
RKADK_JPG_THUMB_TYPE_MFP1	MPF1 thumbnail
RKADK_JPG_THUMB_TYPE_MFP2	MPF2 thumbnail

【Related data types and interfaces】

RKADK_PHOTO_GetThmInJpg

RKADK_PHOTO_GetThmInJpgEx

4.3.11 RKADK_THUMB_TYPE_E

【Description】

Defines the output thumbnail type enumeration.

【Definition】

```
typedef enum {
  RKADK_THUMB_TYPE_NV12 = 0,
  RKADK_THUMB_TYPE_JPEG,
  RKADK_THUMB_TYPE_RGB565,
  RKADK_THUMB_TYPE_RGBA8888,
  RKADK_THUMB_TYPE_BGRA8888
} RKADK_THUMB_TYPE_E;
```

【Members】

Member Name	Description
RKADK_THUMB_TYPE_NV12	Output thumbnails in NV12 format
RKADK_THUMB_TYPE_JPEG	Output thumbnails in JPG format
RKADK_THUMB_TYPE_RGB565	Output thumbnails in RGB565 format
RKADK_THUMB_TYPE_RGBA8888	Output thumbnails in RGBA9888 format
RKADK_THUMB_TYPE_BGRA8888	Output format thumbnails in BGRA8888

【Related data types and interfaces】

RKADK_THUMB_ATTR_S

RKADK_GetThmInMp4Ex

RKADK_PHOTO_GetThmInJpgEx

4.3.12 RKADK_THUMB_ATTR_S

【Description】

Define the thumbnail attribute structure.

【Definition】

```
typedef struct {
  RKADK_THUMB_TYPE_E enType;
  // 4 alignment
  RKADK_U32 u32Width;
  // 2 alignment
  RKADK_U32 u32Height;
  // 4 alignment
  RKADK_U32 u32VirWidth;
  // 2 alignment
  RKADK_U32 u32VirHeight;
  RKADK_U8 *pu8Buf;
  RKADK_U32 u32BufSize;
} RKADK_THUMB_ATTR_S;
```

【Members】

Member Name	Description
enType	The the desired thumbnail type
u32Width	Enter the desired thumbnail width and output the actual thumbnail width
u32Height	Enter the desired thumbnail height and output the actual thumbnail height
u32VirWidth	Enter the desired virtual width of the thumbnail and output the actual virtual width of the thumbnail
u32VirHeight	Enter the desired virtual height of the thumbnail and output the actual virtual height of the thumbnail
pu8Buf	Thumbnail data pointer
u32BufSize	Thumbnail data length

【Related data types and interfaces】

RKADK_THUMB_TYPE_E

RKADK_GetThmInMp4Ex

RKADK_PHOTO_GetThmInJpgEx

4.3.13 ROTATION_E

【Description】

Defines an enumeration of rotation types.

【Definition】

```
typedef enum rkROTATION_E {
    ROTATION_0        = 0,
    ROTATION_90       = 1,
    ROTATION_180      = 2,
    ROTATION_270      = 3,
    ROTATION_BUTT
} ROTATION_E;
```

【Related data types and interfaces】

RKADK_MEDIA_SetVencRotation

5. Remote Preview

5.1 Overview

It provides callback interfaces for previewing to obtain Video and Audio information, start and stop VENC, start and stop AENC, and register functions for processing audio and video frame data.

5.2 API Reference

5.2.1 RKADK_STREAM_VideoInit

【Description】

Initialize Video module: VI, VENC.

【Syntax】

RKADK_S32 RKADK_STREAM_VideoInit(RKADK_STREAM_VIDEO_ATTR_S *pstVideoAttr, RKADK_MW_PTR *ppHandle);

【Parameters】

Parameter name	Description	Input/output
pstVideoAttr	Video attribute pointer	Input
ppHandle	Created Video task pointer	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_stream.h

Library file: librkadk.so

【Notice】

- Repeated initialization of the Video module is not supported.

【Example】

rkadk_stream_test

【See Also】

RKADK_STREAM_VideoDeInit

5.2.2 RKADK_STREAM_VideoDeInit

【Description】

Deinitialize Video modules: VI, VENC.

【Syntax】

RKADK_S32 RKADK_STREAM_VideoDeInit(RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Video task pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_stream.h

Library file: librkadk.so

【Notice】

- Repeated deinitialization is not supported.

【Example】

rkadk_stream_test

【See Also】

RKADK_STREAM_VideoInit

5.2.3 RKADK_STREAM_VencStart

【Description】

Enable VENC.

【Syntax】

RKADK_S32 RKADK_STREAM_VencStart(RKADK_MW_PTR pHandle, RKADK_S32 s32FrameCnt);

【Parameters】

Parameter name	Description	Input/output
PHandle	Video task pointer	Input
s32FrameCnt	Specify the number of image frames to be received, -1 means infinite reception until VencStop is called	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_stream.h

Library file: librkadk.so

【Notice】

- The RKADK_STREAM_VencStart interface can be called only after the Video module is initialized.
- After calling RKADK_STREAM_VencStart, trigger the VENC data callback function to start receiving data.

【Example】

rkadk_stream_test

【See Also】

RKADK_STREAM_VencStop

5.2.4 RKADK_STREAM_VencStop

【Description】

Disable VENC.

【Syntax】

RKADK_S32 RKADK_STREAM_VencStop(RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Video task pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_stream.h

Library file: librkadk.so

【Notice】

None

【Example】

rkadk_stream_test

【See Also】

RKADK_STREAM_VencStart

5.2.5 RKADK_STREAM_GetVideoInfo

【Description】

Get Video information.

【Syntax】

RKADK_S32 RKADK_STREAM_GetVideoInfo(RKADK_U32 u32CamId, RKADK_VIDEO_INFO_S *pstVideoInfo);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
pstVideoInfo	Video information structure pointer	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_stream.h

Library file: librkadk.so

【Notice】

None

【Example】

rkadk_stream_test

【See Also】

None

5.2.6 RKADK_STREAM_AudioInit

【Description】

Initialize Audio modules: AI, AENC.

【Syntax】

RKADK_S32 RKADK_STREAM_AudioInit(RKADK_STREAM_AUDIO_ATTR_S *pstAudioAttr, RKADK_MW_PTR *ppHandle);

【Parameters】

Parameter name	Description	Input/output
pstAudioAttr	Audio attribute pointer	Input
ppHandle	Created Audio task pointer	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_stream.h

Library file: librkadk.so

【Notice】

- Repeated initialization of the Audio module is not supported.

【Example】

rkadk_stream_test

【See Also】

RKADK_STREAM_AudioDeInit

5.2.7 RKADK_STREAM_AudioDeInit

【Description】

Deinitialize Audio modules: AI, AENC.

【Syntax】

RKADK_S32 RKADK_STREAM_AudioDeInit(RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Audio task pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_stream.h

Library file: librkadk.so

【Notice】

None

【Example】

rkadk_stream_test

【See Also】

RKADK_STREAM_AudioInit

5.2.8 RKADK_STREAM_AencStart

【Description】

Enable AENC.

【Syntax】

RKADK_S32 RKADK_STREAM_AencStart(RKADK_MW_PTR *pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Audio task pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_stream.h

Library file: librkadk.so

【Notice】

- The RKADK_STREAM_AencStart interface can be called only after the Audio module is initialized.
- After calling RKADK_STREAM_AencStart, trigger the AENC data callback function to start receiving data.

【Example】

rkadk_stream_test

【See Also】

RKADK_STREAM_AencStop

5.2.9 RKADK_STREAM_AencStop

【Description】

Disable AENC.

【Syntax】

RKADK_S32 RKADK_STREAM_AencStop(RKADK_MW_PTR *pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Audio task pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_stream.h

Library file: librkadk.so

【Notice】

None

【Example】

rkadk_stream_test

【See Also】

RKADK_STREAM_AencStart

5.2.10 RKADK_STREAM_GetAudioInfo

【Description】

Get Audio information.

【Syntax】

RKADK_S32 RKADK_STREAM_GetAudioInfo(RKADK_MW_PTR *pHandle, RKADK_AUDIO_INFO_S *pstAudioInfo);

【Parameters】

Parameter name	Description	Input/output
PHandle	Audio task pointer	Input
pstAudioInfo	Audio information structure pointer	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_stream.h

Library file: librkadk.so

【Notice】

None

【Example】

rkadk_stream_test

【See Also】

None

5.3 Type of Data

The playback module mainly provides the following data types:

RKADK_CODEC_TYPE_E: Encoding format enumeration

RKADK_VENC_DATA_PROC_FUNC: VENC data callback function pointer

RKADK_VIDEO_STREAM_S: Video data stream structure

RKADK_VENC_DATA_PACK_S: VENC data packet structure

RKADK_VENC_DATA_TYPE_S: VENC packet type

RKADK_VIDEO_INFO_S: Video information structure

RKADK_STREAM_VIDEO_ATTR_S: Video task attribute structure

RKADK_AUDIO_DATA_PROC_FUNC: Audio data callback function pointer

RKADK_AUDIO_STREAM_S: Audio data structure

RKADK_AUDIO_INFO_S: Audio information structure

RKADK_STREAM_AUDIO_ATTR_S: Audio task attribute structure

5.3.1 RKADK_CODEC_TYPE_E

【Description】

Define the encoding format enumeration type. Audio encoding does not support AAC by default.

【Definition】

```
typedef enum {
  //Video
  RKADK_CODEC_TYPE_H264 = 0,
  RKADK_CODEC_TYPE_H265,
  RKADK_CODEC_TYPE_MJPEG,
  RKADK_CODEC_TYPE_JPEG,

  //Audio
  RKADK_CODEC_TYPE_G711A,
  RKADK_CODEC_TYPE_G711U,
  RKADK_CODEC_TYPE_G726,
  RKADK_CODEC_TYPE_MP2,
  RKADK_CODEC_TYPE_MP3,
  RKADK_CODEC_TYPE_ACC,
  RKADK_CODEC_TYPE_PCM,
  RKADK_CODEC_TYPE_BUTT
} RKADK_CODEC_TYPE_E;
```

5.3.2 RKADK_VENC_DATA_PROC_FUNC

【Description】

Define the VENC data callback function pointer.

【Definition】

```
typedef RKADK_S32 (*RKADK_VENC_DATA_PROC_FUNC)(RKADK_VIDEO_STREAM_S
*pVStreamData);
```

【Related data types and interfaces】

RKADK_VIDEO_STREAM_S

5.3.3 RKADK_VIDEO_STREAM_S

【Description】

Define the Video stream structure.

【Definition】

```
typedef struct {
  RKADK_VENC_DATA_PACK_S astPack; /* stream pack attribute */
  RKADK_U32 u32Seq;               /* the list number of stream */
  RKADK_BOOL bEndOfStream;        /* frame end flag */
  RKADK_U32 u32CamId;
} RKADK_VIDEO_STREAM_S;
```

【Members】

Member Name	Description
astPack	Packet structure
u32Seq	Packet sequence ID
bEndOfStream	Reserved
u32CamId	Camera ID

【Related data types and interfaces】

RKADK_VENC_DATA_PACK_S

RKADK_VENC_DATA_PROC_FUNC

5.3.4 RKADK_VENC_DATA_PACK_S

【Description】

Define the VENC packet structure.

【Definition】

```
typedef struct {
  RKADK_U8 *apu8Addr;                /* the virtual address of stream */
  RKADK_U32 au32Len;                 /* the length of stream */
  RKADK_U64 u64PTS;                  /* time stamp */
  RKADK_VENC_DATA_TYPE_S stDataType; /* the type of stream */
} RKADK_VENC_DATA_PACK_S;
```

【Members】

Member Name	Description
apu8Addr	Data pointer
au32Len	Data length
u64PTS	Timestamp
stDataType	Data type

【Related data types and interfaces】

RKADK_VENC_DATA_TYPE_S

RKADK_VIDEO_STREAM_S

5.3.5 RKADK_VENC_DATA_TYPE_S

【Description】

Define VENC packet type.

【Definition】

```
/* the nalu type of H264 */
typedef enum {
  RKADK_H264E_NALU_BSLICE = 0,    /* B SLICE types */
  RKADK_H264E_NALU_PSLICE = 1,    /* P SLICE types */
  RKADK_H264E_NALU_ISLICE = 2,    /* I SLICE types */
  RKADK_H264E_NALU_IDRSLICE = 5, /* IDR SLICE types */
  RKADK_H264E_NALU_SEI = 6,       /* SEI types */
  RKADK_H264E_NALU_SPS = 7,       /* SPS types */
  RKADK_H264E_NALU_PPS = 8,       /* PPS types */
  RKADK_H264E_NALU_BUTT
} RKADK_H264E_NALU_TYPE_E;

/* the nalu type of H265 */
```

```
typedef enum {
  RKADK_H265E_NALU_BSLICE = 0,     /* B SLICE types */
  RKADK_H265E_NALU_PSLICE = 1,     /* P SLICE types */
  RKADK_H265E_NALU_ISLICE = 2,     /* I SLICE types */
  RKADK_H265E_NALU_IDRSLICE = 19, /* IDR SLICE types */
  RKADK_H265E_NALU_VPS = 32,       /* VPS types */
  RKADK_H265E_NALU_SPS = 33,       /* SPS types */
  RKADK_H265E_NALU_PPS = 34,       /* PPS types */
  RKADK_H265E_NALU_SEI = 39,       /* SEI types */
  RKADK_H265E_NALU_BUTT
} RKADK_H265E_NALU_TYPE_E;

typedef struct {
  RKADK_CODEC_TYPE_E enPayloadType;      /* H.264/H.265/JPEG/MJPEG */
  union {
    RKADK_H264E_NALU_TYPE_E enH264EType; /* H264E NALU types */
    RKADK_H265E_NALU_TYPE_E enH265EType; /* H265E NALU types */
    RKADK_JPEGE_PACK_TYPE_E enJPEGEType; /* TODO: JPEGE PACK types*/
  };
} RKADK_VENC_DATA_TYPE_S;
```

【Members】

Member Name	Description
enPayloadType	Encoding type
enH264EType	H264 encoded packet type
enH265EType	H265 encoded packet type
enJPEGEType	Reserved

【Related data types and interfaces】

RKADK_CODEC_TYPE_E

RKADK_VENC_DATA_PACK_S

5.3.6 RKADK_VIDEO_INFO_S

【Description】

Define Video information structure.

