



HiMAPI V1.0 媒体处理

## 开发参考

文档版本 00B01

发布日期 2016-11-25

**版权所有 © 深圳市海思半导体有限公司 2016。保留一切权利。**

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

## **商标声明**



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

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

## **注意**

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

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

## **深圳市海思半导体有限公司**

地址：                    深圳市龙岗区坂田华为基地华为电气生产中心                    邮编：518129

网址：                    <http://www.hisilicon.com>

客户服务电话：          +86-755-28788858

客户服务传真：          +86-755-28357515

客户服务邮箱：          [support@hisilicon.com](mailto:support@hisilicon.com)



## 目 录

前 言.....	1
----------	---



# 前言

## 概述

本文为使用 MAPI 进行开发的程序员而写，目的是供您在开发过程中查阅中间件的各种参考信息，包括 API、头文件、错误码等。

本文档描述中间件的各个 API 的使用方法，以及相关的数据结构和错误码。

## 产品版本

与本文档相对应的产品版本如下。

产品名称	产品版本	操作系统
Hi3518E	V200	Huawei LiteOS
Hi3516C	V200	Huawei LiteOS



说明

未有特殊说明，Hi3516CV200 和 Hi3518EV200 一致。

## 读者对象

本文档主要适用于以下工程师：






- 技术支持工程师
- 软件开发工程师

## 约定

### 符号约定

在本文中可能出现下列标志，它们所代表的含义如下。



符号	说明
 <b>危险</b>	表示有高度潜在危险，如果不能避免，会导致人员死亡或严重伤害。
 <b>警告</b>	表示有中度或低度潜在危险，如果不能避免，可能导致人员轻微或中等伤害。
 <b>注意</b>	表示有潜在风险，如果忽视这些文本，可能导致设备损坏、数据丢失、设备性能降低或不可预知的结果。
 窍门	表示能帮助您解决某个问题或节省您的时间。
 说明	表示是正文的附加信息，是对正文的强调和补充。

## 通用格式约定

格式	说明
宋体	正文采用宋体表示。
黑体	一级、二级、三级标题采用 <b>黑体</b> 。
楷体	警告、提示等内容一律用 <b>楷体</b> ，并且在内容前后增加线条与正文隔离。
“Terminal Display” 格式	“Terminal Display” 格式表示屏幕输出信息。此外，屏幕输出信息中夹杂的用户从终端输入的信息采用加粗字体表示。
“ ”	用双引号表示文件路径。如 “C:\Program Files\Huawei”。

## 命令行格式约定

格式	意义
<b>粗体</b>	命令行关键字（命令中保持不变、必须照输的部分）采用 <b>加粗</b> 字体表示。
<i>斜体</i>	命令行参数（命令中必须由实际值进行替代的部分）采用 <i>斜体</i> 表示。
[ ]	表示用 “[ ]” 括起来的部分在命令配置时是可选的。
{ x   y   ... }	表示从两个或多个选项选取一个。
[ x   y   ... ]	表示从两个或多个选项选取一个或者不选。



格式	意义
{ x   y   ... } *	表示从两个或多个选项中选择多个，最少选取一个，最多选取所有选项。
[ x   y   ... ] *	表示从两个或多个选项中选择多个或者不选。

## 修订记录

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

修订日期	版本	修订说明
2016-11-25	00B01	第 1 次临时版本发布。



## 目 录

1 系统概述.....	1-1
1.1 概述.....	1-1
1.2 系统架构.....	1-1
1.3 MAPI 层业务流程.....	1-2



## 插图目录

图 1-1 Hi35xx 典型的系统层次图.....	1-2
图 1-2 MAPI 层内部处理流程图.....	1-3





# 1 系统概述

## 1.1 概述

海思提供的媒体处理软件平台(Media Process Platform,简称 MPP), 可支持应用软件快速开发。该平台对应用软件屏蔽了芯片相关的复杂的底层处理, 并对应用软件直接提供 MPI (MPP Programe Interface) 接口完成相应功能。该平台支持应用软件快速开发以下功能: 输入视频捕获、H.265/H.264/MJPEG/JPEG/MPEG4 编码、H.264/MPEG4/MPEG2 解码、视频输出显示、视频图像前处理 (包括去噪、增强、锐化、Deinterlace)、编码码流叠加 OSD、视频侦测分析、智能分析、音频捕获及输出、音频编解码等功能。

MPI 接口在使用过程中, 很多用户反馈 MPI 接口太多, 调用流程繁琐。为了简化用户的使用流程, 对 MPI 接口和功能模块做提炼和抽象, 形成了 MAPI (Media Application Programming Interface) 接口。同时 MAPI 支持多 CPU 异构的方案, 方便用户开发基于多 CPU 异构的产品。

## 1.2 系统架构

MPP 平台支持的典型的系统层次如图 1-1 所示, 主要分为以下层次:

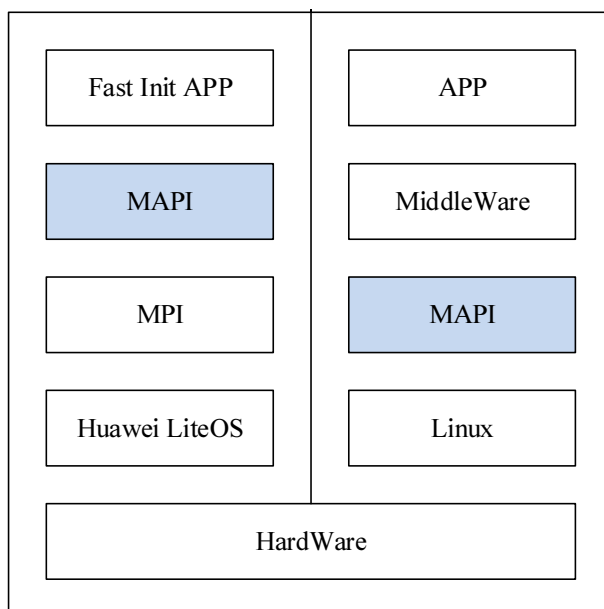
- 硬件层  
硬件层由 Hi35xx 芯片加上必要的外围器件构成。外围器件包括 Flash、DDR (Double Data-Rate)、视频 Sensor 或 AD、音频 AD 等。
- 操作系统层  
基于 Linux /Huawei LiteOS 的 OS 系统。
- MPI 层  
提供媒体处理平台的基础接口, 屏蔽硬件处理细节, 为上层提供 API 完成相应功能。
- MAPI 层  
对 MPI 接口做简化和抽象之后形成的 API。屏蔽各个芯片间 MPI 接口的差异, 支持多 CPU 异构的一套通用接口。
- Fast Init APP



当需要快速开机录像或者开机抓拍的场景，利用 Huawei LiteOS 快速启动的优势，在 Huawei LiteOS 端部署开机启动的用户业务。

- APP  
基于海思媒体处理平台及其他驱动，由用户开发的应用软件系统。

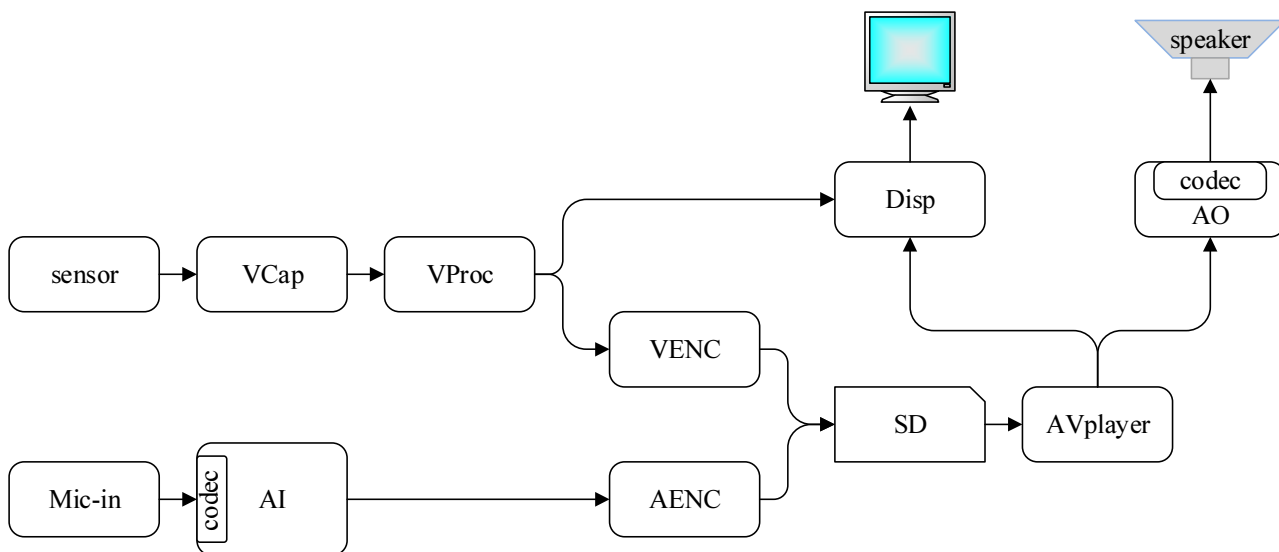
图1-1 Hi35xx 典型的系统层次图



## 1.3 MAPI 层业务流程

MAPI 层内部处理流程如图 1-2 所示，主要分为视频捕获（VCap）、视频处理（VProc）、视频编码（VENC）、显示预览（DISP）、音频输入(AI)、音频输出(AO)、音频编码（AENC）等模块。主要的处理流程介绍如下：

图1-2 MAPI 层内部处理流程图



- VCap 模块捕获视频图像，可对其做 bayer 缩放，叠加 OSD 等处理，并输出 YUV 图像数据。
- VProc 模块接收 VCap 发送过来的图像，可对图像进行去噪、缩放、旋转、拍照算法合成等处理，并实现同源输出多路不同分辨率的图像数据用于编码、预览或抓拍。
- VENC 模块接收 VProc 处理后输出的图像数据，然后按不同协议 H.264/H.265/JPEGE 等进行编码并输出相应码流或者照片。
- DISP 模块接收 VProc 处理后的图像输出到显示设备，或者接收 AVplayer 解码后的图像输出显示。
- AI 模块捕获音频数据，然后 AENC 模块支持按多种音频协议对其进行编码，最后输出音频码流。
- 用户录制到 SD 卡上的码流或者照片可以通过 AVplayer 解码回放，解码后的视频数据送到 DISP 做显示输出，解码后的音频数据送到 AO 模块播放声音。



## 目 录

<b>2 视频采集</b>	<b>2-1</b>
2.1 概述	2-1
2.2 重要概念	2-1
2.4 输入时序配置	2-1
2.5 API 参考	2-2
2.6 数据类型	2-28
2.7 错误码	2-57



## 表格目录

表 2-1 设置 VCAP 高级属性 enCMD 对应的结构 .....	2-21
表 2-2 获取 VCAP 高级属性 enCMD 对应的结构 .....	2-23
表 2-3 视频采集 API 错误码 .....	2-57



# 2 视频采集

## 2.1 概述

视频采集（VideoCapture，简称 VCAP）模块实现的功能：通过 MIPI Rx(含 MIPI 接口、LVDS 接口和 HISPI 接口)接收视频数据，将前端输入的原始数据送入到内部的图像信号处理单元(ISP)进行处理，并最终输出 YUV 数据给视频后处理模块 (VideoProcess，简称 VPROC)。

## 2.2 重要概念

- 视频采集单元  
视频采集单元支持若干种时序输入，负责对时序进行解析，并将图像送入到内部的 ISP 模块进行图像处理。  
Hi3516CV200 芯片支持 1 个 VCAP，对应的 Handle 为 0。
- 镜头畸变校正（LDC）  
镜头畸变校正，一些镜头容易产生图像畸变，需要根据畸变程度对其图像进行校正。
- 图像像素格式说明  
VCAP 输出给后端的 YUV 图像格式包括 semi-planar4:2:2 和 semi-planar4:2:0。
- OSD  
OSD 是指将 BMP 图片数据叠加到 YUV 数据上，用户可用于叠加 LOGO、时间、GPS 等信息。
- BAS  
Bayer scale，即 Bayer 域缩放，支持 VCAP1(仅当 VCAP1 作为视频数据通路时有效)进行 Bayer scale。即 VCAP1 可以对前端输入数据的宽高进行 1 倍，1/2 倍，1/3 倍缩放，否则失败。

## 2.4 输入时序配置

目前支持的前端输入时序为 sensor 输入，具体的配置规则请参看《Sensor 调试指南》。



## 2.5 API 参考

视频采集单元(VCAP)实现配置前端输入时序、接收时序输入并进行解析、最后进行图像处理并输出等功能。

该功能模块提供以下 MAPI:

- [HI\\_MAPI\\_Sensor\\_SetAttr](#): 设置 sensor 的模式。
- [HI\\_MAPI\\_Sensor\\_GetAttr](#): 获取 sensor 的模式。
- [HI\\_MAPI\\_VCap\\_SetAttr](#): 设置 VCap 的属性。
- [HI\\_MAPI\\_VCap\\_GetAttr](#): 获取 VCap 的属性。
- [HI\\_MAPI\\_VCap\\_Isp\\_Start](#): 启动指定 VCap 的 Isp 处理单元。
- [HI\\_MAPI\\_VCap\\_Isp\\_Stop](#): 停止指定 VCap 的 Isp 处理单元。
- [HI\\_MAPI\\_VCap\\_Isp\\_GetIspDev](#): 获取指定 VCap 的 Isp 设备。
- [HI\\_MAPI\\_VCap\\_Start](#): 启动 VCap。
- [HI\\_MAPI\\_VCap\\_Stop](#): 停止 VCap。
- [HI\\_MAPI\\_VCap\\_Trigger](#): 触发抓拍。
- [HI\\_MAPI\\_VCap\\_StopTrigger](#): 在连续抓拍多张图片的情况下终止抓拍。
- [HI\\_MAPI\\_VCap\\_OSD\\_SetAttr](#): 设置 OSD 属性。
- [HI\\_MAPI\\_VCap\\_OSD\\_GetAttr](#): 获取 OSD 属性。
- [HI\\_MAPI\\_VCap\\_OSD\\_Start](#): 启动 OSD。
- [HI\\_MAPI\\_VCap\\_OSD\\_Stop](#): 停止 OSD。
- [HI\\_MAPI\\_VCap\\_SetExifInfo](#): 设置 Exif 信息。
- [HI\\_MAPI\\_VCap\\_GetExifInfo](#): 获取 Exif 信息。
- [HI\\_MAPI\\_VCap\\_SetAttrEx](#): 设置 VCAP 的高级属性, 包括曝光属性、白平衡、亮度、饱和度、锐度、LDC 等属性。
- [HI\\_MAPI\\_VCap\\_GetAttrEx](#): 获取 VCAP 的高级属性, 包括曝光属性、白平衡、亮度、饱和度、锐度、LDC 等属性。
- [HI\\_MAPI\\_VCap\\_EnableDumpRaw](#): 启用 Dump Raw 数据功能。
- [HI\\_MAPI\\_VCap\\_DisableDumpRaw](#): 禁用 Dump Raw 数据功能。
- [HI\\_MAPI\\_VCap\\_DumpRaw](#): Dump Raw 数据。

### HI\_MAPI\_Sensor\_SetAttr

#### 【描述】

设置要启动的 Sensor 的模式。其中, Sensor 所有模式下的序列通过配置文件 sensor\_interface\_cfg\_params.c 中配置, 具体配置规则见 2.4 “输入时序配置”。

#### 【语法】

```
HI_S32 HI_MAPI_Sensor_SetAttr(HI_HANDLE SensorHdl, HI_MPP_SENSOR_ATTR_S*  
pstSensorMode);
```

#### 【参数】



参数名称	描述	输入/输出
SensorHdl	Sensor 的 Handle 号。 取值范围：[0, HI_SENSOR_MAX_NUM]。	输入
pstSensorMode	Sensor 模式。 静态属性。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

- Sensor 支持的所有模式需要产品开发者在 sensor\_interface\_cfg\_params.c 文件中进行配置，具体配置规则见 2.4 节。
- 可通过 HI\_MAPI\_Sensor\_GetAllModes 接口获取指定 Sensor 所支持的所有的模式。
- 设置的模式必须是模式列表中的一种设置才会成功，否则返回失败。
- 若当前要设置的模式与前一次要设置的模式保持一致时，直接返回成功；若不一致时，则必须保证 VCap 处于停止状态才能设置成功，否则返回失败。

#### 【举例】

见 [HI\\_MAPI\\_VCap\\_Start](#) 接口。

#### 【相关主题】

- [HI\\_MAPI\\_Sensor\\_GetAttr](#)
- [HI\\_MAPI\\_VCap\\_Start](#)
- [HI\\_MAPI\\_VCap\\_Isp\\_Start](#)

## HI\_MAPI\_Sensor\_GetAttr

#### 【描述】





获取 Sensor 的当前模式。

#### 【语法】

```
HI_S32 HI_MAPI_Sensor_GetAttr(HI_HANDLE SensorHdl, HI_MPP_SENSOR_ATTR_S*  
pstSensorMode);
```

#### 【参数】

参数名称	描述	输入/输出
SensorHdl	Sensor 的 Handle 号。 取值范围：[0, HI_SENSOR_MAX_NUM]。	输入
pstSensorMode	Sensor 模式属性指针。	输出

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

如果未设置 Sensor 模式属性，该接口将返回失败。

#### 【举例】

无

#### 【相关主题】

[HI\\_MAPI\\_Sensor\\_SetAttr](#)

## HI\_MAPI\_VCap\_SetAttr

#### 【描述】

设置 VCAP 相关属性。

#### 【语法】



```
HI_S32 HI_MAPI_VCap_SetAttr(HI_HANDLE VCapHdl, HI_MPP_VCAP_ATTR_S  
*pstVCapAttr);
```

#### 【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
pstVCapAttr	VCAP 属性。 静态属性。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

- 必须先设置 Sensor 模式才能设置 VCAP 属性。
- 若当前要设置的属性与前一次要设置的保持一致时，直接返回成功；若不一致时，则必须保证 VCap 处于停止状态才能设置成功，否则返回失败。

#### 【举例】

见 [HI\\_MAPI\\_VCap\\_Start](#) 接口。

#### 【相关主题】

[HI\\_MAPI\\_Sensor\\_SetAttr](#)

## HI\_MAPI\_VCap\_GetAttr

#### 【描述】

获取 VCAP 相关属性。

#### 【语法】



```
HI_S32 HI_MAPI_VCap_GetAttr(HI_HANDLE VCapHdl, HI_MPP_VCAP_ATTR_S  
*pstVCapAttr);
```

#### 【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
pstVCapAttr	VCAP 属性指针。	输出

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

如果未设置 VCAP 属性，该接口将返回失败。

#### 【举例】

无

#### 【相关主题】

[HI\\_MAPI\\_VCap\\_SetAttr](#)

## HI\_MAPI\_VCap\_Isp\_Start

#### 【描述】

启用 VCAP 的内部图像处理单元，启动成功后会对原始数据进行图像效果处理。

#### 【语法】

```
HI_S32 HI_MAPI_VCap_Isp_Start(HI_HANDLE VCapHdl);
```

#### 【参数】



参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

- 必须先设置 Sensor 模式和 VCAP 属性之后才能调用该接口，否则返回失败。
- 可支持重复启动，重复启动直接返回成功。
- 在 ISP 第一次启动之后，在后续的模式切换或分辨率切换的过程中，ISP 可不停止，但是 VCAP 必须停止后才能切换。

#### 【举例】

见 [HI\\_MAPI\\_VCap\\_Start](#) 接口。

#### 【相关主题】

- [HI\\_MAPI\\_Sensor\\_SetAttr](#)
- [HI\\_MAPI\\_VCap\\_SetAttr](#)
- [HI\\_MAPI\\_VCap\\_Isp\\_Stop](#)

## HI\_MAPI\_VCap\_Isp\_Stop

#### 【描述】

禁用 VCAP 的内部图像处理单元。

#### 【语法】

```
HI_S32 HI_MAPI_VCap_Isp_Stop(HI_HANDLE VCapHdl);
```

#### 【参数】



参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

可支持重复停止，重复停止直接返回成功。

#### 【举例】

无

#### 【相关主题】

[HI\\_MAPI\\_VCap\\_Isp\\_Start](#)

## HI\_MAPI\_VCap\_Isp\_GetIspDev

#### 【描述】

获取 VCap 的内部 ISP 设备。

#### 【语法】

```
HI_S32 HI_MAPI_VCap_Isp_GetIspDev(HI_HANDLE VCapHdl, HI_S32* pIspDevNo);
```

#### 【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
pIspDevNo	ISP 设备号。	输出



【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无

【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

【注意】

该接口需在 VCAP 和 ISP 启动之后才能获取。

【举例】

无

【相关主题】

无

## HI\_MAPI\_VCap\_Start

【描述】

启用 VCAP。

【语法】

```
HI_S32 HI_MAPI_VCap_Start(HI_HANDLE VCapHdl);
```

【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入

【返回值】

返回值	描述
0	成功。



返回值	描述
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

- 启用前必须已经设置 Sensor 模式和 VCAP 属性，否则返回失败。
- 可重复启用，返回成功。

#### 【举例】

视频场景：

```
HI_S32 s32Ret = HI_SUCCESS;
HI_S32 SensorHdl = 0;
HI_S32 VcapHdl1 = 0;
HI_MPP_SENSOR_ATTR_S stSensorAttr;
HI_MPP_VCAP_ATTR_S stVcapAttr1;

stSensorAttr.s32FrameRate = 30;
stSensorAttr.stResolution.u32Width = 1280;
stSensorAttr.stResolution.u32Height = 720;
stSensorAttr.enWdrMode = HI_MPP_WDR_MODE_NONE;

stVcapAttr1.enWdrMode = HI_MPP_WDR_MODE_NONE;
stVcapAttr1.s32FrameRate = 30;
stVcapAttr1.stResolution.u32Width = 1280;
stVcapAttr1.stResolution.u32Height = 720;
stVcapAttr1.enPixelFormat = HI_MPP_PIXEL_FORMAT_420;
stVcapAttr1.enVCapType = HI_MPP_VCAP_TYPE_VIDEO;

//Media init
s32Ret = HI_MAPI_Media_Init();
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_Media_Init fail\n");
    return HI_FAILURE;
}
```



```
s32Ret = HI_MAPI_Sensor_SetAttr(SensorHdl, &stSensorAttr);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_Sensor_SetAttr fail\n");
    return HI_FAILURE;
}

//start cap2
s32Ret = HI_MAPI_VCap_SetAttr(VcapHdl1, &stVcapAttr1);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_VCap_SetAttr capture:%d fail\n", VcapHdl1);
    return HI_FAILURE;
}

s32Ret = HI_MAPI_VCap_Isp_Start(VcapHdl1);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_VCap_Isp_Start fail\n");
    return HI_FAILURE;
}

s32Ret = HI_MAPI_VCap_Start(VcapHdl1);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_VCap_Start fail\n");
    return HI_FAILURE;
}
```

#### 【相关主题】

- [HI\\_MAPI\\_Sensor\\_SetAttr](#)
- [HI\\_MAPI\\_VCap\\_SetAttr](#)
- [HI\\_MAPI\\_VCap\\_Stop](#)

## HI\_MAPI\_VCap\_Stop

#### 【描述】

停止 VCap 工作。

#### 【语法】

```
HI_S32 HI_MAPI_VCap_Stop(HI_HANDLE VCapHdl);
```

#### 【参数】





参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

- 可重复停止，不返回失败。
- NDK 支持低功耗处理，禁用 VCAP 将完全关闭硬件单元，需要重新设置 Sensor 模式和 VCAP 属性，才能启用视频采集单元。

#### 【举例】

无

#### 【相关主题】

- [HI\\_MAPI\\_VCap\\_SetAttr](#)
- [HI\\_MAPI\\_VCap\\_Start](#)

## HI\_MAPI\_VCap\_Trigger

#### 【描述】

触发抓拍。

#### 【语法】

```
HI_S32 HI_MAPI_VCap_Trigger(HI_HANDLE VCapHdl);
```

#### 【参数】



参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为错误码。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

- 必须在调用 [HI\\_MAPI\\_VCap\\_SetSnapAttr](#) 之后触发才能成功，否则返回失败。
- 必须在触发之后，VCAP0 才有数据。并按照指定帧率产生数据。
- 该接口支持多次触发。
- 在一次触发多帧后，如果需要中途停止，则调用 [HI\\_MAPI\\_VCap\\_StopTrigger](#) 停止。

#### 【举例】

无

#### 【相关主题】

- [HI\\_MAPI\\_VCap\\_SetSnapAttr](#)
- [HI\\_MAPI\\_VCap\\_StopTrigger](#)

## HI\_MAPI\_VCap\_StopTrigger

#### 【描述】

在连续抓拍多张图片的情况下终止抓拍。

#### 【语法】

```
HI_S32 HI_MAPI_VCap_StopTrigger(HI_HANDLE VCapHdl);
```

#### 【参数】



参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无

【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

【注意】

- 在连续拍多张的情况下，如果需要中途终止抓拍，则调用该接口。
- 重复停止直接返回成功。

【举例】

无

【相关主题】

[HI\\_MAPI\\_VCap\\_Trigger](#)

## HI\_MAPI\_VCap\_OSD\_SetAttr

【描述】

设置 OSD 的属性。

【语法】

```
HI_S32 HI_MAPI_VCap_OSD_SetAttr(HI_HANDLE VCapHdl, HI_HANDLE VOsdHdl,  
HI\_MPP\_OSD\_ATTR\_S\* pstOsdAttr);
```

【参数】



参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
VOsdHdl	OSD 的 Handle 号。 取值范围：[0, HI_VCAP_OSD_MAX_NUM)。	输入
pstOsdAttr	OSD 的属性。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

- VCAP 中可叠加 BMP 图片，其中支持的图片格式为 HI\_MPP\_PIXEL\_FORMAT\_RGB\_1555、HI\_MPP\_PIXEL\_FORMAT\_RGB\_8888。
- 在启动 OSD 之前，必须先调用该接口设置叠加的 OSD 的相关属性。
- pstOsdAttr.stBitmapAttr.pData 必须要传入 BMP 图片的物理地址。
  - 若在 Linux 端使用该接口，存放图片的内存可通过 HI\_MAPI\_AllocBuffer 分配，在使用完之后，需要调用 HI\_MAPI\_FreeBuffer 对该内存进行释放，否则会造成内存泄露，对应的借口请参考“系统控制”章节。
  - 若在 Huawei LiteOS 端使用该接口，存放图片的内存既可通过 HI\_MAPI\_AllocBuffer 分配，也可直接通过 malloc 分配。
- 该接口支持动态修改，如时间 OSD 需要定时刷新，可将需要显示的时间通过该接口定时设置实现。

#### 【举例】

无

#### 【相关主题】

[HI\\_MAPI\\_VCap\\_OSD\\_Start](#)



## HI\_MAPI\_VCap\_OSD\_GetAttr

### 【描述】

获取 OSD 的属性。

### 【语法】

```
HI_S32 HI_MAPI_VCap_OSD_GetAttr(HI_HANDLE VCapHdl, HI_HANDLE VOsdHdl,  
HI_MPP_OSD_ATTR_S *pstOsdAttr);
```

### 【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
VOsdHdl	OSD 的 Handle 号。 取值范围：[0, HI_VCAP_OSD_MAX_NUM)。	输入
pstOsdAttr	OSD 的属性指针。	输出

### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为错误码。

### 【芯片差异】

无

### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

### 【注意】

必须在设置 OSD 属性之后才能获取，否则会返回失败。

### 【举例】

无

### 【相关主题】

[HI\\_MAPI\\_VCap\\_OSD\\_SetAttr](#)



## HI\_MAPI\_VCap\_OSD\_Start

### 【描述】

启动 OSD，将 BMP 图片叠加到 YUV 数据流中。

### 【语法】

```
HI_S32 HI_MAPI_VCap_OSD_Start(HI_HANDLE VCapHdl, HI_HANDLE OsdHdl);
```

### 【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
VOsdHdl	OSD 的 Handle 号。 取值范围：[0, HI_VCAP_OSD_MAX_NUM)。	输入

### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

### 【芯片差异】

无

### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

### 【注意】

- 必须在设置 OSD 属性之后才能启动 OSD，否则会返回失败。
- 该接口支持重复启动，重复启动时直接返回成功。

### 【举例】

无

### 【相关主题】

[HI\\_MAPI\\_VCap\\_OSD\\_Stop](#)



## HI\_MAPI\_VCap\_OSD\_Stop

### 【描述】

停止 OSD，YUV 数据流中将没有 OSD。

### 【语法】

```
HI_S32 HI_MAPI_VCap_OSD_Stop(HI_HANDLE VCapHdl, HI_HANDLE OsdHdl);
```

### 【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
VOsdHdl	OSD 的 Handle 号。 取值范围：[0, HI_VCAP_OSD_MAX_NUM)。	输入

### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

### 【芯片差异】

无

### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

### 【注意】

该接口支持重复停止，直接返回成功。

### 【举例】

无

### 【相关主题】

[HI\\_MAPI\\_VCap\\_OSD\\_Start](#)