【Definition】

```
typedef struct {
  RKADK_CODEC_TYPE_E enCodecType;
  RKADK_U32 u32Width;
  RKADK_U32 u32Height;
  RKADK_U32 u32BitRate;
  RKADK_U32 u32FrameRate;
  RKADK_U32 u32Gop;
} RKADK_VIDEO_INFO_S;
```

【Members】

Member Name	Description
enPayloadType	Encoding type
u32Width	Resolution width
u32Height	Resolution height
u32BitRate	Bit rate
u32FrameRate	Frame rate
u32Gop	I frame interval

【Related data types and interfaces】

RKADK_CODEC_TYPE_E

RKADK_STREAM_GetVideoInfo

5.3.7 RKADK_STREAM_VIDEO_ATTR_S

【Description】

Define the Video task attribute structure.

【Definition】

```
typedef struct {
  RKADK_U32 u32CamId;
  RKADK_VENC_DATA_PROC_FUNC pfnDataCB;
} RKADK_STREAM_VIDEO_ATTR_S;
```

【Members】

Member Name	Description
u32CamId	Camera ID
pfnDataCB	Video data output callback function

【Related data types and interfaces】

RKADK_VENC_DATA_PROC_FUNC

RKADK_STREAM_VideoInit

5.3.8 RKADK_AUDIO_DATA_PROC_FUNC

【Description】

Define the AENC data callback function pointer.

【Definition】

```
typedef RKADK_S32 (*RKADK_AUDIO_DATA_PROC_FUNC)(RKADK_AUDIO_STREAM_S
*pAStreamData);
```

【Related data types and interfaces】

RKADK_AUDIO_STREAM_S

5.3.9 RKADK_AUDIO_STREAM_S

【Description】

Define the Audio data flow structure.

【Definition】

```
typedef struct {
  RKADK_U8 *pStream;      /* the virtual address of stream */
  RKADK_U32 u32Len;       /* stream lenth, by bytes */
  RKADK_U64 u64TimeStamp; /* frame time stamp */
  RKADK_U32 u32Seq;       /* frame seq, if stream is not a valid frame,u32Seq is
0 */
  RKADK_CODEC_TYPE_E enType;
} RKADK_AUDIO_STREAM_S;
```

【Members】

Member Name	Description
pStream	Data pointer
u32Len	Data length
u64TimeStamp	Timestamp
u32Seq	Serial number
enType	Audio data encoding type

【Related data types and interfaces】

RKADK_AUDIO_DATA_PROC_FUNC

5.3.10 RKADK_AUDIO_INFO_S

【Description】

Define the Audio information structure.

【Definition】

```
typedef struct {
  RKADK_CODEC_TYPE_E enCodecType;
  RKADK_U32 u32ChnCnt;
  RKADK_U32 u32SampleRate;
  RKADK_U32 u32AvgBytesPerSec;
  RKADK_U32 u32SamplesPerFrame;
  RKADK_U16 u16SampleBitWidth;
} RKADK_AUDIO_INFO_S;
```

【Members】

Member Name	Description
enPayloadType	Encoding type
u32ChnCntt	Number of channels
u32SampleRate	Sampling rate
u32AvgBytesPerSec	Byte rate
u32SamplesPerFrame	Number of samples per frame
u16SampleBitWidth	Number of bits per sample

【Related data types and interfaces】

RKADK_CODEC_TYPE_E

RKADK_STREAM_GetAudioInfo

5.3.11 RKADK_STREAM_AUDIO_ATTR_S

【Description】

Define the Audio task attribute structure.

【Definition】

```
typedef struct {
  RKADK_U32 u32CamId;
  RKADK_CODEC_TYPE_E enCodecType;
  RKADK_AUDIO_DATA_PROC_FUNC pfnPcmDataCB;
  RKADK_AUDIO_DATA_PROC_FUNC pfnAencDataCB;
} RKADK_STREAM_AUDIO_ATTR_S;
```

【Members】

Member Name	Description
u32CamId	Camera ID
enCodecType	Encoding type
pfnPcmDataCB	PCM data output callback function
pfnAencDataCB	Audio encoded data output callback function

【Related data types and interfaces】

RKADK_AUDIO_DATA_PROC_FUNC

RKADK_STREAM_AudioInit

6. Player

6.1 Overview

It provides local audio and video files, RTSP network stream playback functions, and supports basic playback control operations: play, pause, seek, and screen snapshot.

The Player function is compatible with RV1109/RV1126, RV1103/RV1103, RK3308, and RK3506 platforms. RV1109/RV1126 uses hardware decoding, and other platforms use software decoding.

6.2 API Reference

6.2.1 RKADK_PLAYER_Create

【Description】

Create a player.

【Syntax】

RKADK_S32 RKADK_PLAYER_Create(RKADK_MW_PTR *ppPlayer, RKADK_PLAYER_CFG_S *pstPlayCfg);

【Parameters】

Parameter name	Description	Input/output
ppPlayer	Created player pointer	Output
pstPlayCfg	Player Properties	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

- Repeated creation of the same player is not supported.

【Example】

rkadk_player_test

【See Also】

RKADK_PLAYER_Destroy

6.2.2 RKADK_PLAYER_Destroy

【Description】

Destroy the player.

【Syntax】

RKADK_S32 RKADK_PLAYER_Destroy(RKADK_MW_PTR pPlayer);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

- Repeated destruction of the same player is not supported.

【Example】

rkadk_player_test

【See Also】

RKADK_PLAYER_Create

6.2.3 RKADK_PLAYER_SetDataSource

【Description】

Set the path of the file to be played. When Player enables the third-party demuxer library, RKADK_PLAYER_SetDataParam should be used.

【Syntax】

RKADK_S32 RKADK_PLAYER_SetDataSource(RKADK_MW_PTR pPlayer, const RKADK_CHAR *pszfilePath);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input
pszfilePath	The path of the file to be played	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

- The RKADK_PLAYER_SetDataSource interface can be called only after the player is created.

【Example】

rkadk_player_test

【See Also】

None

6.2.4 RKADK_PLAYER_SetDataParam

【Description】

Set the audio and video parameters of the file to be played. Use this interface when Player enables the third-party demuxer library, otherwise use RKADK_PLAYER_SetDataSource.

【Syntax】

RKADK_S32 RKADK_PLAYER_SetDataParam(RKADK_MW_PTR pPlayer, RKADK_PLAYER_DATA_PARAM_S *pstDataParam);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input
pstDataParam	Audio and video parameters of the file to be played	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

- The RKADK_PLAYER_SetDataParam interface can be called only after the player is created.

【Example】

rkadk_player_test

【See Also】

RKADK_PLAYER_SendAudioPacket

RKADK_PLAYER_SendVideoPacket

6.2.5 RKADK_PLAYER_Prepare

【Description】

Ready to play.

【Syntax】

RKADK_S32 RKADK_PLAYER_Prepare(RKADK_MW_PTR pPlayer);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

- The RKADK_PLAYER_Prepare interface can only be called after creating the player and setting the playback path.

【Example】

rkadk_player_test

【See Also】

None

6.2.6 RKADK_PLAYER_GetCurrentPosition

【Description】

Get the current playback progress.

【Syntax】

RKADK_S64 RKADK_PLAYER_GetCurrentPosition(RKADK_MW_PTR pPlayer);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input

【Return value】

Return value	Description
Playback progress, unit ms	Success
-1	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

None

【Example】

rkadk_player_test

【See Also】

None

6.2.7 RKADK_PLAYER_Play

【Description】

Start playing.

【Syntax】

RKADK_S32 RKADK_PLAYER_Play(RKADK_MW_PTR pPlayer);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

- The RKADK_PLAYER_Play interface can be called only after calling RKADK_PLAYER_Prepare.

【Example】

rkadk_player_test

【See Also】

RKADK_PLAYER_Stop

6.2.8 RKADK_PLAYER_Stop

【Description】

Stop playback and release resources.

【Syntax】

RKADK_S32 RKADK_PLAYER_Stop(RKADK_MW_PTR pPlayer);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

None

【Example】

rkadk_player_test

【See Also】

RKADK_PLAYER_Play

6.2.9 RKADK_PLAYER_Pause

【Description】

Pause playback.

【Syntax】

RKADK_S32 RKADK_PLAYER_Pause(RKADK_MW_PTR pPlayer);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

None

【Example】

rkadk_player_test

【See Also】

RKADK_PLAYER_Play

6.2.10 RKADK_PLAYER_Seek

【Description】

Seek.

【Syntax】

RKADK_S32 RKADK_PLAYER_Seek(RKADK_MW_PTR pPlayer, RKADK_S64 s64TimeInMs);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input
s64TimeInMs	Seek duration	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

- The RKADK_PLAYER_Seek interface can be called only after calling RKADK_PLAYER_Play.

【Example】

rkadk_player_test

【See Also】

RKADK_PLAYER_Play

6.2.11 RKADK_PLAYER_GetPlayStatus

【Description】

Get the current playback status.

【Syntax】

RKADK_S32 RKADK_PLAYER_GetPlayStatus(RKADK_MW_PTR pPlayer, RKADK_PLAYER_STATE_E *penState);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input
penState	Current playback state	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

None

【Example】

rkadk_player_test

【See Also】

None

6.2.12 RKADK_PLAYER_GetDuration

【Description】

Get the duration of current playing file.

【Syntax】

RKADK_S32 RKADK_PLAYER_GetDuration(RKADK_MW_PTR pPlayer, RKADK_U32 *pDuration);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input
pDuration	The duration of current playing file, unit is ms	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

None

【Example】

rkadk_player_test

【See Also】

None

6.2.13 RKADK_PLAYER_Snapshot

【Description】

Player screenshot, calling this interface will encode the currently displayed picture into JPEG data. The application can obtain the generated JPEG data by registering the RKADK_PPLAYER_SNAPSHOT_RECV_FN callback.

【Syntax】

RKADK_S32 RKADK_PLAYER_Snapshot(RKADK_MW_PTR pPlayer);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Example】

rkadk_player_test

【See Also】

None

6.2.14 RKADK_PLAYER_SendAudioPacket

【Description】

When Player enables the third-party demuxer library, this interface is used to send de-capsulated audio data to Player for decoding and playback.

【Syntax】

RKADK_S32 RKADK_PLAYER_SendAudioPacket(RKADK_MW_PTR pPlayer, RKADK_PLAYER_PACKET *pstPacket);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input
pstPacket	sent packet	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

- Before using this interface, you need to call RKADK_PLAYER_SetDataParam to set audio and video related parameters.

【Example】

rkadk_player_test

【See Also】

RKADK_PLAYER_SendVideoPacket

RKADK_PLAYER_SetDataParam

6.2.15 RKADK_PLAYER_SendVideoPacket

【Description】

When Player enables the third-party demuxer library, this interface is used to send de-capsulated video data to Player for decoding and playback.

【Syntax】

RKADK_S32 RKADK_PLAYER_SendVideoPacket(RKADK_MW_PTR pPlayer, RKADK_PLAYER_PACKET *pstPacket);

【Parameters】

Parameter name	Description	Input/output
pPlayer	Player pointer	Input
pstPacket	sent packet	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_player.h

Library file: librkadk.so

【Notice】

- Before using this interface, you need to call RKADK_PLAYER_SetDataParam to set audio and video related parameters.

【Example】

rkadk_player_test

【See Also】

RKADK_PLAYER_SendAudioPacket

RKADK_PLAYER_SetDataParam

6.2.16 RKADK_PLAYER_GetSendFrameNum

【Description】

Retrieve the number of frames that have been successfully decoded.

【Syntax】

RKADK_S32 RKADK_PLAYER_GetSendFrameNum(RKADK_MW_PTR pPlayer);

【Parameters】

Parameter Name	Description	Input/Output
pPlayer	Player pointer	Input

【Return Value】

Return Value	Description
Frame count	Success
-1	Failure

【Requirements】

Header file: rkadk_player.h

Library file: librkadk.so

【Note】

- The count will be reset after Pause/Stop.

【Example】

rkadk_player_test

【Related Topics】

None

6.2.17 RKADK_PLAYER_SetVdecWaterline

【Description】

Set the VDEC waterline.

【Syntax】

RKADK_S32 RKADK_PLAYER_SetVdecWaterline(RKADK_MW_PTR pPlayer, RKADK_U32 u32VdecWaterline);

【Parameters】

Parameter Name	Description	Input/Output
pPlayer	Player pointer	Input
u32VdecWaterline	Waterline value	Input

【Return Value】

Return Value	Description
0	Success
Other value	Failure

【Requirements】

Header file: rkadk_player.h

Library file: librkadk.so

【Note】

- After setting the VDEC waterline, playback will only start when the VDEC buffer reaches the waterline value.

【Example】

rkadk_player_test

【Related Topics】

None

6.2.18 RKADK_PLAYER_SetAoVolume

【Description】

Set playback volume.

【Syntax】

RKADK_S32 RKADK_PLAYER_SetAoVolume(RKADK_MW_PTR pPlayer, RKADK_S32 s32Volume);

【Parameters】

Parameter Name	Description	Input/Output
pPlayer	Player pointer	Output
s32Volume	Volume value	Input

【Return Values】

Return Value	Description
0	Success
Other value	Failure

【Requirements】

Header file: rkadk_player.h

Library file: librkadk.so

【Notes】

None

【Example】

rkadk_player_test

【Related Topics】

None

6.3 Type of Data

The playback module mainly provides the following data types:

RKADK_PLAYER_EVENT_E: Playback event enumeration type

RKADK_PLAYER_EVENT_FN: Playback event callback function pointer

RKADK_PLAYER_CFG_S: Player attribute structure

RKADK_VO_FORMAT_E: Image pixel format enumeration type

RKADK_VO_INTF_TYPE_E: Display interface enumeration type

RKADK_PLAYER_FRAME_INFO_S: Image information structure

RKADK_PLAYER_STATE_E: Playback status enumeration type

RKADK_PLAYER_DATA_PARAM_S: Audio and video parameter structure

RKADK_PLAYER_PACKET: Player data packet structure

RKADK_PLAYER_RTSP_CFG_S: RTSP attribute structure

RKADK_PLAYER_VDEC_CFG_S: VDEC attribute structure

RKADK_PLAYER_SNAPSHOT_CFG_S: Screenshot attribute structure

RKADK_PLAYER_SNAPSHOT_S: Screenshot data structure

RKADK_PPLAYER_SNAPSHOT_RECV_FN: Screenshot data callback function pointer

6.3.1 RKADK_PLAYER_EVENT_E

【Description】

Defines the playback event enumeration type.