## HI\_MAPI\_VCap\_SetExifInfo

### 【描述】



设置 Exif 信息，包括图像的描述、生产厂商、设备型号、软件版本等信息。

#### 【语法】

```
HI_S32 HI_MAPI_VCap_SetExifInfo(HI_HANDLE VCapHdl,  
HI_MPP_SNAP_EXIF_INFO_S* pstExifInfo);
```

#### 【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
pstExifInfo	Exif 信息。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

无

#### 【举例】

无

#### 【相关主题】

[HI\\_MAPI\\_VCap\\_GetExifInfo](#)

## HI\_MAPI\_VCap\_GetExifInfo

#### 【描述】

获取 Exif 信息。

#### 【语法】





```
HI_S32 HI_MAPI_VCap_GetExifInfo(HI_HANDLE  
VCapHdl, HI_MPP_SNAP_EXIF_INFO_S* pstExifInfo);
```

#### 【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
pstExifInfo	Exif 信息指针。	输出

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

无

#### 【举例】

无

#### 【相关主题】

[HI\\_MAPI\\_VCap\\_SetExifInfo](#)

## HI\_MAPI\_VCap\_SetAttrEx

#### 【描述】

设置 VCAP 的高级属性。

#### 【语法】

```
HI_S32 HI_MAPI_VCap_SetAttrEx(HI_HANDLE VCapHdl, HI_MPP_VCAP_CMD_E enCMD,  
HI_VOID* pAttr, HI_U32 u32Len);
```

#### 【参数】



参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
enCMD	要设置的高级属性对应的命令。	输入
pAttr	高级属性的结构体指针。	输入
u32Len	高级属性结构的长度。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

无

表2-1 设置 VCAP 高级属性 enCMD 对应的结构

enCMD	高级属性结构
HI_MPP_VCAP_CMD_SetExposure	HI_MPP_VCAP_EXPOSURE_ATTR_S
HI_MPP_VCAP_CMD_SetWBAttr	HI_MPP_VCAP_WB_ATTR_S
HI_MPP_VCAP_CMD_SetSharpen	HI_MPP_VCAP_SHARPEN_S
HI_MPP_VCAP_CMD_SetBrightness	HI_MPP_VCAP_BRIGHTNESS_S
HI_MPP_VCAP_CMD_SetSaturation	HI_MPP_VCAP_SATURATION_S
HI_MPP_VCAP_CMD_SetLDC	HI_MPP_VCAP_LDC_ATTR_S

#### 【举例】

```
HI_MPP_VCAP_SATURATION_S stSaturation;
```



```
stSaturation.s32Saturation = 98;
s32Ret =
HI_MAPI_VCap_SetAttrEx(VcapHdl1, HI_MPP_VCAP_CMD_SetSaturation, &stSaturation,
sizeof(HI_MPP_VCAP_SATURATION_S));
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_VCap_SetAttrEx fail s32Ret:%x\n", s32Ret);
    return HI_FAILURE;
}
```

#### 【相关主题】

无

## HI\_MAPI\_VCap\_GetAttrEx

#### 【描述】

获取 VCAP 的高级属性，包括曝光属性、白平衡、亮度、饱和度、锐度、LDC 等属性。

#### 【语法】

```
HI_S32 HI_MAPI_VCap_GetAttrEx(HI_HANDLE VCapHdl, HI_MPP_VCAP_CMD_E enCMD,
HI_VOID* pAttr, HI_U32 u32Len);
```

#### 【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
enCMD	要获取的高级属性对应的命令。	输入
pAttr	高级属性的结构体指针。	输出
u32Len	高级属性结构的长度。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无



【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

【注意】

无

表2-2 获取 VCAP 高级属性 enCMD 对应的结构

enCMD	高级属性结构
HI_MPP_VCAP_CMD_GetExposure	HI_MPP_VCAP_EXPOSURE_ATTR_S
HI_MPP_VCAP_CMD_GetWBAttr	HI_MPP_VCAP_WB_ATTR_S
HI_MPP_VCAP_CMD_GetSharpen	HI_MPP_VCAP_SHARPEN_S
HI_MPP_VCAP_CMD_GetBrightness	HI_MPP_VCAP_BRIGHTNESS_S
HI_MPP_VCAP_CMD_GetSaturation	HI_MPP_VCAP_SATURATION_S
HI_MPP_VCAP_CMD_GetLDC	HI_MPP_VCAP_LDC_ATTR_S

【举例】

无

【相关主题】

无

## HI\_MAPI\_VCap\_EnableDumpRaw

【描述】

启用 Dump Raw 数据功能。

【语法】

```
HI_S32 HI_MAPI_VCap_EnableDumpRaw(HI_HANDLE VCapHdl,  
HI_MPP_VCAP_DUMP_ATTR_S* pstDumpAttr);
```

【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
pstDumpAttr	Dump 属性。	输入



【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无

【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

【注意】

无

【举例】

无

【相关主题】

- [HI\\_MAPI\\_VCap\\_DisableDumpRaw](#)
- [HI\\_MAPI\\_VCap\\_DumpRaw](#)

## HI\_MAPI\_VCap\_DisableDumpRaw

【描述】

禁用 Dump Raw 数据功能。

【语法】

```
HI_S32 HI_MAPI_VCap_DisableDumpRaw(HI_HANDLE VCapHdl);
```

【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入

【返回值】

返回值	描述
正数值	有效返回值。



返回值	描述
非正数值	无效返回值。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

无

#### 【举例】

无

#### 【相关主题】

[HI\\_MAPI\\_VCap\\_EnableDumpRaw](#)

## HI\_MAPI\_VCap\_DumpRaw

#### 【描述】

Dump Raw 数据。

#### 【语法】

```
HI_S32 HI_MAPI_VCap_DumpRaw(HI_HANDLE VCapHdl, HI_U32 u32Count,  
PFUN_VCAP_RawDataProc pfunVCapRawProc);
```

#### 【参数】

参数名称	描述	输入/输出
VCapHdl	VCAP 的 Handle 号。 取值范围：[0, HI_VCAP_MAX_NUM)。	输入
u32Count	需要 dump 的 Raw 数据的张数。	输入
pfunVCapRawProc	Raw 数据返回的回调函数指针。	输入

#### 【返回值】

返回值	描述
0	成功。



返回值	描述
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无

#### 【需求】

- 头文件：hi\_mapi\_comm\_define.h、hi\_mapi\_vcap\_define.h、hi\_mapi\_vcap.h
- 库文件：libhi3518e\_mapi\_vcap.a

#### 【注意】

- 设置不同的 Bit 为进行存储时，则需要按照不同的存储方式进行解析。
- 对不同的 Bit 位的 Raw 数据进行解析的算法见【举例】。
- 当在抓拍的 VCAP 上进行 Dump 数据，必须要调 [HI\\_MAPI\\_VCap\\_Trigger](#) 之后才能 Dump 到数据，且 Dump 数据的的位数不由 [HI\\_MAPI\\_VCap\\_EnableDumpRaw](#) 接口中设置生效，而默认为 VCAP 的 Y 和 C 的分量掩码决定，该分量掩码在 sensor\_interface\_cfg\_params.c 中配置，默认为 au32CompMask = {0xffff0000,0}。因此，对于抓拍 VCAP 的 Raw 数据，应该默认按照 HI\_MPP\_PIXEL\_FORMAT\_RGB\_BAYER\_12BPP 进行解析。

#### 【举例】

```
HI_S32 PDT_Snap_ConvertBitPixel(HI_U8 *pu8Data, HI_U32 u32DataNum,
HI_MPP_PIXEL_FORMAT_E enPixelFormat, HI_U16 *pul6OutData)
{
    HI_S32 i, u32Tmp, s32OutCnt;
    HI_U32 u32Val;
    HI_U64 u64Val;
    HI_U8 *pu8Tmp = pu8Data;

    s32OutCnt = 0;
    switch(enPixelFormat)
    {
        case HI_MPP_PIXEL_FORMAT_RGB_BAYER_10BPP:
        {
            /* 4 pixels consist of 5 bytes */
            u32Tmp = u32DataNum / 4;

            for (i = 0; i < u32Tmp; i++)
            {
                /* byte4 byte3 byte2 byte1 byte0 */
                pu8Tmp = pu8Data + 5 * i;
                u64Val = pu8Tmp[0] + ((HI_U32)pu8Tmp[1] << 8) +
```



```
((HI_U32)pu8Tmp[2] << 16) +
    ((HI_U32)pu8Tmp[3] << 24) + ((HI_U64)pu8Tmp[4] << 32);

    pul6OutData[s32OutCnt++] = u64Val & 0x3ff;
    pul6OutData[s32OutCnt++] = (u64Val >> 10) & 0x3ff;
    pul6OutData[s32OutCnt++] = (u64Val >> 20) & 0x3ff;
    pul6OutData[s32OutCnt++] = (u64Val >> 30) & 0x3ff;
}
}
break;
case HI_MPP_PIXEL_FORMAT_RGB_BAYER_12BPP:
{
    /* 2 pixels consist of 3 bytes */
    u32Tmp = u32DataNum / 2;

    for (i = 0; i < u32Tmp; i++)
    {
        /* byte2 byte1 byte0 */
        pu8Tmp = pu8Data + 3 * i;
        u32Val = pu8Tmp[0] + (pu8Tmp[1] << 8) + (pu8Tmp[2] << 16);
        pul6OutData[s32OutCnt++] = u32Val & 0xffff;
        pul6OutData[s32OutCnt++] = (u32Val >> 12) & 0xffff;
    }
}
break;
case HI_MPP_PIXEL_FORMAT_RGB_BAYER_14BPP:
{
    /* 4 pixels consist of 7 bytes */
    u32Tmp = u32DataNum / 4;

    for (i = 0; i < u32Tmp; i++)
    {
        pu8Tmp = pu8Data + 7 * i;
        u64Val = pu8Tmp[0] + ((HI_U32)pu8Tmp[1] << 8) +
            ((HI_U32)pu8Tmp[2] << 16) +
            ((HI_U32)pu8Tmp[3] << 24) + ((HI_U64)pu8Tmp[4] << 32) +
            ((HI_U64)pu8Tmp[5] << 40) + ((HI_U64)pu8Tmp[6] << 48);

        pul6OutData[s32OutCnt++] = u64Val & 0x3fff;
        pul6OutData[s32OutCnt++] = (u64Val >> 14) & 0x3fff;
        pul6OutData[s32OutCnt++] = (u64Val >> 28) & 0x3fff;
        pul6OutData[s32OutCnt++] = (u64Val >> 42) & 0x3fff;
    }
}
}
```





```
        break;
    default:
        fprintf(stderr, "unsuport enPixelFormat: %d\n", enPixelFormat);
        return -1;
        break;
}

return s32OutCnt;
}
```

#### 【相关主题】

[HI\\_MAPI\\_VCap\\_EnableDumpRaw](#)

## 2.6 数据类型

视频输入相关数据类型定义如下：

- [HI\\_VCAP\\_MAX\\_NUM](#)：定义支持的 VCAP 的最大个数。
- [HI\\_MPP\\_VCAP\\_MAX\\_BRIGHTNESS](#)：定义亮度属性的最大值。
- [HI\\_MPP\\_VCAP\\_MIN\\_BRIGHTNESS](#)：定义亮度属性的最小值。
- [HI\\_MPP\\_VCAP\\_MAX\\_SATURATION](#)：定义饱和度属性的最大值。
- [HI\\_MPP\\_VCAP\\_MIN\\_SATURATION](#)：定义饱和度属性的最小值。
- [HI\\_MPP\\_VCAP\\_MAX\\_SHARPEN\\_ST](#)：定义 Sharpen 强度的最大值。
- [HI\\_MPP\\_VCAP\\_MIN\\_SHARPEN\\_ST](#)：定义 Sharpen 强度的最小值。
- [HI\\_MPP\\_VCAP\\_MAX\\_COLORTEMP](#)：定义色温的最大值。
- [HI\\_MPP\\_VCAP\\_MIN\\_COLORTEMP](#)：定义色温的最小值。
- [HI\\_MPP\\_VCAP\\_MAX\\_LDC\\_CENTERX\\_OFFSET](#)：定义畸变中心点相对图象中心点水平偏移的最大值。
- [HI\\_MPP\\_VCAP\\_MIN\\_LDC\\_CENTERX\\_OFFSET](#)：定义畸变中心点相对图象中心点水平偏移的最小值。
- [HI\\_MPP\\_VCAP\\_MAX\\_LDC\\_CENTERY\\_OFFSET](#)：定义畸变中心点相对图象中心点垂直偏移的最大值。
- [HI\\_MPP\\_VCAP\\_MIN\\_LDC\\_CENTERY\\_OFFSET](#)：定义畸变中心点相对图象中心点垂直偏移的最小值。
- [HI\\_MPP\\_VCAP\\_MAX\\_LDC\\_RATIO](#)：定义畸变程度的最大值。
- [HI\\_MPP\\_VCAP\\_MIN\\_LDC\\_RATIO](#)：定义畸变程度的最小值。
- [HI\\_MPP\\_VCAP\\_MAX\\_LDC\\_MINRATIO](#)：定义最小畸变程度的最大值。
- [HI\\_MPP\\_VCAP\\_MIN\\_LDC\\_MINRATIO](#)：定义最小畸变程度的最小值。
- [HI\\_MPP\\_VCAP\\_MAX\\_RAWDUMP\\_DEPTH](#)：定义 Raw 数据 Dump 深度的最大值。