【Definition】

```
typedef enum {
  RKADK_PLAYER_EVENT_STATE_CHANGED = 0x0,
  RKADK_PLAYER_EVENT_PREPARED,
  RKADK_PLAYER_EVENT_PLAY,
  RKADK_PLAYER_EVENT_PAUSED,
  RKADK_PLAYER_EVENT_STOPPED,
  RKADK_PLAYER_EVENT_EOF,
  RKADK_PLAYER_EVENT_SOF,
  RKADK_PLAYER_EVENT_SEEK_END,
  RKADK_PLAYER_EVENT_ERROR,
  RKADK_PLAYER_EVENT_BUTT
} RKADK_PLAYER_EVENT_E;
```

【Members】

Member Name	Description
RKADK_PLAYER_EVENT_STATE_CHANGED	Status change (Reserved)
RKADK_PLAYER_EVENT_PREPARED	The `Prepared` is completed
RKADK_PLAYER_EVENT_PLAY	Start playing
RKADK_PLAYER_EVENT_PAUSED	Pause playback (Reserved)
RKADK_PLAYER_EVENT_STOPPED	Stop playing
RKADK_PLAYER_EVENT_EOF	End of playback
RKADK_PLAYER_EVENT_SOF	Reserved
RKADK_PLAYER_EVENT_SEEK_END	The `Seek` is completed (Reserved)

【Related data types and interfaces】

RKADK_PLAYER_EVENT_FN

6.3.2 RKADK_PLAYER_EVENT_FN

【Description】

Define the playback event callback function pointer.

【Definition】

```
typedef RKADK_VOID (*RKADK_PLAYER_EVENT_FN)(RKADK_MW_PTR pPlayer,
RKADK_PLAYER_EVENT_E enEvent, RKADK_VOID *pData);
```

【Members】

Member Name	Description
pPlayer	Player pointer
enEvent	event type
pData	Event related parameters

【Related data types and interfaces】

RKADK_PLAYER_EVENT_E

RKADK_PLAYER_CFG_S

6.3.3 RKADK_PLAYER_CFG_S

【Description】

Define the player attribute structure.

【Definition】

```
typedef struct {
  RKADK_BOOL bEnableVideo;
  RKADK_BOOL bEnableAudio;
  RKADK_BOOL bEnableThirdDemuxer;
  RKADK_PLAYER_FRAME_INFO_S stFrmInfo;
  RKADK_PLAYER_RTSP_CFG_S stRtspCfg;
  RKADK_PLAYER_VDEC_CFG_S stVdecCfg;
  RKADK_PLAYER_SNAPSHOT_CFG_S stSnapshotCfg;
  RKADK_BOOL bEnableBlackBackground;
  RKADK_PLAYER_EVENT_FN pfnPlayerCallback;
} RKADK_PLAYER_CFG_S;
```

【Members】

Member Name	Description
bEnableVideo	Enable video playback
bEnableAudio	Enable audio playback
bEnableThirdDemuxer	Enable third-party demuxer library
pfnPlayerCallback	Player event callback function pointer
stFrmInfo	Define image information
stRtspCfg	RTSP attribute
stVdecCfg	VDEC attribute
stSnapshotCfg	Screenshot attribute
bEnableBlackBackground	Whether the screen goes black after playing

【Related data types and interfaces】

RKADK_PLAYER_EVENT_FN

RKADK_PLAYER_FRAME_INFO_S

RKADK_PLAYER_RTSP_CFG_S

RKADK_PLAYER_VDEC_CFG_S

RKADK_PLAYER_SNAPSHOT_CFG_S

RKADK_PLAYER_Create

6.3.4 RKADK_VO_FORMAT_E

【Description】

Defines the image pixel format enumeration type.

【Definition】

```
typedef enum {
  VO_FORMAT_ARGB8888 = 0,
  VO_FORMAT_ABGR8888,
  VO_FORMAT_RGB888,
  VO_FORMAT_BGR888,
  VO_FORMAT_ARGB1555,
  VO_FORMAT_ABGR1555,
  VO_FORMAT_RGB565,
  VO_FORMAT_RGB444,
  VO_FORMAT_NV12,
  VO_FORMAT_NV21
} RKADK_VO_FORMAT_E;
```

【Related data types and interfaces】

RKADK_PLAYER_FRAME_INFO_S

6.3.5 RKADK_VO_INTF_TYPE_E

【Description】

Defines the display interface enumeration type.

【Definition】

```
typedef enum {
  DISPLAY_TYPE_HDMI = 0,
  DISPLAY_TYPE_EDP,
  DISPLAY_TYPE_VGA,
  DISPLAY_TYPE_DP,
  DISPLAY_TYPE_HDMI_EDP,
  DISPLAY_TYPE_MIPI,
  DISPLAY_TYPE_DEFAULT,
} RKADK_VO_INTF_TYPE_E;
```

【Members】

Member Name	Description
DISPLAY_TYPE_HDMI	The display interface is HDMI
DISPLAY_TYPE_EDP	The display interface is EDP
DISPLAY_TYPE_VGA	The display interface is VGA
DISPLAY_TYPE_MIPI	The display interface is MIPI
DISPLAY_TYPE_DP	The display interface is DP
DISPLAY_TYPE_HDMI_EDP	The display interface is HDMI | EDP
DISPLAY_TYPE_DEFAULT	Internal detection, display interface is the actual connected hardware

【Related data types and interfaces】

RKADK_PLAYER_FRAME_INFO_S

6.3.6 RKADK_VO_SPLICE_MODE_E

【Description】

Defines the layer splicing method enumeration type.

【Definition】

```
typedef enum {
  SPLICE_MODE_RGA = 0,
  SPLICE_MODE_GPU,
  SPLICE_MODE_BYPASS
} RKADK_VO_INTF_TYPE_E;
```

【Members】

Member Name	Description
SPLICE_MODE_RGA	RGA splicing
SPLICE_MODE_GPU	GPU splicing
SPLICE_MODE_BYPASS	BYPASS, no splicing

【Related data types and interfaces】

RKADK_PLAYER_FRAME_INFO_S

6.3.7 RKADK_PLAYER_FRAME_INFO_S

【Description】

Define the image information structure.

【Definition】

```
typedef struct {
  RKADK_U32 u32FrmInfoX;
  RKADK_U32 u32FrmInfoY;
  RKADK_U32 u32DispWidth;
  RKADK_U32 u32DispHeight;
  RKADK_U32 u32ImgWidth;
  RKADK_U32 u32ImgHeight;
  RKADK_U32 u32VoLay;
  RKADK_U32 u32VoDev;
  RKADK_U32 u32VoChn;
  RKADK_U32 u32BorderColor;
  RKADK_U32 u32BorderTopWidth;
  RKADK_U32 u32BorderBottomWidth;
  RKADK_U32 u32BorderLeftWidth;
  RKADK_U32 u32BorderRightWidth;
  RKADK_BOOL bMirror;
  RKADK_BOOL bFlip;
  RKADK_U32 u32Rotation; //0: 0, 1: 90, 2: 180, 3: 270
  RKADK_VO_FORMAT_E u32VoFormat;
  RKADK_VO_INTF_TYPE_E u32EnIntfType;
  RKADK_VO_INTF_SYNC_E enIntfSync;
  RKADK_VO_SYNC_INFO_S stSyncInfo;
  RKADK_VO_SPLICE_MODE_E enVoSpliceMode;
} RKADK_PLAYER_FRAMEINFO_S;
```

【Members】

Member Name	Description
u32FrmInfoX	The x coordinate of the layer display area
u32FrmInfoY	The y coordinate of the layer display area
u32DispWidth	The width of the layer display area
u32DispHeight	The height of the layer display area
u32ImgWidth	Layer image width
u32ImgHeight	Layer image height
u32VoLay	Video output video layer number
u32VoDev	Display output device number
u32VoChn	Video output channel number, value range: [0, VO_MAX_CHN_NUM(128)]
u32BorderColor	Video output channel border attribute: color (Reserved)
u32BorderTopWidth	Video output channel border attribute: top border width (Reserved)
u32BorderBottomWidth	Video output channel border attribute: bottom border width (Reserved)
u32BorderLeftWidth	Video output channel border attribute: left border width (Reserved)
u32BorderRightWidth	Video output channel border attribute: right border width (Reserved)
bMirror	Enable mirror
bFlip	Enable flip
u32Rotation	Rotation, values: [0: 0, 1: 90, 2: 180, 3: 270]
u32VoFormat	Define image pixel format
u32EnIntfType	Display interface type
enIntfSync	Screen interface synchronization mode
stSyncInfo	Screen attribute structure
enVoSpliceMode	Layer splice mode

【Notice】

- For detailed video output related attributes, please refer to the VO chapter of the Rockit document `Rockchip_Developer_Guide_MPI.pdf`.

【Related data types and interfaces】

RKADK_VO_FORMAT_E

RKADK_VO_INTF_TYPE_E

RKADK_VO_SPLICE_MODE_E

RKADK_PLAYER_Create

6.3.8 RKADK_PLAYER_STATE_E

【Description】

Defines the playback status enumeration type.

【Definition】

```
typedef enum {
  RKADK_PLAYER_STATE_IDLE = 0, /* The player state before init */
  RKADK_PLAYER_STATE_INIT,     /* The player is in the initial state. It changes
                                  to the initial state after being SetDataSource
*/
  RKADK_PLAYER_STATE_PREPARED, /* The player is in the prepared state */
  RKADK_PLAYER_STATE_PLAY,     /* The player is in the playing state */
  RKADK_PLAYER_STATE_TPLAY,    /* The player is in the trick playing state,
Reserved */
  RKADK_PLAYER_STATE_PAUSE,    /* The player is in the pause state */
  RKADK_PLAYER_STATE_ERR,      /* The player is in the err state */
  RKADK_PLAYER_STATE_BUTT
} RKADK_PLAYER_STATE_E;
```

【Related data types and interfaces】

RKADK_PLAYER_GetPlayStatus

6.3.9 RKADK_PLAYER_SNAPSHOT_S

【Description】

Define the screenshot data structure.

【Definition】

```
typedef struct {
  RKADK_U32 u32Width;
  RKADK_U32 u32Height;
  RKADK_U32 u32DataLen;
  RKADK_U8 *pu8DataBuf;
} RKADK_PLAYER_SNAPSHOT_S;
```

【Members】

Member Name	Description
u32Width	The width of the screenshot
u32Height	The height of the screenshot
u32DataLen	The data length of the screenshot
pu8DataBuf	The data pointer of the screenshot

【Related data types and interfaces】

RKADK_PPLAYER_SNAPSHOT_RECV_FN

6.3.10 RKADK_PPLAYER_SNAPSHOT_RECV_FN

【Description】

Defines the screenshot data callback function pointer.

【Definition】

```
typedef void (*RKADK_PPLAYER_SNAPSHOT_RECV_FN)(RKADK_PLAYER_SNAPSHOT_S *pstData);
```

【Members】

Member Name	Description
pstData	The data pointer of the screenshot

【Related data types and interfaces】

RKADK_PLAYER_SNAPSHOT_S

RKADK_PLAYER_SNAPSHOT_CFG_S

6.3.11 RKADK_PLAYER_SNAPSHOT_CFG_S

【Description】

Define the screenshot attribute structure.

【Definition】

```
typedef struct {
  RKADK_U32 u32VencChn;
  RKADK_U32 u32MaxWidth;    //Support snapshot max width, default 4096
  RKADK_U32 u32MaxHeight;   //Support snapshot max height, default 4096
  RKADK_PPLAYER_SNAPSHOT_RECV_FN pfnDataCallback;
} RKADK_PLAYER_SNAPSHOT_CFG_S;
```

【Members】

Member Name	Description
u32VencChn	JPEG encoding channel
u32MaxWidth	Maximum width of screenshot, default 4096
u32MaxHeight	Maximum height of screenshot, default 4096
pfnDataCallback	Screenshot data callback pointer

【Related data types and interfaces】

RKADK_PPLAYER_SNAPSHOT_RECV_FN

RKADK_PLAYER_CFG_S

6.3.12 RKADK_PLAYER_VDEC_CFG_S

【Description】

Define the VDEC attribute structure.

【Definition】

```
typedef struct {
  RKADK_U32 u32FrameBufCnt; //frame buffer cnt(output), default: 3
  RKADK_U32 u32StreamBufCnt; //stream buffer cnt(input), default: 3
} RKADK_PLAYER_VDEC_CFG_S;
```

【Members】

Member Name	Description
u32FrameBufCnt	Number of output buffers, it is 3 by default
u32StreamBufCnt	Number of input buffers, it is 3 by default

【Related data types and interfaces】

RKADK_PLAYER_CFG_S

6.3.13 RKADK_PLAYER_RTSP_CFG_S

【Description】

Define the RTSP attribute structure.

【Definition】

```
typedef struct {
  const char *transport; //udp or tcp, default: udp
  RKADK_U32 u32IoTimeout; //timeout (in microseconds) of socket I/O operations
} RKADK_PLAYER_RTSP_CFG_S;
```

【Members】

Member Name	Description
transport	Transport protocol, it is UDP by default
u32ITimeout	Socket I/O operation timeout duration, unit is ms, it is no timeout by default

【Related data types and interfaces】

RKADK_PLAYER_CFG_S

6.3.14 RKADK_PLAYER_PACKET

【Description】

Define the data packet structure and enable the use of the third-party demuxer library.

【Definition】

```
typedef struct {
  bool bEofFlag;
  RKADK_S8 *s8PacketData;
  RKADK_S32 s32PacketSize;
  RKADK_U32 u32Seq;
  RKADK_S64 s64Pts;

  //if bypass, must set pFreeCB;
  bool bBypass;
  RKADK_MPI_MB_FREE_CB pFreeCB;
} RKADK_PLAYER_PACKET;
```

【Members】

Member Name	Description
bEofFlag	Whether it is the last frame of data
s8PacketData	Data pointer
s32PacketSize	Data length
u32Seq	Serial number
s64Pts	Timestamp
bBypass	Data transmission method, true: pass-through, false: secondary copy
pFreeCB	The s8PacketData releases the function pointer, it must be set when bBypass is true

【Related data types and interfaces】

RKADK_PLAYER_SendAudioPacket

RKADK_PLAYER_SendVideoPacket

6.3.15 RKADK_PLAYER_DATA_PARAM_S

【Description】

Define the audio and video parameter structure, it can be used when enabling the third-party demuxer library.