- [HI\\_MPP\\_VCAP\\_MIN\\_RAWDUMP\\_DEPTH](#): 定义 Raw 数据 Dump 深度的最小值。
- [HI\\_SNAP\\_EXIF\\_DRSCRIPTION\\_LENGTH](#): 定义 Exif 描述信息的最大长度。
- [HI\\_MPP\\_WDR\\_MODE\\_E](#): 定义 WDR 模式。
- [HI\\_MPP\\_SENSOR\\_ATTR\\_S](#): 定义 Sensor 模式。
- [HI\\_MPP\\_VCAP\\_ATTR\\_S](#): 定义 VCAP 属性。
- [HI\\_MPP\\_VCAP\\_TYPE\\_E](#): 定义 VCAP 类型。
- [HI\\_MPP\\_VCAP\\_DUMP\\_ATTR\\_S](#): 定义 Dump 属性。
- [PFN\\_VCAP\\_RawDataProc](#): DumpRaw 的回调函数指针。
- [HI\\_FRAME\\_DATA\\_S](#): 帧数据结构。
- [HI\\_FRAME\\_DATA\\_TYPE\\_E](#): 帧数据类型
- [HI\\_MPP\\_RESOLUTION\\_S](#): 定义分辨率结构体。
- [HI\\_MPP\\_VCAP\\_OP\\_TYPE\\_E](#): 定义操作模式。
- [HI\\_MPP\\_VCAP\\_CMD\\_E](#): 定义 VCAP 高级属性的 CMD 值。
- [HI\\_MPP\\_VCAP\\_EXPOSURE\\_MANUAL\\_MODE\\_S](#): 定义手动曝光属性。
- [HI\\_MPP\\_VCAP\\_AE\\_RANGE\\_S](#): 定义曝光增益的最大值和最小值。
- [HI\\_MPP\\_VCAP\\_EXPOSURE\\_AUTO\\_MODE\\_S](#): 定义自动曝光属性。
- [HI\\_MPP\\_VCAP\\_EXPOSURE\\_ATTR\\_S](#): 定义曝光属性。
- [HI\\_MPP\\_VCAP\\_WB\\_MANUAL\\_MODE\\_S](#): 定义手动 WB 模式。
- [HI\\_MPP\\_VCAP\\_WB\\_ATTR\\_S](#): 定义 WB 属性。
- [HI\\_MPP\\_VCAP\\_SHARPEN\\_MANUAL\\_ATTR\\_S](#): 定义手动 Sharpen 属性。
- [HI\\_MPP\\_VCAP\\_SHARPEN\\_S](#): 定义 Sharpen 属性。
- [HI\\_MPP\\_VCAP\\_BRIGHTNESS\\_S](#): 定义亮度属性。
- [HI\\_MPP\\_VCAP\\_SATURATION\\_S](#): 定义饱和度属性。
- [HI\\_MPP\\_VCAP\\_LDC\\_TYPE\\_E](#): 定义畸变校正类型。
- [HI\\_MPP\\_VCAP\\_LDC\\_ATTR\\_S](#): 定义畸变校正属性。
- [HI\\_MPP\\_SNAP\\_EXIF\\_INFO\\_S](#): 定义 Exif 信息。
- [HI\\_MPP\\_OSD\\_DISPATTR\\_S](#): 定义 OSD 的显示属性。
- [HI\\_MPP\\_OSD\\_BITMAP\\_ATTR\\_S](#): 定义 OSD 的 BMP 图片属性。
- [HI\\_MPP\\_OSD\\_ATTR\\_S](#): 定义 OSD 的属性。
- [HI\\_MPP\\_PIXEL\\_FORMAT\\_E](#): 定义像素格式。

## HI\_VCAP\_MAX\_NUM

### 【说明】

定义支持的 VCAP 的最大个数。

### 【定义】

```
#define HI_VCAP_MAX_NUM 1
```



**【注意事项】**

无

**【相关数据类型及接口】**

无

## HI\_MPP\_VCAP\_MAX\_BRIGHTNESS

**【说明】**

定义亮度属性的最大值。

**【定义】**

```
#define HI_MPP_VCAP_MAX_BRIGHTNESS 100
```

**【注意事项】**

无

**【相关数据类型及接口】**

无

## HI\_MPP\_VCAP\_MIN\_BRIGHTNESS

**【说明】**

定义亮度属性的最小值。

**【定义】**

```
#define HI_MPP_VCAP_MIN_BRIGHTNESS 0
```

**【注意事项】**

无

**【相关数据类型及接口】**

无

## HI\_MPP\_VCAP\_MAX\_SATURATION

**【说明】**

定义饱和度的最大值。

**【定义】**

```
#define HI_MPP_VCAP_MAX_SATURATION 100
```

**【注意事项】**

无

**【相关数据类型及接口】**



无

## HI\_MPP\_VCAP\_MIN\_SATURATION

### 【说明】

定义饱和度的最小值。

### 【定义】

```
#define HI_MPP_VCAP_MIN_SATURATION 0
```

### 【注意事项】

无

### 【相关数据类型及接口】

无

## HI\_MPP\_VCAP\_MAX\_SHARPEN\_ST

### 【说明】

定义 Sharpen 强度的最大值。

### 【定义】

```
#define HI_MPP_VCAP_MAX_SHARPEN_ST 255
```

### 【注意事项】

无

### 【相关数据类型及接口】

无

## HI\_MPP\_VCAP\_MIN\_SHARPEN\_ST

### 【说明】

定义 Sharpen 强度的最小值。

### 【定义】

```
#define HI_MPP_VCAP_MIN_SHARPEN_ST 0
```

### 【注意事项】

无

### 【相关数据类型及接口】

无



## HI\_MPP\_VCAP\_MAX\_COLORTEMP

### 【说明】

定义色温的最大值。

### 【定义】

```
#define HI_MPP_VCAP_MAX_COLORTEMP 15000
```

### 【注意事项】

无

### 【相关数据类型及接口】

无

## HI\_MPP\_VCAP\_MIN\_COLORTEMP

### 【说明】

定义色温的最小值。

### 【定义】

```
#define HI_MPP_VCAP_MIN_COLORTEMP 1500
```

### 【注意事项】

无

### 【相关数据类型及接口】

无

## HI\_MPP\_VCAP\_MAX\_LDC\_CENTERX\_OFFSET

### 【说明】

定义畸变中心点相对图象中心点水平偏移的最大值。

### 【定义】

```
#define HI_MPP_VCAP_MAX_LDC_CENTERX_OFFSET 127
```

### 【注意事项】

无

### 【相关数据类型及接口】

无

## HI\_MPP\_VCAP\_MIN\_LDC\_CENTERX\_OFFSET

### 【说明】



定义畸变中心点相对图象中心点水平偏移的最小值。

**【定义】**

```
#define HI_MPP_VCAP_MIN_LDC_CENTERX_OFFSET -127
```

**【注意事项】**

无

**【相关数据类型及接口】**

无

## HI\_MPP\_VCAP\_MAX\_LDC\_CENTERY\_OFFSET

**【说明】**

定义畸变中心点相对图象中心点垂直偏移的最大值。

**【定义】**

```
#define HI_MPP_VCAP_MIN_LDC_CENTERY_OFFSET 127
```

**【芯片差异】**

无

**【注意事项】**

无

**【相关数据类型及接口】**

无

## HI\_MPP\_VCAP\_MIN\_LDC\_CENTERY\_OFFSET

**【说明】**

定义畸变中心点相对图象中心点垂直偏移的最小值。

**【定义】**

```
#define HI_MPP_VCAP_MIN_LDC_CENTERY_OFFSET -127
```

**【芯片差异】**

无

**【注意事项】**

无

**【相关数据类型及接口】**

无



## HI\_MPP\_VCAP\_MAX\_LDC\_RATIO

### 【说明】

定义畸变程度的最大值。

### 【定义】

```
#define HI_MPP_VCAP_MAX_LDC_RATIO 300
```

### 【注意事项】

无

### 【相关数据类型及接口】

无

## HI\_MPP\_VCAP\_MIN\_LDC\_RATIO

### 【说明】

定义畸变程度的最小值。

### 【定义】

```
#define HI_MPP_VCAP_MIN_LDC_RATIO -300
```

### 【注意事项】

无

### 【相关数据类型及接口】

无

## HI\_MPP\_VCAP\_MAX\_LDC\_MINRATIO

### 【说明】

定义最小畸变程度的最大值。

### 【定义】

```
#define HI_MPP_VCAP_MAX_LDC_MINRATIO 0
```

### 【注意事项】

无

### 【相关数据类型及接口】

无

## HI\_MPP\_VCAP\_MIN\_LDC\_MINRATIO

### 【说明】



定义最小畸变程度的最小值。

**【定义】**

```
#define HI_MPP_VCAP_MIN_LDC_MINRATIO -300
```

**【注意事项】**

无

**【相关数据类型及接口】**

无

## HI\_MPP\_VCAP\_MAX\_RAWDUMP\_DEPTH

**【说明】**

定义 DumpRaw 的深度的最大值。

**【定义】**

```
#define HI_MPP_VCAP_MAX_RAWDUMP_DEPTH 8
```

**【注意事项】**

无

**【相关数据类型及接口】**

无

## HI\_MPP\_VCAP\_MIN\_RAWDUMP\_DEPTH

**【说明】**

定义 DumpRaw 的深度的最小值。

**【定义】**

```
#define HI_MPP_VCAP_MIN_RAWDUMP_DEPTH 0
```

**【注意事项】**

无

**【相关数据类型及接口】**

无

## HI\_SNAP\_EXIF\_DRSCRIPTION\_LENGTH

**【说明】**

定义 Exif 描述信息的最大长度。

**【定义】**

```
#define HI_SNAP_EXIF_DRSCRIPTION_LENGTH 32
```





【注意事项】

无

【相关数据类型及接口】

[HI\\_MPP\\_SNAP\\_EXIF\\_INFO\\_S](#)

## HI\_MPP\_WDR\_MODE\_E

【说明】

定义 WDR 模式。

【定义】

```
typedef enum hiHI_MPP_WDR_MODE_E
{
    HI_MPP_WDR_MODE_NONE = 0,
    HI_MPP_WDR_MODE_BUILT_IN,
    HI_MPP_WDR_MODE_2To1_LINE,
    HI_MPP_WDR_MODE_2To1_FRAME,
    HI_MPP_WDR_MODE_2To1_FRAME_FULL_RATE,
    HI_MPP_WDR_MODE_3To1_LINE,
    HI_MPP_WDR_MODE_3To1_FRAME,
    HI_MPP_WDR_MODE_3To1_FRAME_FULL_RATE,
    HI_MPP_WDR_MODE_4To1_LINE,
    HI_MPP_WDR_MODE_4To1_FRAME,
    HI_MPP_WDR_MODE_4To1_FRAME_FULL_RATE,
    HI_MPP_WDR_MODE_BUTT
}HI_MPP_WDR_MODE_E;
```

【成员】

成员名称	描述
HI_MPP_WDR_MODE_NONE	线性模式。则不进行 WDR 合成
HI_MPP_WDR_MODE_BUILT_IN	Sensor 合成 WDR 模式。
HI_MPP_WDR_MODE_2To1_LINE	2 帧合成行 WDR 模式。
HI_MPP_WDR_MODE_2To1_FRAME	2 帧合成帧 WDR 模式。
HI_MPP_WDR_MODE_2To1_FRAME_FULL_RATE	2 帧合成帧 WDR 全帧率模式。
HI_MPP_WDR_MODE_3To1_LINE	3 帧合成行 WDR 模式。
HI_MPP_WDR_MODE_3To1_FRAME	3 帧合成帧 WDR 模式。



成员名称	描述
HI_MPP_WDR_MODE_3To1_FRAME_FULL_RATE	3 帧合成帧 WDR 全帧率模式。
HI_MPP_WDR_MODE_4To1_LINE	4 帧合成行 WDR 模式。
HI_MPP_WDR_MODE_4To1_FRAME	4 帧合成帧 WDR 模式。
HI_MPP_WDR_MODE_4To1_FRAME_FULL_RATE	4 帧合成帧 WDR 全帧率模式。

#### 【注意事项】

无

#### 【相关数据类型及接口】

- [HI\\_MAPI\\_Sensor\\_SetAttr](#)
- [HI\\_MAPI\\_Sensor\\_GetAttr](#)
- [HI\\_MAPI\\_VCap\\_SetAttr](#)
- [HI\\_MAPI\\_VCap\\_GetAttr](#)

## HI\_MPP\_SENSOR\_ATTR\_S

#### 【说明】

定义 Sensor 的模式属性。

#### 【定义】

```
typedef struct hiHI_MPP_SENSOR_ATTR_S
{
    HI_S32 s32FrameRate;
    HI_MPP_RESOLUTION_S stResolution;
    HI_MPP_WDR_MODE_E enWdrMode;
} HI_MPP_SENSOR_ATTR_S;
```

#### 【成员】

成员名称	描述
s32FrameRate	Sensor 的输入帧率。
stResolution	Sensor 的输入分辨率。
enWdrMode	Sensor 的 WDR 模式。

#### 【注意事项】



对于支持 WDR 模式的 sensor，则在 sensor\_interface\_cfg\_params.c 中配置对应 WDR 模式下的时序。

【相关数据类型及接口】

- [HI\\_MAPI\\_Sensor\\_SetAttr](#)
- [HI\\_MAPI\\_Sensor\\_GetAttr](#)

## HI\_MPP\_VCAP\_ATTR\_S

【说明】

定义 VCAP 的属性。

【定义】

```
typedef struct hiHI_MPP_VCAP_ATTR_S
{
    HI_S32 s32FrameRate;
    HI_MPP_RESOLUTION_S stResolution;
    HI_MPP_WDR_MODE_E enWdrMode;
    HI_MPP_PIXEL_FORMAT_E enPixelFormat;
    HI_MPP_VCAP_TYPE_E enVCapType;
} HI_MPP_VCAP_ATTR_S;
```

【成员】

成员名称	描述
s32FrameRate	VCAP 的帧率。
stResolution	VCAP 的分辨率。
enWdrMode	VCAP 的 WDR 模式。
enPixelFormat	VCAP 输出的像素格式。
enVCapType	Vcap 类型，包括视频和抓拍。

【注意事项】

- 当 sensor 支持 WDR 时，VCAP 可选择期望进行 WDR 的行曝光模式。
- 当 sensor 不支持 WDR 时，VCAP 本身也可进行前后帧参考 WDR，相应的模式需选用帧合成模式。
- enPixelFormat 仅支持 HI\_MPP\_PIXEL\_FORMAT\_422、HI\_MPP\_PIXEL\_FORMAT\_420，且 DIS 和 LDC 功能仅在像素格式为 HI\_MPP\_PIXEL\_FORMAT\_420 的情况下支持。
- Vcap 类型的注意事项请参看 2.2 “重要概念”中的视频采集单元类型部分。

【相关数据类型及接口】



- [HI\\_MAPI\\_VCap\\_SetAttr](#)
- [HI\\_MAPI\\_VCap\\_GetAttr](#)

## HI\_MPP\_VCAP\_TYPE\_E

### 【说明】

定义 VCAP 类型。

### 【定义】

```
typedef enum hiHI_MPP_VCAP_TYPE_E
{
    HI_MPP_VCAP_TYPE_VIDEO = 0,
    HI_MPP_VCAP_TYPE_SNAP,
    HI_MPP_VCAP_TYPE_BUTT
}HI_MPP_VCAP_TYPE_E;
```

### 【成员】

成员名称	描述
HI_MPP_VCAP_TYPE_VIDEO	视频类型。
HI_MPP_VCAP_TYPE_SNAP	抓拍类型。

### 【注意事项】

无

### 【相关数据类型及接口】

[HI\\_MPP\\_VCAP\\_ATTR\\_S](#)

## HI\_MPP\_VCAP\_DUMP\_ATTR\_S

### 【说明】

定义 Raw 数据的 Dump 属性。

### 【定义】

```
typedef struct hiHI_MPP_VCAP_DUMP_ATTR_S
{
    HI\_MPP\_PIXEL\_FORMAT\_E enPixelForamt;           /**<
    [8bit,10bit,12bit,14bit,16bit]*/
    HI_U32 u32Depth;                                /**< [0,8]*/
}HI_MPP_VCAP_DUMP_ATTR_S;
```

### 【成员】



成员名称	描述
enPixelFormat	Raw 数据的格式。
u32Depth	设置获取 Raw 图像的最大深度，即用于存储 Raw 数据的 Buffer 的个数。 默认值 0。 取值范围：[0,8]。

#### 【注意事项】

Raw 数据支持的像素格式为 HI\_MPP\_PIXEL\_FORMAT\_RGB\_BAYER\_8BPP、  
HI\_MPP\_PIXEL\_FORMAT\_RGB\_BAYER\_10BPP、  
HI\_MPP\_PIXEL\_FORMAT\_RGB\_BAYER\_12BPP、  
HI\_MPP\_PIXEL\_FORMAT\_RGB\_BAYER\_14BPP、  
HI\_MPP\_PIXEL\_FORMAT\_RGB\_BAYER

#### 【相关数据类型及接口】

[HI\\_MAPI\\_VCap\\_EnableDumpRaw](#)

## PFN\_VCAP\_RawDataProc

#### 【说明】

DumpRaw 的回调函数指针。

#### 【定义】

```
typedef HI_S32 (*PFN_VCAP_RawDataProc) (HI_HANDLE VCapHdl,  
HI\_FRAME\_DATA\_S\* pVCapRawData);
```

#### 【成员】

成员名称	描述
VCapHdl	VCap 的 Handle。
HI_FRAME_DATA_S	返回的 Raw 数据结构。

#### 【注意事项】

无

#### 【相关数据类型及接口】

[HI\\_MAPI\\_VCap\\_DumpRaw](#)

## HI\_FRAME\_DATA\_S

#### 【说明】



帧数据结构。

#### 【定义】

```
typedef struct hiFRAME_DATA_S
{
    HI_FRAME_DATA_TYPE_E enFrameDataType;
    HI_U32                u32Width;
    HI_U32                u32Height;
    HI_MPP_PIXEL_FORMAT_E enPixelFormat;
    HI_U32                u32PhyAddr[3];
    HI_VOID*              pVirAddr[3];
    HI_U32                u32Stride[3];
    HI_U64                u64pts;
} HI_FRAME_DATA_S;
```

#### 【成员】

成员名称	描述
enFrameDataType	数据类型，Raw 或者 YUV。
u32Width	Raw 数据的宽。
u32Height	Raw 数据的高。
enPixelFormat	数据格式： HI_MPP_PIXEL_FORMAT_RGB_BAYER_8BPP HI_MPP_PIXEL_FORMAT_RGB_BAYER_10BPP HI_MPP_PIXEL_FORMAT_RGB_BAYER_12BPP HI_MPP_PIXEL_FORMAT_RGB_BAYER_14BPP HI_MPP_PIXEL_FORMAT_RGB_BAYER
u32PhyAddr	Raw 数据的物理地址
u32Stride	Raw 数据的 Stride
u64pts	数据的 PTS

#### 【注意事项】

无

#### 【相关数据类型及接口】

[PFN\\_VCAP\\_RawDataProc](#)

HI\_FRAME\_DATA\_TYPE\_E

#### 【说明】



帧数据类型。

#### 【定义】

```
typedef enum hiFRAME_DATA_TYPE_E
{
    HI_FRAME_DATA_TYPE_RAW,
    HI_FRAME_DATA_TYPE_YUV,
    HI_FRAME_DATA_TYPE_BUTT
} HI_FRAME_DATA_TYPE_E;
```

#### 【成员】

成员名称	描述
HI_FRAME_DATA_TYPE_RAW	Raw 数据类型。
HI_FRAME_DATA_TYPE_YUV	YUV 数据类型。

#### 【注意事项】

无

#### 【相关数据类型及接口】

[HI\\_FRAME\\_DATA\\_S](#)

## HI\_MPP\_RESOLUTION\_S

#### 【说明】

定义分辨率结构体。

#### 【定义】

```
typedef struct hiMPP_RESOLUTION_S
{
    HI_U32 u32Width;
    HI_U32 u32Height;
} HI_MPP_RESOLUTION_S;
```

#### 【成员】

成员名称	描述
u32Width	分辨率的宽。
u32Height	分辨率的高。

#### 【注意事项】



无

【相关数据类型及接口】

- [HI\\_MPP\\_SENSOR\\_ATTR\\_S](#)
- [HI\\_MPP\\_VCAP\\_ATTR\\_S](#)

## HI\_MPP\_VCAP\_CMD\_E

【说明】

定义 VCAP 高级属性的 CMD 值。

【定义】

```
typedef enum hiHI_MPP_VCAP_CMD_E
{
    HI_MPP_VCAP_CMD_SetExposure,
    HI_MPP_VCAP_CMD_GetExposure,
    HI_MPP_VCAP_CMD_SetWBAttr,
    HI_MPP_VCAP_CMD_GetWBAttr,
    HI_MPP_VCAP_CMD_SetSharpen,
    HI_MPP_VCAP_CMD_GetSharpen,
    HI_MPP_VCAP_CMD_SetBrightness,
    HI_MPP_VCAP_CMD_GetBrightness,
    HI_MPP_VCAP_CMD_SetSaturation,
    HI_MPP_VCAP_CMD_GetSaturation,
    HI_MPP_VCAP_CMD_SetLDC,
    HI_MPP_VCAP_CMD_GetLDC,
    HI_MPP_VCAP_CMD_BUTT
}HI_MPP_VCAP_CMD_E;
```

【成员】

成员名称	描述
HI_MPP_VCAP_CMD_SetExposure	设置曝光属性 CMD。
HI_MPP_VCAP_CMD_GetExposure	获取曝光属性 CMD。
HI_MPP_VCAP_CMD_SetWBAttr	设置 WB 属性 CMD。
HI_MPP_VCAP_CMD_GetWBAttr	获取 WB 属性 CMD。
HI_MPP_VCAP_CMD_SetSharpen	设置 Sharpen 属性 CMD。
HI_MPP_VCAP_CMD_GetSharpen	获取 Sharpen 属性 CMD。
HI_MPP_VCAP_CMD_SetBrightness	设置亮度属性 CMD。
HI_MPP_VCAP_CMD_GetBrightness	获取亮度属性 CMD。





成员名称	描述
HI_MPP_VCAP_CMD_SetSaturation	设置饱和度属性 CMD。
HI_MPP_VCAP_CMD_GetSaturation	获取饱和度属性 CMD。
HI_MPP_VCAP_CMD_SetLDC	设置畸变校正属性 CMD。
HI_MPP_VCAP_CMD_GetLDC	获取畸变校正属性 CMD。

【注意事项】

无

【相关数据类型及接口】

- [HI\\_MAPI\\_VCap\\_SetAttrEx](#)
- [HI\\_MAPI\\_VCap\\_GetAttrEx](#)

## HI\_MPP\_VCAP\_OP\_TYPE\_E

【说明】

定义 VCAP 高级属性的操作模式。

【定义】

```
typedef enum hiHI_MPP_VCAP_OP_TYPE_E
{
    HI_MPP_VCAP_OP_TYPE_AUTO = 0,
    HI_MPP_VCAP_OP_TYPE_MANUAL = 1,
    HI_MPP_VCAP_OP_TYPE_DISABLE = 2,
    HI_MPP_VCAP_OP_TYPE_BUTT
} HI_MPP_VCAP_OP_TYPE_E;
```

【成员】

成员名称	描述
HI_MPP_VCAP_OP_TYPE_AUTO	自动模式。
HI_MPP_VCAP_OP_TYPE_MANUAL	手动模式。
HI_MPP_VCAP_OP_TYPE_DISABLE	关闭。

【注意事项】

无

【相关数据类型及接口】

- [HI\\_MAPI\\_VCap\\_SetAttrEx](#)



- [HI\\_MAPI\\_VCap\\_GetAttrEx](#)
- [HI\\_MPP\\_VCAP\\_EXPOSURE\\_ATTR\\_S](#)
- [HI\\_MPP\\_VCAP\\_WB\\_ATTR\\_S](#)
- [HI\\_MPP\\_VCAP\\_SHARPEN\\_S](#)

## HI\_MPP\_VCAP\_EXPOSURE\_MANUAL\_MODE\_S

### 【说明】

定义手动曝光属性。

### 【定义】

```
typedef struct hiHI_MPP_VCAP_EXPOSURE_MANUAL_MODE_S
{
    HI_U32 u32ExposureTime;    /**< Unit:us*/
} HI_MPP_VCAP_EXPOSURE_MANUAL_MODE_S;
```

### 【成员】

成员名称	描述
u32ExposureTime	曝光时长，单位：us。默认值为 0x4000。 取值范围：[0x0, 0xFFFFFFFF]，具体范围与 sensor 相关。

### 【注意事项】

设置曝光时间之后，如果想要恢复到设置之前的曝光，则需要重新设置为之前的值。

### 【相关数据类型及接口】

[HI\\_MPP\\_VCAP\\_EXPOSURE\\_ATTR\\_S](#)

## HI\_MPP\_VCAP\_AE\_RANGE\_S

### 【说明】

定义曝光增益的最大值和最小值。

### 【定义】

```
typedef struct hiHI_MPP_VCAP_AE_RANGE_S
{
    HI_U32 u32Min;
    HI_U32 u32Max;
} HI_MPP_VCAP_AE_RANGE_S;
```

### 【成员】



成员名称	描述
u32Min	曝光增益的最小值。
u32Max	曝光增益的最大值。

【注意事项】

无

【相关数据类型及接口】

[HI\\_MPP\\_VCAP\\_EXPOSURE\\_AUTO\\_MODE\\_S](#)

## HI\_MPP\_VCAP\_EXPOSURE\_AUTO\_MODE\_S

【说明】

定义自动曝光属性。

【定义】

```
typedef struct hiHI_MPP_VCAP_EXPOSURE_AUTO_MODE_S
{
    HI_U32 u32EVBias;          /**< [0,0xFFFF] */
    HI\_MPP\_VCAP\_AE\_RANGE\_S stSysGainRange;    /**ISO*/
}HI_MPP_VCAP_EXPOSURE_AUTO_MODE_S;
```

【成员】

成员名称	描述
u32EVBias	自动曝光调整时的曝光量偏差值，10bit 小数精度。 取值范围：[0x0, 0xFFFF]，默认值为 0x400。
stSysGainRange	系统增益范围，设置最大值和最小值，10bit 小数精度。 取值范围：[0x400, 0xFFFFFFFF]，具体范围与 sensor 相关。

【注意事项】

无

【相关数据类型及接口】

[HI\\_MPP\\_VCAP\\_EXPOSURE\\_ATTR\\_S](#)

## HI\_MPP\_VCAP\_EXPOSURE\_ATTR\_S

【说明】



定义曝光属性。

#### 【定义】

```
typedef struct hiHI_MPP_VCAP_EXPOSURE_ATTR_S
{
    HI_MPP_VCAP_OP_TYPE_E enOpType;
    HI_MPP_VCAP_EXPOSURE_MANUAL_MODE_S stMExposureMode;
    HI_MPP_VCAP_EXPOSURE_AUTO_MODE_S stAExposureMode;
} HI_MPP_VCAP_EXPOSURE_ATTR_S;
```

#### 【成员】

成员名称	描述
enOpType	曝光属性开关。
stMExposureMode	手动曝光属性。
stAExposureMode	自动曝光属性。

#### 【注意事项】

无

#### 【相关数据类型及接口】

- [HI\\_MPP\\_VCAP\\_EXPOSURE\\_MANUAL\\_MODE\\_S](#)
- [HI\\_MPP\\_VCAP\\_EXPOSURE\\_AUTO\\_MODE\\_S](#)
- [HI\\_MPP\\_VCAP\\_OP\\_TYPE\\_E](#)
- [HI\\_MAPI\\_VCap\\_SetAttrEx](#)
- [HI\\_MAPI\\_VCap\\_GetAttrEx](#)

## HI\_MPP\_VCAP\_WB\_MANUAL\_MODE\_S

#### 【说明】

定义手动 WB 属性。

#### 【定义】

```
typedef struct hiHI_MPP_VCAP_WB_MANUAL_MODE_S
{
    HI_U32 u32ColorTemp;                /**< Unit:Kelvin,1500~15000*/
} HI_MPP_VCAP_WB_MANUAL_MODE_S;
```

#### 【成员】

成员名称	描述
u32ColorTemp	当前色温值。取值范围：[1500, 15000]



【注意事项】

无

【相关数据类型及接口】

[HI\\_MPP\\_VCAP\\_WB\\_ATTR\\_S](#)

## HI\_MPP\_VCAP\_WB\_ATTR\_S

【说明】

定义 WB 属性。

【定义】

```
typedef struct hiHI_MPP_VCAP_WB_ATTR_S
{
    HI\_MPP\_VCAP\_OP\_TYPE\_E enOpType;
    HI\_MPP\_VCAP\_WB\_MANUAL\_MODE\_S stWBMode;
} HI_MPP_VCAP_WB_ATTR_S;
```

【成员】

成员名称	描述
enOpType	WB 的模式：手动、自动、关闭。
stWBMode	手动模式下的属性。

【注意事项】

无

【相关数据类型及接口】

- [HI\\_MPP\\_VCAP\\_OP\\_TYPE\\_E](#)
- [HI\\_MPP\\_VCAP\\_WB\\_MANUAL\\_MODE\\_S](#)
- [HI\\_MAPI\\_VCap\\_SetAttrEx](#)
- [HI\\_MAPI\\_VCap\\_GetAttrEx](#)

## HI\_MPP\_VCAP\_SHARPEN\_MANUAL\_ATTR\_S

【说明】

定义手动 sharpen 属性。

【定义】

```
typedef struct hiHI_MPP_VCAP_SHARPEN_MANUAL_ATTR_S
{
```



```
HI_S32 s32Sharpness;          /**> [0,255]*/  
}HI_MPP_VCAP_SHARPEN_MANUAL_ATTR_S;
```

#### 【成员】

成员名称	描述
s32Sharpness	Sharpen 强度。取值范围：[0, 255]。

#### 【注意事项】

无

#### 【相关数据类型及接口】

[HI\\_MPP\\_VCAP\\_SHARPEN\\_S](#)

## HI\_MPP\_VCAP\_SHARPEN\_S

#### 【说明】

定义 sharpen 属性。

#### 【定义】

```
typedef struct hiHI_MPP_VCAP_SHARPEN_S  
{  
    HI\_MPP\_VCAP\_OP\_TYPE\_E enOpType;  
    HI\_MPP\_VCAP\_SHARPEN\_MANUAL\_ATTR\_S stSharpenManualAttr;  
}HI_MPP_VCAP_SHARPEN_S;
```