【Definition】

```
typedef struct {
  const RKADK_CHAR *pFilePath;
  RKADK_BOOL bIsRtsp;
  RKADK_BOOL bVideoExist;
  RKADK_BOOL bAudioExist;

  //video param
  RKADK_CODEC_TYPE_E enVCodecType;
  RKADK_U32 u32Width;
  RKADK_U32 u32Height;
  RKADK_FORMAT_E enPixFmt; //output pixel format
  RKADK_U32 u32FrameRate;

  //audio param
  RKADK_CODEC_TYPE_E enACodecType;
  RKADK_S32 u32BitWidth;
  RKADK_S32 u32SampleRate;
  RKADK_S32 u32Channel;
} RKADK_PLAYER_DATA_PARAM_S;
```

【Members】

Member Name	Description
pFilePath	The path of the file to be played
bIsRtsp	Whether it is RTSP network stream
bVideoExist	Whether there is a video stream in the file to be played
bAudioExist	Whether there is an audio stream in the file to be played
enVCodecType	Video stream decoding format
u32Width	Video stream width
u32Height	Video stream high
u32FrameRate	Video stream frame rate
enACodecType	Audio stream decoding format
u32BitWidth	Audio stream bit width
u32SampleRate	Audio stream sample rate
u32Channel	Number of audio stream channels

【Related data types and interfaces】

RKADK_PLAYER_SetDataParam

7. Live Streaming

7.1 Overview

Provides basic services of standard RTSP live streaming; provides RTMP live streaming services. RTSP and RTMP do not support starting at the same time.

7.2 API Reference

7.2.1 RTSP

7.2.1.1 RKADK_RTSP_Init

【Description】

Initialize the RTSP module.

【Syntax】

RKADK_S32 RKADK_RTSP_Init(RKADK_U32 u32CamId, RKADK_U32 port, const char *path, RKADK_MW_PTR *ppHandle);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
port	Port number	Input
path	RTSP address	Input
ppHandle	Created RTSP Handle	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_rtsp.h

Library file: librkadk.so

【Notice】

- Call RKADK_RTSP_Start after RKADK_RTSP_Init to start RTSP live streaming.

【Example】

rkadk_rtsp_test

【See Also】

RKADK_RTSP_DeInit

7.2.1.2 RKADK_RTSP_DeInit

【Description】

Deinitialize the RTSP module.

【Syntax】

RKADK_S32 RKADK_RTSP_DeInit(RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Created RTSP Handle	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_rtsp.h

Library file: librkadk.so

【Example】

rkadk_rtsp_test

【See Also】

RKADK_RTSP_Init

7.2.1.3 RKADK_RTSP_Start

【Description】

Start RTSP live streaming.

【Syntax】

RKADK_S32 RKADK_RTSP_Start(RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Created RTSP Handle	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_rtsp.h

Library file: librkadk.so

【Notice】

- Call this interface after RKADK_RTSP_Init.

【Example】

rkadk_rtsp_test

【See Also】

RKADK_RTSP_Stop

7.2.1.4 RKADK_RTSP_Stop

【Description】

Stop RTSP live streaming.

【Syntax】

RKADK_S32 RKADK_RTSP_Stop(RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Created RTSP Handle	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_rtsp.h

Library file: librkadk.so

【Example】

rkadk_rtsp_test

【See Also】

RKADK_RTSP_Start

7.2.2 RTMP

7.2.2.1 RKADK_RTMP_Init

【Description】

Initialize RTMP module.

【Syntax】

RKADK_S32 RKADK_RTMP_Init(RKADK_U32 u32CamId, const char *path, RKADK_MW_PTR *ppHandle);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
path	RTMP address	Input
ppHandle	Created RTMP Handle	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_rtmp.h

Library file: librkadk.so

【Example】

rkadk_rtmp_test

【See Also】

RKADK_RTMP_DeInit

7.2.2.2 RKADK_RTMP_DeInit

【Description】

Deinitialize the RTMP module.

【Syntax】

RKADK_S32 RKADK_RTMP_DeInit(RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Created RTMP Handle	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_rtmp.h

Library file: librkadk.so

【Example】

rkadk_rtmp_test

【See Also】

RKADK_RTMP_Init

8. Storage

8.1 Overview

Provides basic storage functions, currently including the following functions:

- File detection, storage, acquisition, management
- Device capacity and status query
- Automatically delete files
- Format

8.2 API Reference

8.2.1 RKADK_STORAGE_Init

【Description】

Storage module initialization.

【Syntax】

RKADK_S32 RKADK_STORAGE_Init(RKADK_MW_PTR *ppHandle, RKADK_STR_DEV_ATTR *pstDevAttr);

【Parameters】

Parameter name	Description	Input/output
ppHandle	Created storage module handle	Output
pstDevAttr	Mounted device attribute pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_storage.h

Library file: librkadk.so

【Notice】

- Repeated initialization is not supported.
- If RKADK_STR_DEV_ATTR pass in NULL, default attributes are used. The default attributes are: 2 folders, named video_front and video_back, sorted by file name, automatic deletion threshold 500~1000M, limit the number of files not used, with the ratio of 50%.

【Example】

rkadk_storage_test

【See Also】

RKADK_STORAGE_Deinit

8.2.2 RKADK_STORAGE_Deinit

【Description】

Storage module deinitialization.

【Syntax】

RKADK_S32 RKADK_STORAGE_Deinit(RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Storage module handle pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_storage.h

Library file: librkadk.so

【Notice】

- Repeated deinitialization is not supported.

【Example】

rkadk_storage_test

【See Also】

RKADK_STORAGE_Init

8.2.3 RKADK_STORAGE_GetDevAttr

【Description】

Get the mounted device attribute.

【Syntax】

RKADK_STR_DEV_ATTR RKADK_STORAGE_GetDevAttr(RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Storage module handle pointer	Input

【Return value】

Return value	Description
RKADK_STR_DEV_ATTR	Mounted device attribute structure

【Requirement】

Header file: rkadk_storage.h

Library file: librkadk.so

【Notice】

- The RKADK_STORAGE_GetDevAttr interface can only be used after the storage module is initialized.

【Example】

None

【See Also】

None

8.2.4 RKADK_STORAGE_GetMountStatus

【Description】

Get device mounting status.

【Syntax】

RKADK_MOUNT_STATUS RKADK_STORAGE_GetMountStatus(RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Storage module handle pointer	Input

【Return value】

Return value	Description
RKADK_MOUNT_STATUS	Mount status enumeration type

【Requirement】

Header file: rkadk_storage.h

Library file: librkadk.so

【Example】

rkadk_storage_test

【See Also】

None

8.2.5 RKADK_STORAGE_GetCapacity

【Description】

Get the device capacity.

【Syntax】

RKADK_S32 RKADK_STORAGE_GetCapacity(RKADK_MW_PTR *ppHandle, RKADK_S32 *totalSize, RKADK_S32 *freeSize);

【Parameters】

Parameter name	Description	Input/output
ppHandle	Created storage module handle	Input/Output
totalSize	Total device capacity pointer	Output
freeSize	Device remaining capacity pointer	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_storage.h

Library file: librkadk.so

【Example】

rkadk_storage_test

【See Also】

None

8.2.6 RKADK_STORAGE_GetFileList

【Description】

Get a list of files.

【Syntax】

RKADK_S32 RKADK_STORAGE_GetFileList(RKADK_FILE_LIST *list, RKADK_MW_PTR pHandle, RKADK_SORT_TYPE sort);

【Parameters】

Parameter name	Description	Input/output
list	File list structure pointer	Input/output
PHandle	Storage module handle pointer	Input
sort	Sort type	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_storage.h

Library file: librkadk.so

【Notice】

- Must be used together with RKADK_STORAGE_FreeFileList.

【Example】

rkadk_storage_test

【See Also】

RKADK_STORAGE_FreeFileList

8.2.7 RKADK_STORAGE_FreeFileList

【Description】

Release file list.

【Syntax】

RKADK_S32 RKADK_STORAGE_FreeFileList(RKADK_FILE_LIST *list);

【Parameters】

Parameter name	Description	Input/output
list	file list structure pointer	Input/output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_storage.h

Library file: librkadk.so

【Notice】

- Must be used together with RKADK_STORAGE_GetFileList.

【Example】

rkadk_storage_test

【See Also】

RKADK_STORAGE_GetFileList

8.2.8 RKADK_STORAGE_GetFileNum

【Description】

Get the number of files.

【Syntax】

RKADK_S32 RKADK_STORAGE_GetFileNum(RKADK_CHAR *fileListPath, RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
fileListPath	File list path pointer	Input
PHandle	Storage module handle pointer	Input

【Return value】

Return value	Description
non-negative	Number of files
-1	Failure

【Requirement】

Header file: rkadk_storage.h

Library file: librkadk.so

【Example】

None

【See Also】

None

8.2.9 RKADK_STORAGE_GetDevPath

【Description】

Get the path of the mounted device.

【Syntax】

RKADK_CHAR *RKADK_STORAGE_GetDevPath(RKADK_MW_PTR pHandle);

【Parameters】

Parameter name	Description	Input/output
PHandle	Storage module handle pointer	Input

【Return value】

Return value	Description
RKADK_CHAR *	Mount device path pointer

【Requirement】

Header file: rkadk_storage.h

Library file: librkadk.so

【Notice】

- The RKADK_STORAGE_GetDevPath interface can only be used after the storage module is initialized.

【Example】

rkadk_storage_test

【See Also】

None

8.2.10 RKADK_STORAGE_Format

【Description】

Device formatting.

【Syntax】

RKADK_S32 RKADK_STORAGE_Format(RKADK_MW_PTR pHandle, RKADK_CHAR *cFormat);

【Parameters】

Parameter name	Description	Input/output
PHandle	Storage module handle pointer	Input
cFormat	File system type	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_storage.h

Library file: librkadk.so

【Example】

None

【See Also】

None

8.3 Type of Data

The storage module mainly provides the following data types:

RKADK_MOUNT_STATUS: Mount status enumeration type

RKADK_SORT_TYPE: Sort type enumeration

RKADK_SORT_CONDITION: Sort condition enumeration type

RKADK_STR_FOLDER_ATTR: Folder attribute structure

RKADK_STR_DEV_ATTR: Device attribute structure

RKADK_FILE_INFO: File information structure

RKADK_FILE_LIST: File list structure

RKADK_FILE_LIST_ARRAY: File list group structure

8.3.1 RKADK_MOUNT_STATUS

【Description】

Defines the mount status enumeration type.

【Definition】

```
typedef enum {
  DISK_UNMOUNTED = 0,
  DISK_NOT_FORMATTED,
  DISK_FORMAT_ERR,
  DISK_SCANNING,
  DISK_MOUNTED,
  DISK_MOUNT_BUTT,
} RKADK_MOUNT_STATUS;
```

【Members】

Member Name	Description
DISK_UNMOUNTED	The disk is not mounted
DISK_NOT_FORMATTED	The disk is not formatted
DISK_FORMAT_ERR	Disk format error
DISK_SCANNING	The disk is being scanned
DISK_MOUNTED	The disk is mounted

【Related data types and interfaces】

RKADK_STORAGE_GetMountStatus

8.3.2 RKADK_SORT_TYPE

【Description】

Defines the sorting type enum.

【Definition】

```
typedef enum {
  LIST_ASCENDING = 0,
  LIST_DESCENDING,
  LIST_BUTT,
} RKADK_SORT_TYPE;
```

【Members】

Member Name	Description
LIST_ASCENDING	Sort the list in ascending order
LIST_DESCENDING	Sort the list in descending order

【Related data types and interfaces】

RKADK_STORAGE_GetFileList

8.3.3 RKADK_SORT_CONDITION

【Description】

Defines the sorting condition enumeration type.

【Definition】

```
typedef enum {
  SORT_MODIFY_TIME = 0,
  SORT_FILE_NAME,
  SORT_BUTT,
} RKADK_SORT_CONDITION;
```

【Members】

Member Name	Description
SORT_MODIFY_TIME	The list is sorted by file modification time
SORT_FILE_NAME	List sorted by file name

【Related data types and interfaces】

RKADK_STR_FOLDER_ATTR

8.3.4 RKADK_STR_FOLDER_ATTR

【Description】

Define the folder attribute structure.

【Definition】

```
typedef struct {
  RKADK_CHAR cFolderPath[RKADK_MAX_FILE_PATH_LEN];
  RKADK_SORT_CONDITION s32SortCond;
  RKADK_BOOL bNumLimit;
  RKADK_S32 s32Limit;
} RKADK_STR_FOLDER_ATTR;
```

【Members】

Member Name	Description
cFolderPath	Folder path
s32SortCond	Sorting conditions
bNumLimit	Option: Whether to set the upper limit based on the number of files
s32Limit	Maximum folder capacity (ratio/number)

【Related data types and interfaces】

RKADK_SORT_CONDITION

RKADK_STR_DEV_ATTR

8.3.5 RKADK_STR_DEV_ATTR

【Description】

Define the device attribute structure.

【Definition】

```
typedef struct {
  RKADK_CHAR cDevPath[RKADK_MAX_FILE_PATH_LEN];
  RKADK_CHAR cMountPath[RKADK_MAX_FILE_PATH_LEN];
  RKADK_S32 s32FreeSizeDelMin;
  RKADK_S32 s32FreeSizeDelMax;
  RKADK_S32 s32AutoDel;
  RKADK_S32 s32FolderNum;
  RKADK_CHAR cFormatId[RKADK_MAX_FORMAT_ID_LEN];
  RKADK_CHAR cVolume[RKADK_MAX_VOLUME_LEN];
  RKADK_S32 s32CheckFormatId;
  RKADK_STR_FOLDER_ATTR *pstFolderAttr;
} RKADK_STR_DEV_ATTR;
```

【Members】

Member Name	Description
cDevPath	Device name (device path)
cMountPath	Device mount path
s32FreeSizeDelMin	Automatically delete the lower limit of the threshold
s32FreeSizeDelMax	Automatically delete the upper limit of the threshold
s32AutoDel	Automatic deletion option
s32FolderNum	Number of folders
cFormatId	Format ID
cVolume	Volume label
s32CheckFormatId	Check whether the format ID matches
pstFolderAttr	Folder attribute structure pointer

【Related data types and interfaces】

RKADK_STR_FOLDER_ATTR

RKADK_STORAGE_Init

RKADK_STORAGE_GetDevAttr

8.3.6 RKADK_FILE_INFO

【Description】

Define the file information structure.

【Definition】

```
typedef struct {
  RKADK_CHAR filename[RKADK_MAX_FILE_PATH_LEN];
  off_t stSize;
  time_t stTime;
  void *thumb;
} RKADK_FILE_INFO;
```

【Members】

Member Name	Description
filename	File name
stSize	File size
stTime	File modification time
thumb	Thumbnail pointer

【Related data types and interfaces】

RKADK_FILE_LIST

8.3.7 RKADK_FILE_LIST

【Description】

Define the file list structure.

【Definition】

```
typedef struct {
  RKADK_CHAR path[RKADK_MAX_FILE_PATH_LEN];
  RKADK_S32 s32FileNum;
  RKADK_FILE_INFO *file;
} RKADK_FILE_LIST;
```

【Members】

Member Name	Description
path	File list (folder) path
s32FileNum	Number of files
file	File information structure pointer

【Related data types and interfaces】

RKADK_FILE_INFO

RKADK_FILE_LIST_ARRAY

8.3.8 RKADK_FILE_LIST_ARRAY

【Description】

Define the file list group structure.

【Definition】

```
typedef struct {
  RKADK_S32 s32ListNum;
  RKADK_FILE_LIST *list;
} RKADK_FILE_LIST_ARRAY;
```

【Members】

Member Name	Description
s32ListNum	Number of file lists (folders)
list	File list structure pointer

【Related data types and interfaces】

RKADK_FILE_LIST

9. Local preview

9.1 Overview

Provides local preview function.

9.2 API Reference

9.2.1 RKADK_DISP_Init

【Description】

Initialize the local preview module.

【Syntax】

RKADK_S32 RKADK_DISP_Init(RKADK_U32 u32CamId);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_disp.h

Library file: librkadk.so

【Example】

rkadk_disp_test

【See Also】

RKADK_DISP_DeInit

9.2.2 RKADK_DISP_DeInit

【Description】

Deinitialize the local preview module.

【Syntax】

RKADK_S32 RKADK_DISP_DeInit(RKADK_U32 u32CamId);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_disp.h

Library file: librkadk.so

【Example】

rkadk_disp_test

【See Also】

RKADK_DISP_Init

9.2.3 RKADK_DISP_SetAttr

【Description】

Set preview properties.

【Syntax】

RKADK_S32 RKADK_DISP_SetAttr(RKADK_U32 u32CamId, RKADK_DISP_ATTR_S *pstAttr);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
pstAttr	preview attributes	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_disp.h

Library file: librkadk.so

9.3 Type of Data

9.3.1 RKADK_DISP_ATTR_S

【Description】

Define the preview attribute structure.

【Definition】

```
typedef struct {
  RKADK_RECT_S stVpssCropRect;
  RKADK_RECT_S stVoRect;
} RKADK_DISP_ATTR_S;
```

【Members】

Member Name	Description
stVpssCropRect	Input display area
stVoRect	Output display area

10. Watermark

10.1 Overview

The Watermark module provides basic watermark functionality

10.2 API Reference

10.2.1 RKADK_OSD_Init

【Description】

Initialize the watermark task.

【Syntax】

RKADK_S32 RKADK_OSD_Init(RKADK_U32 u32OsdId, RKADK_OSD_ATTR_S *pstOsdAttr);

【Parameters】

Parameter name	Description	Input/output
u32OsdId	Watermark ID	Input
pstOsdAttr	Watermark task attribute pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_osd.h

Library file: librkadk.so

【Notice】

- Repeated initialization is not supported.

10.2.2 RKADK_OSD_Deinit

【Description】

Deinitialize the watermark task.

【Syntax】

RKADK_S32 RKADK_OSD_Deinit(RKADK_U32 u32OsdId);

【Parameters】

Parameter name	Description	Input/output
u32OsdId	Watermark ID	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_osd.h

Library file: librkadk.so

【Notice】

- Repeated deinitialization is not supported.

10.2.3 RKADK_OSD_UpdateBitMap

【Description】

Watermark content updated.

【Syntax】

RKADK_S32 RKADK_OSD_UpdateBitMap(RKADK_U32 u32OsdId, RKADK_OSD_ATTR_S *pstOsdAttr);

【Parameters】

Parameter name	Description	Input/output
u32OsdId	Watermark ID	Input
pstOsdAttr	Watermark task attribute pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_osd.h

Library file: librkadk.so

【Notice】

- Must be used after the de-initialization task is completed.

10.2.4 RKADK_OSD_AttachToStream

【Description】

Attach watermark to the target stream.

【Syntax】

RKADK_S32 RKADK_OSD_AttachToStream(RKADK_U32 u32OsdId, RKADK_U32 u32CamId, RKADK_STREAM_TYPE_E enStrmType, RKADK_OSD_STREAM_ATTR_S *pstOsdStreamAttr);

【Parameters】

Parameter name	Description	Input/output
u32OsdId	Watermark ID	Input
u32CamId	Camera ID	Input
enStrmType	Target stream type	Input
pstOsdStreamAttr	Watermark position information pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_osd.h

Library file: librkadk.so

【Notice】

- The same watermark cannot be attached repeatedly on the same stream type.

10.2.5 RKADK_OSD_DettachFromStream

【Description】

Remove watermark from the target stream.

【Syntax】

RKADK_S32 RKADK_OSD_DettachFromStream(RKADK_U32 u32OsdId, RKADK_U32 u32CamId, RKADK_STREAM_TYPE_E enStrmType);

【Parameters】

Parameter name	Description	Input/output
u32OsdId	Watermark ID	Input
u32CamId	Camera ID	Input
enStrmType	Target stream type	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_osd.h

Library file: librkadk.so

10.2.6 RKADK_OSD_UpdateOsdSize

【Description】

Update watermark size.

【Syntax】

RKADK_S32 RKADK_OSD_UpdateOsdSize(RKADK_U32 u32OsdId, RKADK_OSD_ATTR_S *pstOsdAttr);

【Parameters】

Parameter name	Description	Input/output
u32OsdId	Watermark ID	Input
pstOsdAttr	Watermark task attribute pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_osd.h

Library file: librkadk.so

10.2.7 RKADK_OSD_UpdateDisplayAttr

【Description】

Update the watermark display area.

【Syntax】

RKADK_S32 RKADK_OSD_UpdateDisplayAttr(RKADK_U32 u32OsdId, RKADK_U32 u32CamId, RKADK_STREAM_TYPE_E enStrmType, RKADK_OSD_STREAM_ATTR_S *pstOsdStreamAttr);

【Parameters】

Parameter name	Description	Input/output
u32OsdId	Watermark ID	Input
u32CamId	Camera ID	Input
enStrmType	Stream type	Input
pstOsdStreamAttr	Watermark location information	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_osd.h

Library file: librkadk.so

10.3 Type of Data

The watermark module mainly provides the following data types:

RKADK_OSD_ATTR_S: Watermark attribute structure
RKADK_OSD_STREAM_ATTR_S: Watermark location information structure

RKADK_OSD_TYPE_E: Watermark overlay type

10.3.1 RKADK_OSD_ATTR_S

【Description】

Define watermark attributes.

【Definition】

```
typedef struct {
  RKADK_U32 Width;
  RKADK_U32 Height;
  RKADK_VOID *pData;
  RKADK_FORMAT_E Format;
  RKADK_OSD_TYPE_E enOsdType;
} RKADK_OSD_ATTR_S;
```

【Members】

Member Name	Description
Width	Watermark width
Height	Watermark height
pData	Watermark content
Format	Watermark format
enOsdType	Watermark overlay type

【Related data types and interfaces】

RKADK_OSD_ATTR_S

RKADK_OSD_TYPE_E

10.3.2 RKADK_OSD_STREAM_ATTR_S

【Description】

Define watermark location information.

【Definition】

```
typedef struct {
  RKADK_BOOL bEnableShow;
  RKADK_U32 Origin_X;
  RKADK_U32 Origin_Y;
} RKADK_OSD_STREAM_ATTR_S;
```

【Members】

Member Name	Description
Origin_X	Watermark starting position X offset
Origin_Y	Watermark starting position Y offset
bEnableShow	Whether to display watermark

【Related data types and interfaces】

RKADK_OSD_STREAM_ATTR_S

10.4 RKADK_OSD_TYPE_E

【Description】

Define the watermark overlay type.

【Definition】

```
typedef enum {
  RKADK_OSD_TYPE_NORMAL = 0, //use encoder do osd
  RKADK_OSD_TYPE_EXTRA,      //use rga do osd
  RKADK_OSD_TYPE_BUTT
} RKADK_OSD_TYPE_E;
```

【Members】

Member Name	Description
RKADK_OSD_TYPE_NORMAL	Encoder OSD overlay
RKADK_OSD_TYPE_EXTRA	RGA overlay

【Notice】

- RV1109/RV1126 JPEG encoding does not support the encoder OSD overlay watermark, and needs to be configured to RKADK_OSD_TYPE_EXTRA.

11. UI Overlay

11.1 Overview

Provides UI overlay function for overlaying UI and video images on single-layer VOP platforms such as RV1103/RV1106.

11.2 API Reference

11.2.1 RKADK_UI_Create

【Description】

Initialize the UI overlay module

【Syntax】

RKADK_S32 RKADK_UI_Create(RKADK_UI_ATTR_S *pstUiAttr, RKADK_MW_PTR *ppUi);

【Parameters】

Parameter name	Description	Input/output
pstUiAttr	UI attributes	Input
ppUi	Created UI task pointer	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_ui.h

Library file: librkadk.so

【Example】

rkadk_ui_test

【See Also】

RKADK_UI_Destroy

11.2.2 RKADK_UI_Destroy

【Description】

Deinitialize UI overlay module

【Syntax】

RKADK_S32 RKADK_UI_Destroy(RKADK_MW_PTR pUi);

【Parameters】

Parameter name	Description	Input/output
pUi	UI task pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_ui.h

Library file: librkadk.so

【Example】

rkadk_ui_test

【See Also】

RKADK_UI_Create

11.2.3 RKADK_UI_Update

【Description】

Refresh UI data.

【Syntax】

RKADK_S32 RKADK_UI_Update(RKADK_MW_PTR pUi, RKADK_UI_FRAME_INFO *pstUiFrameInfo);

【Parameters】

Parameter name	Description	Input/output
pUi	UI task pointer	Input
pstUiFrameInfo	UI data pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_ui.h

Library file: librkadk.so

【Example】

rkadk_ui_test

11.3 Type of Data

RKADK_UI_ATTR_S: UI attribute structure

RKADK_UI_FRAME_INFO: UI data structure

11.3.1 RKADK_UI_ATTR_S

【Description】

Define the UI attribute structure.

【Definition】

```
typedef struct {
  RKADK_U32 u32DispX;
  RKADK_U32 u32DispY;
  RKADK_U32 u32DispWidth;
  RKADK_U32 u32DispHeight;
  RKADK_U32 u32DispFrmRt;
  RKADK_U32 u32ImgWidth;
  RKADK_U32 u32ImgHeight;
  RKADK_U32 u32VoLay;
  RKADK_U32 u32VoDev;
  RKADK_U32 u32VoChn;
  RKADK_U32 u32Rotation; //0: 0, 1: 90, 2: 180, 3: 270
  RKADK_BOOL bMirror;
  RKADK_BOOL bFlip;
  RKADK_VO_FORMAT_E enUiVoFormat;
  RKADK_VO_INTF_TYPE_E enUiVoIntfTye;
  RKADK_VO_SPLICE_MODE_E enVoSpliceMode;
} RKADK_UI_ATTR_S;
```

【Members】

Member Name	Description
u32DispX	Layer display area x coordinate
u32DispY	Layer display area y coordinate
u32DispWidth	Layer display area width
u32DispHeight	Layer display area height
u32ImgWidth	Layer image width
u32ImgHeight	Layer image height
u32DispFrmRt	Refresh frame rate
u32VoLay	Video output video layer number
u32VoDev	Display output device number
u32VoChn	Video output channel number, value range: [0, VO_MAX_CHN_NUM(128)]
u32Rotation	Rotation, values: [0: 0, 1: 90, 2: 180, 3: 270]
bMirror	Enable mirror
bFlip	Enable flip
enUiVoFormat	Image pixel format
enUiVoIntfTye	Display interface type
enVoSpliceMode	Layer composition mode

【Related data types and interfaces】

RKADK_VO_FORMAT_E

RKADK_VO_INTF_TYPE_E

RKADK_VO_SPLICE_MODE_E

RKADK_UI_Create

11.3.2 RKADK_UI_FRAME_INFO

【Description】

Define UI data information.

【Definition】

```
typedef struct {
  RKADK_U32 u32Width;
  RKADK_U32 u32Height;
  RKADK_FORMAT_E Format;
  RKADK_VOID *pMblk;
} RKADK_UI_FRAME_INFO;
```

【Members】

Member Name	Description
u32Width	UI data width
u32Height	UI data height
Format	UI data format
pMblk	UI data pointer

【Related data types and interfaces】

RKADK_FORMAT_E

RKADK_UI_Update

11.3.3 RKADK_FORMAT_E

【Description】

Define the pixel format.