#### 【成员】

成员名称	描述
enOpType	Sharpen 模式，手动、自动、关闭。
stSharpenManualAttr	手动 Sharpen 属性。

#### 【注意事项】

无

#### 【相关数据类型及接口】

- [HI\\_MPP\\_VCAP\\_OP\\_TYPE\\_E](#)
- [HI\\_MPP\\_VCAP\\_SHARPEN\\_MANUAL\\_ATTR\\_S](#)
- [HI\\_MAPI\\_VCap\\_SetAttrEx](#)
- [HI\\_MAPI\\_VCap\\_GetAttrEx](#)



## HI\_MPP\_VCAP\_BRIGHTNESS\_S

### 【说明】

定义亮度属性。

### 【定义】

```
typedef struct hiHI_MPP_VCAP_BRIGHTNESS_S
{
    HI_S32 s32Brightness;          /**> [0,100]*/
}HI_MPP_VCAP_BRIGHTNESS_S;
```

### 【成员】

成员名称	描述
s32Brightness	亮度调节参数。默认取值：50，取值范围：[0, 100]。

### 【注意事项】

无

### 【相关数据类型及接口】

- [HI\\_MAPI\\_VCap\\_SetAttrEx](#)
- [HI\\_MAPI\\_VCap\\_GetAttrEx](#)

## HI\_MPP\_VCAP\_SATURATION\_S

### 【说明】

定义饱和度属性。

### 【定义】

```
typedef struct hiHI_MPP_VCAP_SATURATION_S
{
    HI_S32 s32Saturation;          /**> [0,100]*/
}HI_MPP_VCAP_SATURATION_S;
```

### 【成员】

成员名称	描述
s32Saturation	饱和度调节参数。默认取值：50.取值范围：[0, 100]

### 【注意事项】

无

### 【相关数据类型及接口】



- [HI\\_MAPI\\_VCap\\_SetAttrEx](#)
- [HI\\_MAPI\\_VCap\\_GetAttrEx](#)

## HI\_MPP\_VCAP\_LDC\_TYPE\_E

### 【说明】

定义镜头畸变校正的类型。

### 【定义】

```
typedef enum hiHI_MPP_VCAP_LDC_TYPE_E
{
    HI_MPP_VCAP_LDC_TYPE_ALL = 0,
    HI_MPP_VCAP_LDC_TYPE_CROP,
    HI_MPP_VCAP_LDC_TYPE_BUTT
}HI_MPP_VCAP_LDC_TYPE_E;
```

### 【成员】

成员名称	描述
HI_MPP_VCAP_LDC_TYPE_ALL	全模式，从校正后的图像按边缘获取最大矩形，并缩小到原始图像大小，边缘数据会尽可能保留。
HI_MPP_VCAP_LDC_TYPE_CROP	裁剪模式。

### 【注意事项】

无

### 【相关数据类型及接口】

[HI\\_MPP\\_VCAP\\_LDC\\_ATTR\\_S](#)

## HI\_MPP\_VCAP\_LDC\_ATTR\_S

### 【说明】

定义镜头畸变校正属性。

### 【定义】

```
typedef struct hiHI_MPP_VCAP_LDC_ATTR_S
{
    HI_BOOL bEnable;
    HI\_MPP\_VCAP\_LDC\_TYPE\_E enLDCType;
    HI_S32 s32CenterXOffset;
    HI_S32 s32CenterYOffset;
    HI_S32 s32Ratio;
```





```
HI_S32 s32MinRatio;  
}HI_MPP_VCAP_LDC_ATTR_S;
```

#### 【成员】

成员名称	描述
bEnable	是否开启镜头畸变校正。
enLDCType	畸变校正类型，裁剪模式、全模式。
s32CenterXOffset	畸变中心点相对图象中心点水平偏移。
s32CenterYOffset	畸变中心点相对图象中心点垂直偏移。
s32Ratio	畸变程度。
s32MinRatio	辅助 s32Ratio 进行 LDC 校正，因为在校正的过程可能会存在黑边或跳变，从而影响效果，可调节此变量来避免黑边或跳变。

#### 【注意事项】

- 取值范围：
  - s32CenterXOffset 取值范围：[-127, 127]
  - s32CenterYOffset 取值范围：[-127, 127]
  - s32Ratio 取值范围：[-300, 300]，其中[-300, 0]对应枕形模式，[0, 300]对应桶形模式。
  - s32MinRatio 取值范围：[-300, 0]。
    - 该参数仅对 HI\_MPP\_VCAP\_LDC\_TYPE\_CROP 模式生效，目的是为了消除枕形模式下 s32Ratio 遍历时的不连续跳变现象，并保证  $s32Ratio \geq s32MinRatio$  时没有黑边；对于桶形模式，则无跳变和黑边问题，代价是桶形会丢失更多的图像信息。
    - 在 HI\_MPP\_VCAP\_LDC\_TYPE\_ALL 模式下，该参数无效，且桶形和枕形校正均没有黑边，但仍然存在不连续现象。

在 HI\_MPP\_VCAP\_LDC\_TYPE\_ALL 模式下，该参数无效，且桶形和枕形校正均没有黑边，但仍然存在不连续现象。

#### 【相关数据类型及接口】

- [HI\\_MPP\\_VCAP\\_LDC\\_TYPE\\_E](#)
- [HI\\_MAPI\\_VCap\\_SetAttrEx](#)
- [HI\\_MAPI\\_VCap\\_GetAttrEx](#)

## HI\_MPP\_SNAP\_EXIF\_INFO\_S

#### 【说明】

定义抓拍的 Exif 信息。



### 【定义】

```
typedef struct hiHI_MPP_SNAP_EXIF_INFO_S
{
    HI_U8 au8ImageDescription[HI_SNAP_EXIF_DRSCRIPTION_LENGTH];
    HI_U8 au8Make[HI_SNAP_EXIF_DRSCRIPTION_LENGTH];
    HI_U8 au8Model[HI_SNAP_EXIF_DRSCRIPTION_LENGTH];
    HI_U8 au8Software[HI_SNAP_EXIF_DRSCRIPTION_LENGTH];
} HI_MPP_SNAP_EXIF_INFO_S;
```

### 【成员】

成员名称	描述
au8ImageDescription	图像描述、来源，指生成图像的工具。 数据格式：ascii strings，长度最大 32。
au8Make	生产者，指产品生产厂家。 数据格式：ascii strings，长度最大 32。
au8Model	型号，指设备型号。 数据格式：ascii strings，长度最大 32。
au8Software	软件，显示固件 Firmware 版本。 数据格式：ascii strings，长度最大 32。

### 【芯片差异】

无

### 【注意事项】

无

### 【相关数据类型及接口】

- [HI\\_SNAP\\_EXIF\\_DRSCRIPTION\\_LENGTH](#)
- [HI\\_MAPI\\_VCap\\_SetExifInfo](#)
- [HI\\_MAPI\\_VCap\\_GetExifInfo](#)

## HI\_MPP\_OSD\_DISPATTR\_S

### 【说明】

定义 OSD 的显示属性。

### 【定义】

```
typedef struct hiHI_MPP_OSD_DISPATTR_S
{
    HI_BOOL bShow;
```



```
        HI_U32                u32Color;  
        HI_U32                u32Alpha;  
        HI_S32                s32RegionX;  
        HI_S32                s32RegionY;  
    } HI_MPP_OSD_DISPATTR_S;
```

#### 【成员】

成员名称	描述
bShow	区域是否显示。 取值范围：HI_TRUE 或者 HI_FALSE。 动态属性。
u32Color	区域颜色。取值范围：无。 动态属性。
u32Alpha	取值范围：[0, 255]。取值越小，越透明。 动态属性。
s32RegionX	区域的水平位置。[0, 8190]，要求以 2 对齐。
s32RegionY	区域的垂直位置。[0, 8190]，要求以 2 对齐。

#### 【芯片差异】

无

#### 【注意事项】

无

#### 【相关数据类型及接口】

[HI\\_MPP\\_OSD\\_ATTR\\_S](#)

## HI\_MPP\_OSD\_BITMAP\_ATTR\_S

#### 【说明】

定义 BMP 图片的相关属性。

#### 【定义】

```
typedef struct hiHI_MPP_OSD_BITMAP_ATTR_S  
{  
    HI_MPP_PIXEL_FORMAT_E    enPixelFormat;  
    HI_U32                    u32Width;  
    HI_U32                    u32Height;  
    HI_VOID*                  pData;  
} HI_MPP_OSD_BITMAP_ATTR_S;
```



【成员】

成员名称	描述
enPixelFormat	位图像素格式。 取值范围：HI_MPP_PIXEL_FORMAT_RGB_1555， HI_MPP_PIXEL_FORMAT_RGB_8888。
u32Width	位图宽。
u32Height	位图高。
pData	位图内容指针。

【注意事项】

无

【相关数据类型及接口】

- [HI\\_MPP\\_OSD\\_ATTR\\_S](#)
- [HI\\_MPP\\_PIXEL\\_FORMAT\\_E](#)

## HI\_MPP\_OSD\_ATTR\_S

【说明】

定义 OSD 属性。

【定义】

```
typedef struct hiHI_MPP_OSD_ATTR_S
{
    HI_MPP_OSD_BITMAP_ATTR_S stBitmapAttr;
    HI_MPP_OSD_DISPATTR_S stOsdDisplayAttr;
} HI_MPP_OSD_ATTR_S;
```

【成员】

成员名称	描述
stBitmapAttr	OSD 区域中填充的位图的相关属性。
stOsdDisplayAttr	OSD 区域的显示属性。

【芯片差异】

无

【注意事项】



在 stBitmapAttr 的位图数据在使用完成之后，需对内存地址进行释放，否则会造成内存泄露。

#### 【相关数据类型及接口】

- [HI\\_MPP\\_OSD\\_BITMAP\\_ATTR\\_S](#)
- [HI\\_MPP\\_OSD\\_DISPATTR\\_S](#)
- [HI\\_MAPI\\_VCap\\_OSD\\_SetAttr](#)
- [HI\\_MAPI\\_VCap\\_OSD\\_GetAttr](#)

## HI\_MPP\_PIXEL\_FORMAT\_E

#### 【说明】

定义像素格式。

#### 【定义】

```
typedef enum hiMPP_PIXEL_FORMAT_E
{
    HI_MPP_PIXEL_FORMAT_420 = 0,
    HI_MPP_PIXEL_FORMAT_422,
    HI_MPP_PIXEL_FORMAT_RGB_1555,
    HI_MPP_PIXEL_FORMAT_RGB_8888,
    HI_MPP_PIXEL_FORMAT_RGB_BAYER_8BPP,           /**<Raw 8bit*/
    HI_MPP_PIXEL_FORMAT_RGB_BAYER_10BPP,          /**<Raw 10bit*/
    HI_MPP_PIXEL_FORMAT_RGB_BAYER_12BPP,          /**<Raw 12bit*/
    HI_MPP_PIXEL_FORMAT_RGB_BAYER_14BPP,          /**<Raw 14bit*/
    HI_MPP_PIXEL_FORMAT_RGB_BAYER,                /**<Raw 16bit*/
    HI_MPP_PIXEL_FORMAT_BUTT
} HI_MPP_PIXEL_FORMAT_E;
```

#### 【成员】

成员名称	描述
HI_MPP_PIXEL_FORMAT_420	YUV420
HI_MPP_PIXEL_FORMAT_422	YUV422
HI_MPP_PIXEL_FORMAT_RGB_1555	RGB1555
HI_MPP_PIXEL_FORMAT_RGB_8888	RGB8888
HI_MPP_PIXEL_FORMAT_RGB_BAYER_8BPP	Raw 8Bit
HI_MPP_PIXEL_FORMAT_RGB_BAYER_10BPP	Raw 10bit
HI_MPP_PIXEL_FORMAT_RGB_BAYER_12BPP	Raw 12Bit
HI_MPP_PIXEL_FORMAT_RGB_BAYER_14BPP	Raw 14Bit
HI_MPP_PIXEL_FORMAT_RGB_BAYER	Raw 16Bit



【注意事项】

无

【相关数据类型及接口】

- [HI\\_MPP\\_VCAP\\_ATTR\\_S](#)
- [HI\\_MPP\\_OSD\\_BITMAP\\_ATTR\\_S](#)

## 2.7 错误码

视频采集 API 错误码如表 2-3 所示。

表2-3 视频采集 API 错误码

错误代码	宏定义	描述
0xA3018003	HI_ERR_MAPI_VCAP_ILLEGAL_PARA	参数设置无效
0xA3018006	HI_ERR_MAPI_VCAP_NULL_PTR	参数空指针错误
0xA3018008	HI_ERR_MAPI_VCAP_NOTSUPPORT	操作不支持
0xA3018009	HI_ERR_MAPI_VCAP_NOT_PERM	操作不允许
0xA301800C	HI_ERR_MAPI_VCAP_NOMEM	分配内存失败
0xA3018040	HI_ERR_MAPI_VCAP_INVALID_FD	非法 FD
0xA3018041	HI_ERR_MAPI_VCAP_IOCTL_FAIL	IOCTL 错误
0xA3018042	HI_ERR_MAPI_VCAP_OVERRANGE	参数越界
0xA3018043	MPP_EN_VCAP_NOT_INITED	视频采集模块未初始化
0xA3018044	MPP_EN_VCAP_NOT_STARTED	视频采集模块未启动



## 目 录

3 视频处理.....	3-1
3.1 概述.....	3-1
3.2 功能描述.....	3-1
3.2.1 基本概念 .....	3-1
3.2.2 功能描述 .....	3-1
3.3 API 参考 .....	3-2
3.4 数据类型.....	3-19
3.5 错误码.....	3-28



## 插图目录

图 3-1 VProc 上下文关系 .....	3-2
图 3-2 VProc 处理流程 .....	3-2





## 表格目录

表 3-1 VProc API 错误码 .....	3-28
---------------------------	------



# 3 视频处理

## 3.1 概述

VProc（Video Process）支持对输入的 VCap 图像进行处理，如去噪、拍照算法处理、缩放旋转等处理，同时一路图像输入可以输出多路处理后的不同图像。

## 3.2 功能描述

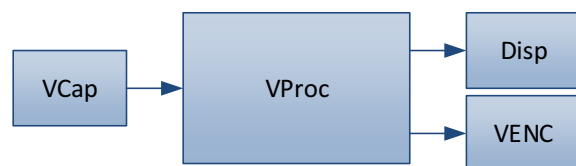
### 3.2.1 基本概念

- VProc  
VProc 可以同时创建多个，前端绑定 VCAP，各 VProc 之间同时进行图像处理。
- VPort  
Vport 是 Vproc 的输出端口，每个 VProc 支持同时输出 4 个 VPort。
- FRC  
帧率控制。
- NR  
去噪。通过参数配置，把图像中的高斯噪声去除，使得图像变得平滑，有助于降低编码码率。
- Scale  
缩放，对图像进行缩小放大。
- HDR  
高动态范围图像。拍照处理时，可以将多帧不同曝光值的图像合成为一帧高动态范围的图像。
- LL  
低照度图像。拍照处理时，将一帧或者多帧图像合成，增加图像的细节。