【Definition】

```
typedef enum {
  RKADK_FMT_ARGB1555,                          /* 16-bit RGB
  */
  RKADK_FMT_ABGR1555,                          /* 16-bit RGB
  */
  RKADK_FMT_RGBA5551,                          /* 16-bit RGB
  */
  RKADK_FMT_BGRA5551,                          /* 16-bit RGB
  */
  RKADK_FMT_ARGB4444,                          /* 16-bit RGB
  */
  RKADK_FMT_ABGR4444,                          /* 16-bit RGB
  */
  RKADK_FMT_RGBA4444,                          /* 16-bit RGB
  */
  RKADK_FMT_BGRA4444,                          /* 16-bit RGB
  */
  RKADK_FMT_ARGB8888,                          /* 32-bit RGB
  */
  RKADK_FMT_ABGR8888,                          /* 32-bit RGB
  */
```

```
  RKADK_FMT_RGBA8888,                                 /* 32-bit RGB
  */
  RKADK_FMT_BGRA8888,                                 /* 32-bit RGB
  */
  RKADK_FMT_2BPP,
  RKADK_FMT_YUV420SP,
  RKADK_FMT_YUV420SP_10BIT,
  RKADK_FMT_YUV422SP,
  RKADK_FMT_BUTT,
} RKADK_FORMAT_E;
```

【Related data types and interfaces】

RKADK_UI_FRAME_INFO

12. Parameter Settings

12.1 Overview

The parameter setting module is strongly related to the product form. By combining and using common component data structures, a data structure suitable for the product form is defined.

This module supports obtaining specified parameters, saving specified parameters, and restoring parameters to default.

To facilitate editing, parameters are stored in the form of ini files.

You can specify the default ini path by setting the environment variable rkadk_default_ini_path. The default path is /oem/usr/etc.

```
export rkadk_default_ini_path=/oem/usr/etc
```

12.2 API Reference

12.2.1 RKADK_PARAM_Init

【Description】

Initialization parameter module

【Syntax】

RKADK_S32 RKADK_PARAM_Init(char *globalSetting, char **sesnorSettingArrary);

【Parameters】

Parameter name	Description	Input/output
globalSetting	Global ini configuration file path	Input
sesnorSettingArrary	Sensor ini configuration file path	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_param.h

Library file: librkadk.so

【Notice】

- Before starting any module, RKADK_PARAM_Init should be called firstly to initialize the parameter module.
- If globalSetting is not set, the default path RKADK_PARAM_PATH is used.
- If sesnorSettingArrary is not set, the default path RKADK_PARAM_PATH_SENSOR_PREFIX is used. The sensor configuration file prefix defaults to rkadk_setting_sensor_n.ini, _n is the sensor camera ID, and the serial number starts from 0.

【Example】

rkadk_record_test

【See Also】

None

12.2.2 RKADK_PARAM_GetCamParam

【Description】

Get Camera related parameters.

【Syntax】

RKADK_S32 RKADK_PARAM_GetCamParam(RKADK_S32 s32CamID, RKADK_PARAM_TYPE_E enParamType, RKADK_VOID *pvParam);

【Parameters】

Parameter name	Description	Input/output
s32CamID	Camera ID	Input
enParamType	Parameter type	Input
pvParam	Obtained parameter pointer	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_param.h

Library file: librkadk.so

【Notice】

- This interface can be called only after calling RKADK_PARAM_Init to initialize the parameter module.

【Example】

rkadk_record_test

【See Also】

RKADK_PARAM_SetCamParam

12.2.3 RKADK_PARAM_SetCamParam

【Description】

Set Camera related parameters.

【Syntax】

RKADK_S32 RKADK_PARAM_SetCamParam(RKADK_S32 s32CamID, RKADK_PARAM_TYPE_E enParamType, const RKADK_VOID *pvParam);

【Parameters】

Parameter name	Description	Input/output
s32CamID	Camera ID	Input
enParamType	Parameter type	Input
pvParam	Set parameter pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_param.h

Library file: librkadk.so

【Notice】

- This interface can be called only after calling RKADK_PARAM_Init to initialize the parameter module.

【Example】

rkadk_record_test

【See Also】

RKADK_PARAM_GetCamParam

12.2.4 RKADK_PARAM_GetCommParam

【Description】

Get common parameters.

【Syntax】

RKADK_S32 RKADK_PARAM_GetCommParam(RKADK_PARAM_TYPE_E enParamType, RKADK_VOID *pvParam);

【Parameters】

Parameter name	Description	Input/output
enParamType	Parameter type	Input
pvParam	Obtained parameter pointer	Output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_param.h

Library file: librkadk.so

【Notice】

- This interface can be called only after calling RKADK_PARAM_Init to initialize the parameter module.

【Example】

rkadk_record_test

【See Also】

RKADK_PARAM_SetCommParam

12.2.5 RKADK_PARAM_SetCommParam

【Description】

Set common parameters.

【Syntax】

RKADK_S32 RKADK_PARAM_SetCommParam(RKADK_PARAM_TYPE_E enParamType, const RKADK_VOID *pvParam);

【Parameters】

Parameter name	Description	Input/output
enParamType	Parameter type	Input
pvParam	Set parameter pointer	Input

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_param.h

Library file: librkadk.so

【Notice】

- This interface can be called only after calling RKADK_PARAM_Init to initialize the parameter module.

【Example】

rkadk_record_test

【See Also】

RKADK_PARAM_GetCommParam

12.2.6 RKADK_PARAM_SetDefault

【Description】

Restore default configuration.

【Syntax】

RKADK_S32 RKADK_PARAM_SetDefault(RKADK_VOID);

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_param.h

Library file: librkadk.so

12.2.7 RKADK_PARAM_GetResolution

【Description】

RKADK_PARAM_RES_E is used to convert to specific resolution.

【Syntax】

RKADK_S32 RKADK_PARAM_GetResolution(RKADK_PARAM_RES_E type, RKADK_U32 *width, RKADK_U32 *height);

【Parameters】

Parameter name	Description	Input/output
type	Resolution type	Input
width	Converted resolution width	output
height	Converted resolution high	output

【Return value】

Return value	Description
0	Success
Other value	Failure

【Requirement】

Header file: rkadk_param.h

Library file: librkadk.so

【Notice】

- This interface can be called only after calling RKADK_PARAM_Init to initialize the parameter module.

【Example】

None

【See Also】

RKADK_PARAM_GetResType

12.2.8 RKADK_PARAM_GetResType

【Description】

Convert resolution to RKADK_PARAM_RES_E.

【Syntax】

RKADK_PARAM_RES_E RKADK_PARAM_GetResType(RKADK_U32 width, RKADK_U32 height);

【Parameters】

Parameter name	Description	Input/output
width	Resolution width	Input
height	Resolution height	Input

【Return value】

Return value	Description
Corresponding RKADK_PARAM_RES_E	Success
RKADK_RES_BUTT	Failure

【Requirement】

Header file: rkadk_param.h

Library file: librkadk.so

【Notice】

- This interface can be called only after calling RKADK_PARAM_Init to initialize the parameter module.

【Example】

None

【See Also】

RKADK_PARAM_GetResolution

12.2.9 RKADK_PARAM_GetVencChnId

【Description】

Get the VENC channel number corresponding to Record, Photo, and Stream.

【Syntax】

RKADK_S32 RKADK_PARAM_GetVencChnId(RKADK_U32 u32CamId, RKADK_STREAM_TYPE_E enStrmType);

【Parameters】

Parameter name	Description	Input/output
u32CamId	Camera ID	Input
enStrmType	Stream type	Input

【Return value】

Return value	Description
Corresponding VENC channel number	Success
-1	Failure

【Requirement】

Header file: rkadk_param.h

Library file: librkadk.so

【Notice】

- This interface can be called only after calling RKADK_PARAM_Init to initialize the parameter module.

【Example】

None

【See Also】

None

12.3 Type of Data

The parameter module mainly provides the following data types:

RKADK_DEFPARAM_PATH: Default global ini configuration file path

RKADK_DEFPARAM_PATH_SENSOR_PREFIX: Default Sensor ini configuration file path

RKADK_PARAM_PATH: Global ini configuration file path

RKADK_PARAM_PATH_SENSOR_PREFIX: Sensor ini configuration file path

RKADK_PARAM_TYPE_E: Parameter type enumeration

RKADK_PARAM_RES_E: Resolution type enumeration

RKADK_STREAM_TYPE_E: Data stream type enumeration

RKADK_PARAM_CODEC_CFG_S: Coding type configuration structure

RKADK_PARAM_BITRATE_S: Bit rate configuration structure

RKADK_PARAM_REC_TIME_S: Recording duration configuration structure

RKADK_PARAM_GOP_S: VENC GOP configuration structure

RKADK_VQE_MODE_E: Audio input sound quality enhancement enumeration

RKADK_MUXER_FILE_TYPE_E: Recording file type enumeration

RKADK_MUXER_PRE_RECORD_MODE_E: Pre-record mode enumeration

RKADK_MIC_TYPE_E: Audio device channel mode type enumeration

12.3.1 RKADK_DEFPARAM_PATH

【Description】

The default global ini configuration file path stores the configuration shared by each sensor and is used to restore the default configuration.

【Definition】

```
#define RKADK_DEFPARAM_PATH "/oem/usr/etc/rkadk_defsetting.ini"
```

12.3.2 RKADK_DEFPARAM_PATH_SENSOR_PREFIX

【Description】

The default sensor ini configuration file path stores the unique configuration of each sensor and is used to restore the default configuration.

【Definition】

```
#define RKADK_DEFPARAM_PATH_SENSOR_PREFIX "/oem/usr/etc/rkadk_defsetting_sensor"
```

【Notice】

- The sensor configuration file prefix defaults to rkadk_defsetting_sensor_n.ini, _n is the sensor camera ID, and the serial number starts from 0.

12.3.3 RKADK_PARAM_PATH

【Description】

Global ini configuration file path, which stores common configurations of all sensors.

【Definition】

```
#define RKADK_PARAM_PATH "/data/rkadk/rkadk_setting.ini"
```

12.3.4 RKADK_PARAM_PATH_SENSOR_PREFIX

【Description】

Sensor ini configuration file path, which stores the unique configuration of each sensor. Used to save new configurations when switching recording resolution, Codec type and other operations.

【Definition】

```
#define RKADK_PARAM_PATH_SENSOR_PREFIX "/data/rkadk/rkadk_setting_sensor"
```

【Notice】

- The sensor configuration file prefix defaults to rkadk_setting_sensor_n.ini, _n is the sensor camera ID and the serial number starts from 0.

12.3.5 RKADK_PARAM_TYPE_E

【Description】

Define parameter type enumeration type.

【Definition】

```
typedef enum {
  /* Cam Dependent Param */
  RKADK_PARAM_TYPE_FPS,             /* framerate */
  RKADK_PARAM_TYPE_GOP,             /* gop */
  RKADK_PARAM_TYPE_RES,             /* specify RKADK_PARAM_RES_E(record) */
  RKADK_PARAM_TYPE_PHOTO_RES,       /* specify RKADK_PARAM_RES_E(photo) */
  RKADK_PARAM_TYPE_CODEC_TYPE,      /* specify RKADK_PARAM_CODEC_CFG_S(record) */
  RKADK_PARAM_TYPE_BITRATE,         /* encode bitrate,specify
RKADK_PARAM_BITRATE_S */
  RKADK_PARAM_TYPE_FLIP,            /* bool */
  RKADK_PARAM_TYPE_MIRROR,          /* bool */
  RKADK_PARAM_TYPE_LDC,             /* ldc level [0,255] */
  RKADK_PARAM_TYPE_ANTIFOG,         /* antifog value, [0,10] */
  RKADK_PARAM_TYPE_WDR,             /* wdr level, [0,10] */
  RKADK_PARAM_TYPE_HDR,             /* 0: normal, 1: HDR2, 2: HDR3, [0,2] */
  RKADK_PARAM_TYPE_RECORD_TYPE,     /* specify RKADK_REC_TYPE_E */
  RKADK_PARAM_TYPE_RECORD_TIME,     /* specify RKADK_PARAM_REC_TIME_S, record
time(s) */
  RKADK_PARAM_TYPE_PRE_RECORD_TIME, /* pre record time, unit in second(s) */
  RKADK_PARAM_TYPE_PRE_RECORD_MODE, /* pre record mode, specify
MUXER_PRE_RECORD_MODE_E */
  RKADK_PARAM_TYPE_SPLITTIME,       /* specify RKADK_PARAM_REC_TIME_S, manual
splite time(s) */
```

```
  RKADK_PARAM_TYPE_FILE_CNT,        /* record file count, maximum
RECORD_FILE_NUM_MAX */
  RKADK_PARAM_TYPE_LAPSE_INTERVAL,  /* specify RKADK_PARAM_REC_TIME_S, lapse
interval(s) */
  RKADK_PARAM_TYPE_LAPSE_MULTIPLE,  /* lapse multiple */
  RKADK_PARAM_TYPE_JPEG_SLICE,      /* enable/disable JPEG slice */
  RKADK_PARAM_TYPE_SLICE_HEIGHT,    /* set JPEG slice height */

  /* COMM Dependent Param */
  RKADK_PARAM_TYPE_REC_MUTE,        /* record audio mute, bool */
  RKADK_PARAM_TYPE_VOLUME,          /* speaker volume, [0,100] */
  RKADK_PARAM_TYPE_MIC_VOLUME,      /* mic volume, [0,100] */
  RKADK_PARAM_TYPE_BUTT
} RKADK_PARAM_TYPE_E;
```

【Members】

Member Name	Description
RKADK_PARAM_TYPE_FPS	Frame rate
RKADK_PARAM_TYPE_GOP	I frame interval, RKADK_PARAM_GOP_S
RKADK_PARAM_TYPE_RES	Video resolution, RKADK_PARAM_RES_E
RKADK_PARAM_TYPE_PHOTO_RES	Photo resolution, RKADK_PARAM_RES_E
RKADK_PARAM_TYPE_CODEC_TYPE	Video encoding type, RKADK_PARAM_CODEC_CFG_S
RKADK_PARAM_TYPE_BITRATE	Bitrate, RKADK_PARAM_BITRATE_S
RKADK_PARAM_TYPE_FLIP	Flip up and down
RKADK_PARAM_TYPE_MIRROR	Mirror left and right
RKADK_PARAM_TYPE_LDC	Distortion correction[0,255]
RKADK_PARAM_TYPE_ANTIFOG	Dehaze[0,10]
RKADK_PARAM_TYPE_WDR	Wide dynamic range[0,10]
RKADK_PARAM_TYPE_HDR	High dynamic range imaging[0,10]
RKADK_PARAM_TYPE_RECORD_TYPE	Recording type, RKADK_REC_TYPE_E
RKADK_PARAM_TYPE_RECORD_TIME	Recording duration, recording main stream and sub-stream support setting different durations, RKADK_PARAM_REC_TIME_S
RKADK_PARAM_TYPE_PRE_RECORD_TIME	Pre-recording duration
RKADK_PARAM_TYPE_PRE_RECORD_MODE	Pre-recording mode, 0: no pre-recording, 1: manually split the pre-recording, 2: first file pre-recording, 3: all files pre-recording
RKADK_PARAM_TYPE_SPLITTIME	Manually split the recording duration. The main video stream and sub-stream of the video support setting different durations, RKADK_PARAM_REC_TIME_S
RKADK_PARAM_TYPE_FILE_CNT	Number of simultaneous recording files, the maximum is 2
RKADK_PARAM_TYPE_LAPSE_INTERVAL	Time-lapse recording duration, recording main stream and sub-stream support setting different durations, RKADK_PARAM_REC_TIME_S
RKADK_PARAM_TYPE_LAPSE_MULTIPLE	The multiple relationship between the playback duration of the time-lapse video file and the actual screen content time
RKADK_PARAM_TYPE_JPEG_SLICE	Whether to enable JPEG Slice

Member Name	Description
RKADK_PARAM_TYPE_SLICE_HEIGHT	JPEG Slice high
RKADK_PARAM_TYPE_REC_MUTE	Whether to enable recording mute
RKADK_PARAM_TYPE_VOLUME	Speaker volume[0,100]
RKADK_PARAM_TYPE_MIC_VOLUME	Microphone volume[0,100]

【Notice】

- Antifog, WDR, HDR and other camere hardware related settings, in addition to calling RKADK_PARAM_SetCamParam to set the ini, you also need to call the ISP corresponding interface to make it actually take effect.
- RKADK_PARAM_TYPE_LAPSE_MULTIPLE: the relationship between the playback duration of the time-lapse video file and the actual screen content time is related to the frame rate. For example, the frame rate of ordinary video is 30fps, and the time-lapse video is 1fps, then the multiple is 30.
- When RV1126/RV1109 switches resolutions and the photo resolution is not set to the maximum supported resolution of the sensor, it needs to be consistent with the Record main stream resolution.
- RKADK_PARAM_TYPE_REC_MUTE: Should be used together with RKADK_RECORD_GetAencChn and RK_MPI_AENC_SetMute.

【Related data types and interfaces】

RKADK_PARAM_GetCamParam

RKADK_PARAM_SetCamParam

RKADK_PARAM_GetCommParam

RKADK_PARAM_SetCommParam

12.3.6 RKADK_PARAM_RES_E

【Description】

Defines the playback event enumeration type.

【Definition】

```
typedef enum {
  RKADK_RES_720P = 0, /* 1280*720 */
  RKADK_RES_1080P,    /* 1920*1080 */
  RKADK_RES_1296P,    /* 2304*1296 */
  RKADK_RES_1440P,    /* 2560*1440 */
  RKADK_RES_1520P,    /* 2688*1520 */
  RKADK_RES_1600P,    /* 2560*1600 */
  RKADK_RES_1620P,    /* 2880*1616, height 8 alignment */
  RKADK_RES_1944P,    /* 2592*1944 */
  RKADK_RES_2160P,    /* 3840*2160 */
  RKADK_RES_BUTT,
} RKADK_PARAM_RES_E;
```

【Related data types and interfaces】

RKADK_PARAM_GetResolution

RKADK_PARAM_GetResType

12.3.7 RKADK_STREAM_TYPE_E

【Description】

Define the stream enumeration type.

【Definition】

```
typedef enum {
  RKADK_STREAM_TYPE_SENSOR,
  RKADK_STREAM_TYPE_VIDEO_MAIN,
  RKADK_STREAM_TYPE_VIDEO_SUB,
  RKADK_STREAM_TYPE_SNAP,
  RKADK_STREAM_TYPE_PREVIEW,
  RKADK_STREAM_TYPE_LIVE,
  RKADK_STREAM_TYPE_DISP,
  RKADK_STREAM_TYPE_BUTT
} RKADK_STREAM_TYPE_E;
```

【Members】

Member Name	Description
RKADK_STREAM_TYPE_VIDEO_MAIN	Video main stream
RKADK_STREAM_TYPE_VIDEO_SUB	Video sub-stream
RKADK_STREAM_TYPE_SNAP	Take photos
RKADK_STREAM_TYPE_PREVIEW	Remote preview
RKADK_STREAM_TYPE_LIVE	Live broadcast
RKADK_STREAM_TYPE_DISP	Local Preview
RKADK_STREAM_TYPE_SENSOR	Sensor

【Related data types and interfaces】

RKADK_PARAM_GetVencChnId

12.3.8 RKADK_PARAM_CODEC_CFG_S

【Description】

Define the encoding type configuration structure.

【Definition】

```
typedef struct {
  RKADK_STREAM_TYPE_E enStreamType;
  RKADK_CODEC_TYPE_E enCodecType;
} RKADK_PARAM_CODEC_CFG_S;
```

【Members】

Member Name	Description
enStreamType	stream type
enCodecType	encoding type

【Related data types and interfaces】

RKADK_CODEC_TYPE_E

RKADK_STREAM_TYPE_E

12.3.9 RKADK_PARAM_BITRATE_S

【Description】

Define the bitrate configuration structure.

【Definition】

```
typedef struct {
  RKADK_STREAM_TYPE_E enStreamType;
  RKADK_U32 u32Bitrate;
} RKADK_PARAM_BITRATE_S;
```

【Members】

Member Name	Description
enStreamType	stream type
u32Bitrate	Bitrate

【Related data types and interfaces】

RKADK_STREAM_TYPE_E

12.3.10 RKADK_PARAM_REC_TIME_S

【Description】

Define the recording duration configuration structure.

【Definition】

```
typedef struct {
  RKADK_STREAM_TYPE_E enStreamType;
  RKADK_U32 time;
} RKADK_PARAM_REC_TIME_S;
```

【Members】

Member Name	Description
enStreamType	Stream type
time	Recording duration

【Related data types and interfaces】

RKADK_STREAM_TYPE_E

12.3.11 RKADK_PARAM_GOP_S

【Description】

Define the VENC I frame interval configuration structure.

【Definition】

```
typedef struct {
  RKADK_STREAM_TYPE_E enStreamType;
  RKADK_U32 u32Gop;
} RKADK_PARAM_GOP_S;
```

【Members】

Member Name	Description
enStreamType	stream type
u32Gop	I frame interval

【Related data types and interfaces】

RKADK_STREAM_TYPE_E

12.3.12 RKADK_VQE_MODE_E

【Description】

Defines the audio input sound quality enhancement enumeration type

【Definition】

```
typedef enum {
  RKADK_VQE_MODE_AI_TALK = 0,
  RKADK_VQE_MODE_AI_RECORD,
  RKADK_VQE_MODE_BUTT
} RKADK_VQE_MODE_E;
```

【Members】

Member Name	Description
RKADK_VQE_MODE_AI_TALK	Enable AEC, ANR, AGC
RKADK_VQE_MODE_AI_RECORD	Enable ANR

12.3.13 RKADK_MUXER_FILE_TYPE_E

【Description】

Define recording file type enumeration

【Definition】

```
typedef enum rkMUXER_TYPE_E {
  RKADK_MUXER_TYPE_MP4 = 0,
  RKADK_MUXER_TYPE_MPEGTS,
  RKADK_MUXER_TYPE_FLV,
  RKADK_MUXER_TYPE_BUTT
} RKADK_MUXER_FILE_TYPE_E;
```

【Members】

Member Name	Description
RKADK_MUXER_TYPE_MP4	MP4
RKADK_MUXER_TYPE_MPEGTS	Reserved
RKADK_MUXER_TYPE_FLV	FLV

12.3.14 RKADK_MUXER_PRE_RECORD_MODE_E

【Description】

Define prerecorded mode enum

【Definition】

```
typedef enum {
  RKADK_MUXER_PRE_RECORD_NONE = 0,
  RKADK_MUXER_PRE_RECORD_MANUAL_SPLIT, /* manual split file prerecord */
  RKADK_MUXER_PRE_RECORD_SINGLE        /* first file prerecord */
} RKADK_MUXER_PRE_RECORD_MODE_E;
```

【Members】

Member Name	Description
RKADK_MUXER_PRE_RECORD_NONE	No pre-recording
RKADK_MUXER_PRE_RECORD_MANUAL_SPLIT	Manually split file pre-recording
RKADK_MUXER_PRE_RECORD_SINGLE	First file pre-recorded

12.3.15 RKADK_MIC_TYPE_E

【Description】

Define the audio device channel mode type

【Definition】

```
typedef enum {
  RKADK_MIC_TYPE_LEFT = 0,
  RKADK_MIC_TYPE_RIGHT,
  RKADK_MIC_TYPE_BOTH,
  RKADK_MIC_TYPE_BUTT
} RKADK_MIC_TYPE_E;
```

【Members】

Member Name	Description
RKADK_MIC_TYPE_LEFT	Left channel sound
RKADK_MIC_TYPE_RIGHT	Right channel sound
RKADK_MIC_TYPE_BOTH	Dual-channel

12.4 INI File Introduction

12.4.1 Global INI Configuration File

```
[version]
version                         = 2.2.0      /* Version */

/* common parameters */
[common]
sensor_count                    = 2          /* Number of Sensors*/
rec_mute                        = FALSE      /* Whether to enable video mute */
speaker_volume                  = 80         /* Speaker volume,[0,100] */
mic_volume                      = 80         /* Microphone volume,[0,100] */
vpss_devcie                     = 1          /* VPSS hardware device type, 0:GPU,
1:RGA */

/* Audio parameters */
[audio]
ai_audio_node                   = hw:0,0     /* AI device node */
ao_audio_node                   = hw:0,0     /* AO device node */
ai_depth                        = 1          /* AI depth */
bit_width                       = 1          /* Sampling accuracy */
channels                        = 1          /* Channels */
mic_type                        = 0          /* Audio device channel mode,
specifically RKADK_MIC_TYPE_E, 0: left channel, 1: right channel, 2: dual
channel*/
samplerate                      = 16000      /* Sample rate   */
samples_per_frame               = 576        /* Number of samples per frame*/
bitrate                         = 64000      /* Bitrate */
vqe_mode                        = 1          /* Configure audio input sound
quality enhancement, specifically RKADK_VQE_MODE_E*/
vqe_config_path                 = /oem/usr/share/vqefiles/config_aivqe.json /*vqe
configuration file path */
codec_type                      = 8          /* Record and Live Audio encoding
types, adapted to MP3 by default, specifically RKADK_CODEC_TYPE_E */
```