### 3.2.2 功能描述

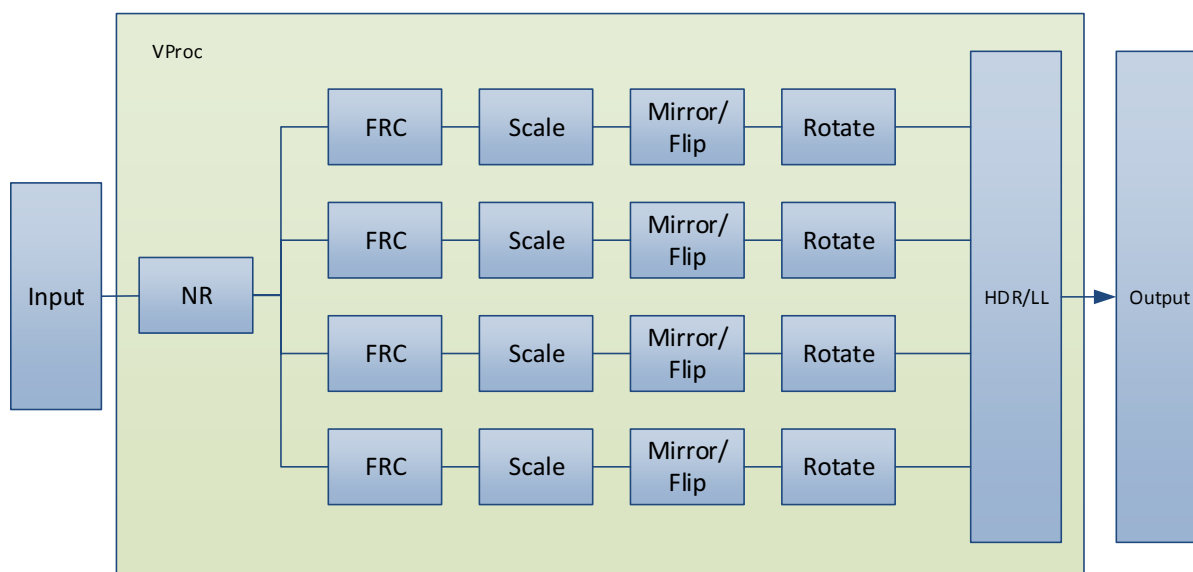
VProc 在系统中位置如图 3-1 所示。

图3-1 VProc 上下文关系



通过调用各模块的绑定接口，可与 VCAP 和 Disp/VENC 模块进行绑定，其中前者为 VProc 的输入源，后者为 VProc 处理后图像的接收者。

图3-2 VProc 处理流程



VProc 支持最多四个通道视频输出，分别为三个编码通道和一个显示通道。

### 3.3 API 参考

该功能模块为用户提供以下 MAPI:

- [HI\\_MAPI\\_VProc\\_Init](#): 初始化一个 Vproc 处理单元。
- [HI\\_MAPI\\_VProc\\_DeInit](#): 销毁一个 Vproc 处理单元。
- [HI\\_MAPI\\_VProc\\_Start](#): 启动 Vproc，开始接收输入图像。
- [HI\\_MAPI\\_VProc\\_Stop](#): 停止 Vproc 接收图像。
- [HI\\_MAPI\\_VProc\\_Bind\\_VCap](#): 将 Vproc 和 Vcap 绑定，建立连接通路。
- [HI\\_MAPI\\_VProc\\_UnBind\\_VCap](#): 将 Vproc 和 Vcap 解绑，断开连接。
- [HI\\_MAPI\\_VProc\\_Port\\_SetAttr](#): 配置 port 输出的图像属性。



- [HI\\_MAPI\\_VProc\\_Port\\_GetAttr](#): 获取当前配置的 port 输出属性。
- [HI\\_MAPI\\_VProc\\_Port\\_Start](#): 启动 port, 输出图像。
- [HI\\_MAPI\\_VProc\\_Port\\_Stop](#): 停止 port 输出。
- [HI\\_MAPI\\_VProc\\_SetAttrEx](#): Vproc 的高级属性设置接口。
- [HI\\_MAPI\\_VProc\\_GetAttrEx](#): Vproc 的高级属性获取接口。
- [HI\\_MAPI\\_VProc\\_SetPhotoAttr](#): 拍照算法处理参数设置。
- [HI\\_MAPI\\_VProc\\_GetPhotoAttr](#): 拍照算法处理参数获取。
- [HI\\_MAPI\\_VProc\\_PhotoProcess](#): 启动拍照算法处理。
- [HI\\_MAPI\\_VProc\\_EnableDumpYUV](#): 使能 port 输出的 YUV 数据获取。
- [HI\\_MAPI\\_VProc\\_DisableDumpYUV](#): 停止 port 输出的 YUV 数据获取。
- [HI\\_MAPI\\_VProc\\_DumpYUV](#): 开始获取 YUV 数据。

## HI\_MAPI\_VProc\_Init

### 【描述】

初始化一个 Vproc 处理单元。

### 【语法】

```
HI_S32 HI_MAPI_VProc_Init(HI_HANDLE VProcHdl, HI_VPROC_ATTR_S*  
pstVprocAttr);
```

### 【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围: [0, 32)。	输入
pstVprocAttr	VProc 属性指针。	输入

### 【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见 <a href="#">错误码</a> 。

### 【需求】

- 头文件: hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件: libhi3518e\_mapi\_vproc.a

### 【注意】



- VProc 的类型分为 VPROC\_TYPE\_VIDEO 和 VPROC\_TYPE\_SNAP 两种。VPROC\_TYPE\_VIDEO 主要是给码流和连拍场景使用；VPROC\_TYPE\_SNAP 主要是给单拍场景使用。

【举例】

无

【相关主题】

- [HI\\_MAPI\\_VProc\\_DeInit](#)
- [HI\\_MAPI\\_VProc\\_Start](#)

## HI\_MAPI\_VProc\_DeInit

【描述】

销毁一个 Vproc 处理单元。

【语法】

```
HI_S32 HI_MAPI_VProc_DeInit(HI_HANDLE VProcHdl);
```

【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】

- VProc 必须已初始化。
- 调用此接口之前，必须先调用 [HI\\_MAPI\\_VProc\\_Stop](#) 停用此 VProc。

【举例】

无



【相关主题】

[HI\\_MAPI\\_VProc\\_Init](#)

## HI\_MAPI\_VProc\_Start

【描述】

启动 Vproc，开始接收输入图像。

【语法】

```
HI_S32 HI_MAPI_VProc_Start(HI_HANDLE VProcHdl);
```

【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】

VProc 必须已初始化。

【举例】

无

【相关主题】

[HI\\_MAPI\\_VProc\\_Init](#)

## HI\_MAPI\_VProc\_Stop

【描述】

停止 VProc 接收图像。

【语法】



```
HI_S32 HI_MAPI_VProc_Stop(HI_HANDLE VProcHdl);
```

#### 【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

#### 【注意】

VProc 必须已经启动。

#### 【举例】

无

#### 【相关主题】

[HI\\_MAPI\\_VProc\\_Stop](#)

## HI\_MAPI\_VProc\_Bind\_VCap

#### 【描述】

将 Vproc 和 Vcap 绑定，用于接收 Vcap 的图像。

#### 【语法】

```
HI_S32 HI_MAPI_VProc_Bind_VCap(HI_HANDLE VCapHdl, HI_HANDLE VProcHdl);
```

#### 【参数】

参数名称	描述	输入/输出
VCapHdl	VCap handle 号。 取值范围：[0, 2)。	输入



参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】

无

【举例】

无

【相关主题】

无

## HI\_MAPI\_VProc\_UnBind\_VCap

【描述】

将 Vproc 和 Vcap 解绑，断开连接。

【语法】

```
HI_S32 HI_MAPI_VProc_UnBind_VCap(HI_HANDLE VCapHdl, HI_HANDLE VProcHdl);
```

【参数】

参数名称	描述	输入/输出
VCapHdl	VCap handle 号。 取值范围：[0, 2)。	输入
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入





【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】

VCap 和 VProc 必须已经绑定。

【举例】

无

【相关主题】

无

## HI\_MAPI\_VProc\_Port\_SetAttr

【描述】

配置 port 输出的图像属性。

【语法】

```
HI_S32 HI_MAPI_VProc_Port_SetAttr(HI_HANDLE VProcHdl, HI_HANDLE VPortHdl,  
HI\_VPORT\_ATTR\_S *pstVPortAttr);
```

【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入
VPortHdl	Port handle 号。 取值范围：[0, 4)。	输入
pstVPortAttr	Port 输出图像属性信息结构体。	输入

【返回值】



返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】

图像输出的像素格式可以和输入的像素格式相同，也可以做 SP422 到 SP420 的转换，但不支持 SP420 到 SP422 的转换。

【举例】

无

【相关主题】

无

## HI\_MAPI\_VProc\_Port\_GetAttr

【描述】

获取当前配置的 port 输出属性。

【语法】

```
HI_S32 HI_MAPI_VProc_Port_GetAttr(HI_HANDLE VProcHdl, HI_HANDLE VPortHdl,  
HI_VPORT_ATTR_S *pstVPortAttr);
```

【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入
VPortHdl	Port handle 号。 取值范围：[0, 4)。	输入
pstVPortAttr	Port 输出图像属性信息结构体。	输出

【返回值】



返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】

无

【举例】

无

【相关主题】

[HI\\_MAPI\\_VProc\\_Port\\_SetAttr](#)

## HI\_MAPI\_VProc\_Port\_Start

【描述】

启动 port，输出图像。

【语法】

```
HI_S32 HI_MAPI_VProc_Port_Start(HI_HANDLE VProcHdl, HI_HANDLE VPortHdl);
```

【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入
VPortHdl	Port handle 号。 取值范围：[0, 4)。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。



【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】

无

【举例】

无

【相关主题】

无

## HI\_MAPI\_VProc\_Port\_Stop

【描述】

停止 port 输出。

【语法】

```
HI_S32 HI_MAPI_VProc_Port_Stop(HI_HANDLE VProcHdl, HI_HANDLE VPortHdl);
```

【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入
VPortHdl	Port handle 号。 取值范围：[0, 4)。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】



无

【举例】

无

【相关主题】

无

## HI\_MAPI\_VProc\_SetAttrEx

【描述】

设置 VProc 的高级属性接口。

【语法】

```
HI_S32 HI_MAPI_VProc_SetAttrEx(HI_HANDLE VProcHdl, HI_HANDLE VPortHdl,  
HI_VPROC_CMD_E enCMD, HI_VOID* pAttr, HI_U32 u32Len);
```

【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入
VPortHdl	Port handle 号。 取值范围：[0, 4)。	输入
enCMD	需要设置的那个命令的枚举值。	输入
pAttr	需要设置的那个命令的具体属性值。	输入
u32Len	需要设置的那个命令的属性结构体长度	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】



无

【举例】

无

【相关主题】

无

## HI\_MAPI\_VProc\_GetAttrEx

【描述】

获取 Vproc 高级属性的接口。

【语法】

```
HI_S32 HI_MAPI_VProc_GetAttrEx(HI_HANDLE VProcHdl, HI_HANDLE VPortHdl,  
HI_VPROC_CMD_E enCMD, HI_VOID* pAttr, HI_U32 u32Len);
```

【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入
VPortHdl	Port handle 号。 取值范围：[0, 4)。	输入
enCMD	需要获取的那个命令的枚举值。	输入
pAttr	需要获取的那个命令的具体属性值。	输出
u32Len	需要获取的那个命令的属性结构体长度	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】



无

【举例】

无

【相关主题】

无

## HI\_MAPI\_VProc\_SetPhotoAttr

【描述】

拍照算法处理参数设置。

【语法】

```
HI_S32 HI_MAPI_VProc_SetPhotoAttr(HI_HANDLE VProcHdl, HI_HANDLE VPortHdl,  
HI_PHOTO_PROC_ATTR_S* pstPhotoAttr);
```

【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入
VPortHdl	Port handle 号。 取值范围：[0, 4)。	输入
pstPhotoAttr	拍照处理算法参数结构体。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】

无

【举例】



无

【相关主题】

无

## HI\_MAPI\_VProc\_GetPhotoAttr

【描述】

拍照算法处理参数获取。

【语法】

```
HI_S32 HI_MAPI_VProc_GetPhotoAttr(HI_HANDLE VProcHdl, HI_HANDLE VPortHdl,  
HI_PHOTO_PROC_ATTR_S* pstPhotoAttr);
```

【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入
VPortHdl	Port handle 号。 取值范围：[0, 4)。	输入
pstPhotoAttr	拍照处理算法参数结构体	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】

无

【相关主题】

[HI\\_MAPI\\_VProc\\_SetPhotoAttr](#)





## HI\_MAPI\_VProc\_PhotoProcess

### 【描述】

启动拍照算法处理。

### 【语法】

```
HI_S32 HI_MAPI_VProc_PhotoProcess(HI_HANDLE VProcHdl, HI_HANDLE VPortHdl);
```

### 【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入
VPortHdl	Port handle 号。 取值范围：[0, 4)。	输入

### 【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

### 【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

### 【注意】

- 每次调用前必须调用 HI\_MAPI\_VProc\_SetPhotoAttr 设置相关的拍照算法参数。
- 拍 HDR/LL 照片，需要依次调用 HI\_MAPI\_VCap\_SetSnapAttr，HI\_MAPI\_VProc\_SetPhotoAttr，HI\_MAPI\_VCap\_Trigger，HI\_MAPI\_VProc\_PhotoProcess 这四个接口。
- 该接口不支持多个线程同时调用。

### 【举例】

无

### 【相关主题】

[HI\\_MAPI\\_VProc\\_SetPhotoAttr](#)



## HI\_MAPI\_VProc\_EnableDumpYUV

### 【描述】

使能 port 输出的 YUV 数据获取。

### 【语法】

```
HI_S32 HI_MAPI_VProc_EnableDumpYUV(HI_HANDLE VProcHdl, HI_HANDLE  
VPortHdl);
```

### 【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入
VPortHdl	Port handle 号。 取值范围：[0, 4)。	输入

### 【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

### 【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

### 【注意】

无

### 【相关主题】

无

## HI\_MAPI\_VProc\_DisableDumpYUV

### 【描述】

停止 port 输出的 YUV 数据获取。

### 【语法】

```
HI_S32 HI_MAPI_VProc_DisableDumpYUV(HI_HANDLE VProcHdl, HI_HANDLE  
VPortHdl);
```



【参数】

参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入
VPortHdl	Port handle 号。 取值范围：[0, 4)。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

【注意】

调用该接口前要保证 dump YUV 已经使能。

【举例】

无

【相关主题】

[HI\\_MAPI\\_VProc\\_EnableDumpYUV](#)