12.4.2 Sensor INI Configuration Files

```
[sensor]
used_isp                        = TRUE       /* Has the Sensor used ISP? */
max_width                       = 2688       /* Max width */
max_height                      = 1520       /* Max height    */
framerate                       = 30         /* Framerate*/
flip                            = FALSE      /* Flip up and down*/
mirror                          = FALSE      /* Mirror left and right */
ldc                             = 0          /* Lens distortion correction, [0,255]
*/
wdr                             = 0          /* Wide dynamic range, [0,10] */
hdr                             = 0          /* High dynamic range imaging, [0,10]
*/
antifog                         = 0          /* Remove fog, [0,10] */
```

```
enable_wrap                     = FALSE      /* Whether the VI enables wrap */
wrap_buf_line                   = 1620       /* The line height of the wrap buffer
*/

/* VI channel configuration parameters */
[vi.0]
chn_id                          = 0                  /* Channel ID */
device_name                     = rkispp_m_bypass    /* Video node path*/
buf_cnt                         = 4                  /* The total number of buffer
blocks in the output channel */
depth                           = 0                  /* VI depth depth  */
width                           = 2688               /* Video width   */
height                          = 1520               /* Video height  */
pix_fmt                         = FBC0               /* VI output format*/
module                          = RECORD_MAIN|PHOTO  /* The usage module of this
VI,Options: NONE/RECORD_MAIN/RECORD_SUB/PREVIEW/PHOTO/LIVE/DISP */

[vi.1]
chn_id                          = 1
device_name                     = rkispp_scale0
buf_cnt                         = 4
depth                           = 0
width                           = 0
height                          = 0
pix_fmt                         = NV12
module                          = RECORD_MAIN|PHOTO

[vi.2]
chn_id                          = 2
device_name                     = rkispp_scale1
buf_cnt                         = 2
depth                           = 0
width                           = 0
height                          = 0
pix_fmt                         = NV12
module                          = NONE

[vi.3]
chn_id                          = 3
device_name                     = rkispp_scale2
buf_cnt                         = 4
depth                           = 0
width                           = 848
height                          = 480
pix_fmt                         = NV12
module                          = RECORD_SUB|PREVIEW|LIVE|DISP

/* Record parameters */
[record]
record_type                     = 0          /* Recording type, specifically
RKADK_REC_TYPE_E*/
file_type                       = 0          /* Video file type, specifically
RKADK_MUXER_FILE_TYPE_E */
pre_record_time                 = 0          /* Pre-recording duration */
pre_record_mode                 = 0          /* Pre-recorded mode */
```

```
lapse_multiple                  = 30         /* The multiple relationship between
the playback duration of a time-lapse video file and the actual screen content
duration */
file_num                        = 1          /* Number of files recorded
simultaneously, maximum 2 */
switch_res                      = TRUE       /* Whether to switch resolution */
enable_audio                    = TRUE       /* Whether to enable recording */

/* Record 0 VENC parameters of Main stream  */
[record.0]
record_time                     = 60         /* Recording duration*/
splite_time                     = 60         /* Manually divide video duration */
lapse_interval                  = 60         /* Time-lapse recording duration */
width                           = 2688       /* Video width */
height                          = 1520       /* Video height */
bufsize                         = 10379776   /* The buffer size of the code stream
*/
framerate                       = 30         /* Venc framerate */
bitrate                         = 8294400    /* Bitrate */
gop                             = 30         /* I frame interval */
profile                         = 100        /* Encoder profile */
codec_type                      = 0          /* Encoding type,
specificallyRKADK_CODEC_TYPE_E */
venc_chn                        = 0          /* VENC channel ID */
vpss_grp                        = 0          /* VPSS GROUP ID*/
vpss_chn                        = 0          /* VPSS channel ID */
post_aiisp                      = FALSE      /* Whether to enbale Post AI ISP */
rc_mode                         = CBR        /* Encoding protocol type, supports
CBR, VBR, AVBR*/
max_qp                          = -1         /* QP maximum value, value range [1,
51], -1 is the default value */
min_qp                          = -1         /* QP minimum value, value range [1,
max_qp], -1 is the default value */
i_min_qp                        = -1
i_frame_min_qp                  = -1
full_range                      = TRUE
scaling_list                    = FALSE
hier_qp_en                      = FALSE
hier_qp_delta                   = -3,0,0,0
hier_frame_num                  = 3,0,0,0

/* VENC parameter of the substream Record 1, when file_num = 1, rec.1 does not
need to be configured*/
[record.1]
record_time                     = 60
splite_time                     = 60
lapse_interval                  = 60
width                           = 848
height                          = 480
bufsize                         = 2367488
bitrate                         = 407040
framerate                       = 30
gop                             = 30
profile                         = 100
codec_type                      = 0
venc_chn                        = 1
```

```
vpss_grp                        = 0          /* VPSS GROUP ID*/
vpss_chn                        = 0          /* VPSS channel ID */
post_aiisp                      = FALSE      /* Whether to enbale Post AI ISP */
rc_mode                         = VBR
max_qp                          = 48
min_qp                          = 8
i_min_qp                        = -1
i_frame_min_qp                  = -1
full_range                      = TRUE
scaling_list                    = TRUE
hier_qp_en                      = TRUE
hier_qp_delta                   = -3,0,0,0
hier_frame_num                  = 3,0,0,0

/* Photo VENC parameters */
[photo]
image_width                     = 3840       /* Image width */
image_height                    = 2160       /* Image_height  */
venc_chn                        = 2          /* VENC channel ID */
vpss_grp                        = 0          /* VPSS GROUP ID*/
vpss_chn                        = 0          /* VPSS channel ID */
post_aiisp                      = FALSE      /* Whether to enbale Post AI ISP */
enable_combo                    = FALSE      /* Enable the Combo attribute of the
encoding channel */
combo_venc_chn                  = 0          /* Combo data source channel */
qfactor                         = 50         /* For the detailed meaning, please
refer to the RFC2435 protocol. The system default value is 70 and the value range
is [1, 99] */
switch_res                      = TRUE       /* Whether to switch resolution */
jpeg_slice                      = FALSE      /* Whether to enable JPEG Slice */
slice_height                    = 0          /* The height of JPEG Slice must not
be greater than max_slice_height*/
max_slice_width                 = 0          /* Maximum width of JPEG Slice */
max_slice_height                = 0          /* Maximum height of JPEG Slice*/

/* Remote preview of VENC parameters */
[preview]
width                           = 848        /* Video width */
height                          = 480        /* Video height */
bufsize                         = 2367488
bitrate                         = 407040     /* Bitrate */
framerate                       = 30         /* Venc framerate  */
gop                             = 30         /* I frame interval */
profile                         = 100        /* Encoder profile */
codec_type                      = 0          /* Coding type, specifically
RKADK_CODEC_TYPE_E */
venc_chn                        = 1          /* Venc channel ID */
vpss_grp                        = 0          /* VPSS GROUP ID */
vpss_chn                        = 0          /* VPSS  channel ID */
post_aiisp                      = FALSE      /* Whether to enbale Post AI ISP */
rc_mode                         = VBR        /* Encoding protocol type, supports
CBR, VBR, AVBR */
max_qp                          = 48         /* QP maximum value, value range [1,
51] */
min_qp                          = 8          /* QP minimum value, value range [1,
min_qp] */
```

```
i_min_qp                        = -1
i_frame_min_qp                  = -1
full_range                      = TRUE
scaling_list                    = TRUE
hier_qp_en                      = TRUE
hier_qp_delta                   = -3,0,0,0
hier_frame_num                  = 3,0,0,0

/* Live VENC parameters */
[live]
width                           = 1280
height                          = 720
bufsize                         = 2367488
bitrate                         = 4194304
framerate                       = 30
gop                             = 30
profile                         = 100
codec_type                      = 0
venc_chn                        = 1
vpss_grp                        = 0
vpss_chn                        = 0
post_aiisp                      = FALSE
rc_mode                         = VBR
max_qp                          = 48
min_qp                          = 8
i_min_qp                        = -1
i_frame_min_qp                  = -1
full_range                      = TRUE
scaling_list                    = TRUE
hier_qp_en                      = TRUE
hier_qp_delta                   = -3,0,0,0
hier_frame_num                  = 3,0,0,0

/* Local preview parameters */
[display]
x                               = 0             /* Display X coordinate */
y                               = 0             /* Display y coordinate */
width                           = 720           /* Display the width  */
height                          = 1280          /* Display the height */
rotaion                         = 90            /* Degrees of rotation,
Options: 0:0, 1:90, 2:180, 3:270 */
vpss_grp                        = 0             /* VPSS GROUP ID */
vpss_chn                        = 0             /* VPSS channel ID */
img_type                        = RGB888        /* Video output format */
vo_device                       = 0             /* Display output device number
*/
vo_layer                        = 0             /* Video output video layer
number */
vo_chn                          = 0             /* VO channel ID */
frame_rate                      = 30            /* VO frame rate */
intf_type                       = default       /* Show interface type,
Options:  MIPI, HDMI, EDP, VGA, DP, HDMI_EDP, LCD, default*/
splice_mode                     = RGA           /* Splice mode, Options: RGA,
GPU, BYPASS */
```

12.4.3 INI Configuration Precaution

- Configure the default INI file path through environment variables, for example: export rkadk_default_ini_path=/oem/usr/etc.
- Set the INI file path through RKADK_PARAM_Init API.
- The versions in rkadk_defsetting.ini and rkadk_setting.ini must be consistent, otherwise version detection will fail and the default INI configuration will be used. When the SDK is updated, the ini parameters may increase or decrease, so you need to pay attention at this time.
- The sensor_count represents the actual number of sensors used. It is set according to actual use and cannot be greater than RKADK_MAX_SENSOR_CNT. Currently, 3 sensors have been actually debugged.
- If the sensor uses ISP, `used_isp` must be configured to TRUE, and scaling is achieved directly by configuring the ISP node resolution. There is no need to configure the VPSS channel. vpss_grp and vpss_chn are both configured to 0; if the sensor does not use ISP, `used_isp` must be configured to FALSE, and configured the corresponding vpss_grp and vpss_chn, VPSS is used internally for scaling.
- When the resolution of recording, remote preview, live broadcast, etc. are the same, it is recommended to reuse VENC (VENC parameters are configured to be the same) to improve bandwidth and memory utilization.
- When VENC is reused or the VENC resolution is the same, it is recommended to reuse VPSS channels to improve bandwidth utilization.
- The gop is recommended to be configured the same as framerate to ensure that there is one I frame every second.

13. Examples

Function examples are provided below, and precautions for use are as follows:

- Before running the examples, make sure that no other application occupies the node used by the example, such as mediaserver and ispserver.
- The default parameters of the examples are adapted to RK EVBs. When the hardware are different from our EVB, customers need to specify command parameters themselves or adjust the code.

13.1 rkadk_record_test

【Description】

Record test.

【Code path】

rkadk/examples/rkadk_record_test.c

【Quick use】

```
./rkadk_record_test
```

【Option】

Options	Description	Default
-a	Enable the built-in ISP function. Enter this option to enable the built-in ISP function. If there are no parameters, the default value will be used. The parameter is the path to the folder where the aiq file is located.	/etc/iqfiles
-I	Camera ID	0
-m	Enable dual sensor test, options: 0(isp sensor), 1(all isp sensors), 2(isp+ahd sensors)	0
-p	The ini configuration file directory path	/data/rkadk
-k	Whether the video file is sliced into I frames	Not sliced

13.2 rkadk_photo_test

【Description】

Photo testing.

【Code path】

/rkadk/examples/rkadk_photo_test.c

【Quick use】

```
./rkadk_photo_test
```

【Option】

Options	Description	Default
-a	Enable the built-in ISP function. Enter this option to enable the built-in ISP function. If there are no parameters, the default value will be used. The parameter is the path to the folder where the aiq file is located.	/etc/iqfiles
-I	Camera ID	0
-p	The ini configuration file directory path	/data/rkadk
-t	Data type of the obtained JPG image	NV12
-m	Enable dual sensor test, options: 0(isp sensor), 1(all isp sensors), 2(isp+ahd sensors)	0
-o	The osd file path	NULL
-W	The osd width	0
-H	The osd high	0

13.3 rkadk_stream_test

【Description】

Get the audio stream, encode it, and output it to a file; get the video stream, encode it, and output it to a file.

【Code path】

/rkadk/examples/rkadk_stream_test.c

【Quick use】

```
./rkadk_stream_test
```

【Option】

Options	Description	Default
-a	Enable the built-in ISP function. Enter this option to enable the built-in ISP function. If there are no parameters, the default value will be used. The parameter is the path to the folder where the aiq file is located.	/etc/iqfiles
-I	Camera ID	0
-M	Test mode: audio, video	audio
-e	Encoding type	pcm
-o	Output file path	/tmp/ai.pcm
-p	The ini configuration file directory path	/data/rkadk
-m	Enable dual Sensor test, options: 0(isp sensor), 1(all isp sensors), 2(isp+ahd sensors)	0

13.4 rkadk_player_test

【Description】

Local file playback test.

【Code path】

/rkadk/examples/rkadk_player_test.c

【Quick use】

```
./rkadk_player_test
```

【Option】

Options	Description	Default
-i	Playback file path	/etc/bsa_file/8k8bpsMono.wav
-x	Video display starting x coordinate	0
-y	Video display starting y coordinate	0
-W	Video display width	Screen physical width
-H	Video display height	Screen physical height
-r	Video rotation angle, option: 0, 90, 180, 270	0
-p	The ini configuration file directory path	/data/rkadk
-m	Video mirror	disable
-f	Video flip	disable
-a	Whether to enable audio playback, option: 0 (disable), 1 (enable)	1
-v	Whether to enable Video playback	disable
-s	Set layer synthesis method, option: 0(RGA), 1(GPU), 2(ByPass)	0
-P	Screen display pixel format, option: 0(RGB888), 1(NV12), 2(RGB565)	0
-I	Display interface type, option: 0(DEFAILT), 1(MIPI), 2(LCD)	1106: 0, other platforms: 1
-F	Refresh frame rate	30
-t	RTSP transmission protocol, option: 0(udp), 1(tcp)	0
-b	Enable black screen after playing	disable
-T	RTSP socket I/O operation timeout exit time, unit ms	Blocking, no timeout
-l	VO layer ID	0
-O	Vdec output buffer number	3 [1, 8]
-D	Enable third-party demuxer library	disable

【Notice】

- When playing video files, -v is required to enable Video playback.
- Mirror/flip cannot be set at the same time as rotation.

13.5 rkadk_thumb_test

Get file thumbnail test.

【Code path】

/rkadk/examples/rkadk_thumb_test.c

【Quick use】

```
Get MP4 thumbnail: ./rkadk_thumb_test -i test_file.mp4
Get JPG thumbnail: ./rkadk_thumb_test -i test_file.jpg -f jpg -t MPF1
```

【Option】

Options	Description	Default
-i	Test file path	None
-f	File format: mp4, jpg	mp4
-t	JPG thumbnail types: DCF, MPF1, MPF2	DCF
-T	Output thumbnail type: JPG, NV12, RGB565, RGB888	JPG
-W	Thumbnail width	Get from ini
-H	Thumbnail height	Get from ini

13.6 rkadk_rtsp_test

RTSP live test.

【Code path】

/rkadk/examples/rkadk_rtsp_test.c

【Quick use】

```
1. Start wifi on the board end
2. Run the board end./rkadk_rtsp_test
3. Open VLC on PC -> Media -> Open network streaming -> Enter the network URL:
rtsp://board end ip address/live/main_stream
```

【Option】

Options	Description	Default
-a	Enable the built-in ISP function. Enter this option to enable the built-in ISP function. If there are no parameters, the default value will be used. The parameter is the path to the folder where the aiq file is located.	/etc/iqfiles
-I	Camera ID	0
-p	The ini configuration file directory path	/data/rkadk
-o	The osd file path	/userdata/rkadk_ARGB8888

13.7 rkadk_rtmp_test

RTMP live test.

【Code path】

/rkadk/examples/rkadk_rtmp_test.c

【Quick use】

```
1. Start wifi on the board
2. Run ./rkadk_rtmp_test on the board
3. Open VLC on the PC -> Media -> Open network streaming -> Enter the network
URL: rtmp://The ip address of the board::1935/live/substream
```

【Option】

Options	Description	Default
-a	Enable the built-in ISP function. Enter this option to enable the built-in ISP function. If there are no parameters, the default value will be used. The parameter is the path to the folder where the aiq file is located.	/etc/iqfiles
-I	Camera ID	0
-p	The ini configuration file directory path	/data/rkadk

13.8 rkadk_storage_test

【Description】

Storage module testing.

【Code path】

/rkadk/examples/rkadk_storage_test.c

【Quick use】

```
./rkadk_storage_test
```

【Option】

None

【Notice】

- This test generates an mp4 file with all 0s written in it to test functions and interfaces such as automatic deletion and file list obtaining. The mp4 file has no actual data and cannot be played.

13.9 rkadk_disp_test

Local preview testing.

【Code path】

/rkadk/examples/rkadk_disp_test.c

【Quick use】

```
./rkadk_disp_test
```

【Option】

Options	Description	Default
-a	Enable the built-in ISP function. Enter this option to enable the built-in ISP function. If there are no parameters, the default value will be used. The parameter is the path to the folder where the aiq file is located.	/etc/iqfiles
-I	Camera ID	0
-p	The ini configuration file directory path	/data/rkadk

13.10 rkadk_ui_test

UI overlay testing.

【Code path】

/rkadk/examples/rkadk_ui_test.c

【Quick use】

```
./rkadk_ui_test
```

【Option】

Options	Description	Default
-a	Enable the built-in ISP function. Enter this option to enable the built-in ISP function. If there are no parameters, the default value will be used. The parameter is the path to the folder where the aiq file is located.	/etc/iqfiles
-I	Camera ID	0
-p	The ini configuration file directory path	/data/rkadk
-W	Display area width	720
-H	Display area height	1280
-f	Screen display pixel format, option: 0(RGB888), 1(NV12), 2(RGB565), 3(RGB444)	0