## HI\_MAPI\_VProc\_DumpYUV

【描述】

开始获取 YUV 数据。

【语法】

```
HI_S32 HI_MAPI_VProc_DumpYUV(HI_HANDLE VProcHdl, HI_HANDLE VPortHdl,  
HI_S32 s32Count, PFN_VPROC_YUVDataProc pfunVProcYUVProc);
```

【参数】



参数名称	描述	输入/输出
VProcHdl	VProc handle 号。 取值范围：[0, 32)。	输入
VPortHdl	Port handle 号。 取值范围：[0, 4)。	输入
s32Count	Count 帧计数。 取值范围：[-1, 0xFFFFFFFF)。	输入
pfunVProcYUVProc	注册给 dump 函数的函数指针，用于处理 dump 的数据。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_vproc\_define.h、hi\_mapi\_vproc.h
- 库文件：libhi3518e\_mapi\_vproc.a

#### 【注意】

- 调用该接口前要保证 dump YUV 已经使能。
- 该接口和 [HI\\_MAPI\\_VProc\\_PhotoProcess](#) 接口，不支持在相同的 port 上同时调用。

#### 【举例】

无

#### 【相关主题】

无

## 3.4 数据类型

VProc 模块相关数据类型定义如下：

- [HI\\_VPROC\\_ATTR\\_S](#)：定义 Vproc 属性结构体。
- [HI\\_VPROC\\_TYPE\\_E](#)：定义 Vproc 处理类型的枚举值。
- [HI\\_VPORT\\_ATTR\\_S](#)：定义 Vport 属性结构体。



- [HI\\_VPROC\\_CMD\\_E](#): 定义 Vproc 高级参数设置命令的枚举。
- [HI\\_VPROC\\_MIRROR\\_ATTR\\_S](#): 定义高级参数中 Mirror 属性的结构体。
- [HI\\_VPROC\\_FLIP\\_ATTR\\_S](#): 定义高级参数中 Flip 属性的结构体。
- [HI\\_MPP\\_ROTATE\\_E](#): 定义高级参数中 Rotate 参数的枚举值。
- [HI\\_PHOTO\\_PROC\\_ATTR\\_S](#): 定义拍照算法处理参数的结构体。
- [HI\\_PHOTO\\_PROC\\_TYPE\\_E](#): 定义拍照算法类型的枚举值。
- [HI\\_PHOTO\\_HDR\\_ATTR\\_S](#): 定义 HDR 算法处理参数的结构体。
- [HI\\_PHOTO\\_LL\\_ATTR\\_S](#): 定义 LL 算法处理参数的结构体。
- [PFN\\_VPROC\\_YUVDataProc](#): 定义注册 dump YUV 数据处理函数的函数指针。

## HI\_VPROC\_ATTR\_S

### 【说明】

定义 Vproc 属性的结构体。

### 【定义】

```
typedef struct hiVPROC_ATTR_S
{
    HI\_VPROC\_TYPE\_E enVProcType;
    HI_U32 u32MaxW;
    HI_U32 u32MaxH;
} HI_VPROC_ATTR_S;
```

### 【成员】

成员名称	描述
enVProcType	Vproc 处理哪种类型的 Vcap 帧数据。
u32MaxW	需要处理的 VProc 输入数据的最大宽度。 取值范围: [64, 4608];
u32MaxH	需要处理的 VProc 输入数据的最大高度。 取值范围: [64, 4608];

### 【注意事项】

无

### 【相关数据类型及接口】

无

## HI\_VPROC\_TYPE\_E

### 【说明】



定义 Vproc 处理数据类型的枚举。

#### 【定义】

```
typedef enum hiVPROC_TYPE_E
{
    VPROC_TYPE_VIDEO = 0,
    VPROC_TYPE_SNAP,
    VPROC_TYPE_BUTT
}HI_VPROC_TYPE_E;
```

#### 【成员】

成员名称	描述
VPROC_TYPE_VIDEO	处理视频帧数据。
VPROC_TYPE_SNAP	处理拍照的帧数据。

#### 【注意事项】

当 Vproc 类型设置为 VPROC\_TYPE\_SNAP 时，只能接收 Vcap 的抓拍帧，无法接收视频帧。

#### 【相关数据类型及接口】

[HI\\_VPROC\\_ATTR\\_S](#)

## HI\_VPORT\_ATTR\_S

#### 【说明】

Port 输出图像信息。

#### 【定义】

```
typedef struct hiVPORT_ATTR_S
{
    HI_S32 s32FrameRate;
    HI_MPP_RESOLUTION_S stResolution;
    HI_MPP_PIXEL_FORMAT_E enPixFormat;
} HI_VPORT_ATTR_S;
```

#### 【成员】

成员名称	描述
s32FrameRate	Port 输出的帧率。 取值范围：[-1, 240]。
stResolution	Port 输出的分辨率。



成员名称	描述
enPixFormat	Port 输出的像素格式。

【注意事项】

无

【相关数据类型及接口】

[HI\\_MAPI\\_VProc\\_Port\\_SetAttr](#)

## HI\_VPROC\_CMD\_E

【说明】

Vproc 高级参数设置命令的枚举。

【定义】

```
typedef enum hiVPROC_CMD_E
{
    HI_VPROC_CMD_SetPortMirror,
    HI_VPROC_CMD_GetPortMirror,
    HI_VPROC_CMD_SetPortFlip,
    HI_VPROC_CMD_GetPortFlip,
    HI_VPROC_CMD_SetPortRotate,
    HI_VPROC_CMD_GetPortRotate,
    HI_VPROC_CMD_BUTT
}HI_VPROC_CMD_E;
```

【成员】

成员名称	描述
HI_VPROC_CMD_SetPortMirror	设置 port 输出的 Mirror 属性。
HI_VPROC_CMD_GetPortMirror	获取 port 输出的 Mirror 属性。
HI_VPROC_CMD_SetPortFlip	设置 port 输出的 Flip 属性。
HI_VPROC_CMD_GetPortFlip	获取 port 输出的 Flip 属性。
HI_VPROC_CMD_SetPortRotate	设置 port 输出的 Rotate 属性。
HI_VPROC_CMD_GetPortRotate	获取 port 输出的 Rotate 属性。

【注意事项】

无



【相关数据类型及接口】

- [HI\\_MAPI\\_VProc\\_SetAttrEx](#)
- [HI\\_MAPI\\_VProc\\_GetAttrEx](#)

## HI\_VPROC\_MIRROR\_ATTR\_S

【说明】

高级属性中 Mirror 属性的结构体。

【定义】

```
typedef struct hiVPROC_MIRROR_ATTR_S
{
    HI_BOOL bEnable;
}HI_VPROC_MIRROR_ATTR_S;
```

【成员】

成员名称	描述
bEnable	Port 输出是否开启 mirror。

【注意事项】

无

【相关数据类型及接口】

- [HI\\_MAPI\\_VProc\\_SetAttrEx](#)
- [HI\\_MAPI\\_VProc\\_GetAttrEx](#)

## HI\_VPROC\_FLIP\_ATTR\_S

【说明】

高级属性中 Flip 属性的结构体。

【定义】

```
typedef struct hiVPROC_FLIP_ATTR_S
{
    HI_BOOL bEnable;
}HI_VPROC_FLIP_ATTR_S;
```

【成员】

成员名称	描述
bEnable	Port 输出是否开启 flip。





【注意事项】

无

【相关数据类型及接口】

- [HI\\_MAPI\\_VProc\\_SetAttrEx](#)
- [HI\\_MAPI\\_VProc\\_GetAttrEx](#)

## HI\_MPP\_ROTATE\_E

【说明】

高级属性中 Rotate 属性的结构体。

【定义】

```
typedef enum hiMPP_ROTATE_E
{
    HI_ROTATE_NONE = 0,
    HI_ROTATE_90 = 1,
    HI_ROTATE_180 = 2,
    HI_ROTATE_270 = 3,
    HI_ROTATE_BUTT
} HI_MPP_ROTATE_E;
```

【成员】

成员名称	描述
HI_ROTATE_NONE	Port 输出不做旋转。
HI_ROTATE_90	Port 输出做 90 度旋转。
HI_ROTATE_180	Port 输出做 180 度旋转。
HI_ROTATE_270	Port 输出做 270 度旋转。

【注意事项】

无

【相关数据类型及接口】

- [HI\\_MAPI\\_VProc\\_SetAttrEx](#)
- [HI\\_MAPI\\_VProc\\_GetAttrEx](#)

## HI\_PHOTO\_PROC\_ATTR\_S

【说明】

拍照算法参数结构体。



### 【定义】

```
typedef struct hiPHOTO_PROC_ATTR_S
{
    HI_PHOTO_PROC_TYPE_E enPhotoType;
    union
    {
        HI_PHOTO_HDR_ATTR_S stHDRAttr;
        HI_PHOTO_LL_ATTR_S stLLAttr;
    };
}HI_PHOTO_PROC_ATTR_S;
```

### 【成员】

成员名称	描述
enPhotoType	拍照处理类型。
stHDRAttr	拍照算法 HDR 的参数。
stLLAttr	拍照算法 LL 的参数。

### 【注意事项】

无

### 【相关数据类型及接口】

- [HI\\_MAPI\\_VProc\\_SetPhotoAttr](#)
- [HI\\_MAPI\\_VProc\\_GetPhotoAttr](#)

## HI\_PHOTO\_PROC\_TYPE\_E

### 【说明】

定义拍照处理类型的枚举。

### 【定义】

```
typedef enum hiPHOTO_PROC_TYPE_E
{
    PHOTO_PROC_TYPE_LL = 0,
    PHOTO_PROC_TYPE_HDR,
    PHOTO_PROC_TYPE_BUTT
} HI_PHOTO_PROC_TYPE_E;
```

### 【成员】

成员名称	描述
PHOTO_PROC_TYPE_LL	LL 拍照处理。



成员名称	描述
PHOTO_PROC_TYPE_HDR	HDR 拍照处理。

【注意事项】

无

【相关数据类型及接口】

[HI\\_PHOTO\\_PROC\\_ATTR\\_S](#)

## HI\_PHOTO\_HDR\_ATTR\_S

【说明】

定义拍照 HDR 处理的参数。

【定义】

```
typedef struct hiPHOTO_PROC_HDR_ATTR_S
{
    HI_S32 s32NrLuma;
    HI_S32 s32NrChroma;
    HI_S32 s32Sharpen;
    HI_S32 s32Saturation;
    HI_S32 s32GlobalContrast;
    HI_S32 s32LocalContrast;
}HI_PHOTO_HDR_ATTR_S;
```

【成员】

成员名称	描述
s32NrLuma	亮度降噪强度。 取值范围：[0, 10]。
s32NrChroma	色度降噪强度。 取值范围：[0, 10]。
s32Sharpen	边沿锐度强度。 取值范围：[0, 10]。
s32Saturation	颜色饱和度。 取值范围：[0, 10]。
s32GlobalContrast	全局对比度。 取值范围：[0, 10]。



成员名称	描述
s32LocalContrast	局部对比度。 取值范围：[0, 10]。

【注意事项】

无

【相关数据类型及接口】

[HI\\_PHOTO\\_PROC\\_ATTR\\_S](#)

## HI\_PHOTO\_LL\_ATTR\_S

【说明】

定义拍照 LL 处理的参数。

【定义】

```
typedef struct hiPHOTO_PROC_LL_ATTR_S
{
    HI_S32 s32NrLuma;
    HI_S32 s32NrChroma;
    HI_S32 s32Sharpen;
    HI_S32 s32Saturation;
    HI_S32 s32Iso;
}HI_PHOTO_LL_ATTR_S;
```

【成员】

成员名称	描述
s32NrLuma	亮度降噪强度。 取值范围：[0, 21]。
s32NrChroma	色度降噪强度。 取值范围：[0, 10]。
s32Sharpen	边沿锐度强度。 取值范围：[0, 10]。
s32Saturation	颜色饱和度。 取值范围：[0, 10]。
s32Iso	图像 ISO 值； 取值范围：[0, 12800]。



【注意事项】

无

【相关数据类型及接口】

[HI\\_PHOTO\\_PROC\\_ATTR\\_S](#)

## PFN\_VPROC\_YUVDataProc

【说明】

处理 dump YUV 数据的函数指针。

【定义】

```
typedef HI_S32 (*PFN_VPROC_YUVDataProc)(HI_HANDLE VProcHdl, HI_HANDLE  
VPortHdl, HI_FRAME_DATA_S* pVPortYUV);
```

【成员】

成员名称	描述
VProcHdl	Vproc 的 handle 号。
VPortHdl	输出 YUV 数据的 port handle 号。
pVPortYUV	输出 YUV 数据的指针。

【注意事项】

无。

【相关数据类型及接口】

[HI\\_MAPI\\_VProc\\_DumpYUV](#)

## 3.5 错误码

视频处理 API 错误码如表 3-1 所示。

表3-1 VProc API 错误码

错误代码	宏定义	描述
0xA3028001	HI_MAPI_ERR_VPROC_INVALID_DEVID	VProc handle 号无效
0xA3028002	HI_MAPI_ERR_VPROC_INVALID_CHNID	Port handle 号无效
0xA3028003	HI_MAPI_ERR_VPROC_ILLEGAL_PARAM	VProc 参数设置无效
0xA3028004	HI_MAPI_ERR_VPROC_EXIST	VProc 已创建



错误代码	宏定义	描述
0xA3028005	HI_MAPI_ERR_VPROC_UNEXIST	VProc 未创建
0xA3028006	HI_MAPI_ERR_VPROC_NULL_PTR	输入参数空指针错误
0xA3028008	HI_MAPI_ERR_VPROC_NOT_SUPPORT	操作不支持
0xA3028009	HI_MAPI_ERR_VPROC_NOT_PERM	操作不允许
0xA302800C	HI_MAPI_ERR_VPROC_NOMEM	分配内存失败
0xA302800D	HI_MAPI_ERR_VPROC_NOBUF	分配 BUF 池失败
0xA302800E	HI_MAPI_ERR_VPROC_BUF_EMPTY	图像队列为空
0xA3028010	HI_MAPI_ERR_VPROC_NOTREADY	VProc 系统未初始化
0xA3028012	HI_MAPI_ERR_VPROC_BUSY	VProc 系统忙



## 目 录

4 视频编码.....	4-1
4.1 概述.....	4-1
4.2 功能描述.....	4-1
4.2.1 编码数据流程图.....	4-1
4.2.2 编码通道 .....	4-2
4.2.3 码率控制 .....	4-2
4.3 API 参考 .....	4-2
4.4 数据类型.....	4-17
4.5 错误码.....	4-34



## 插图目录

图 4-1 VENC 的数据流程图.....	4-1
------------------------	-----





## 表格目录

表 4-1 视频编码 API 错误码 .....	4-35
--------------------------	------



# 4 视频编码

## 4.1 概述

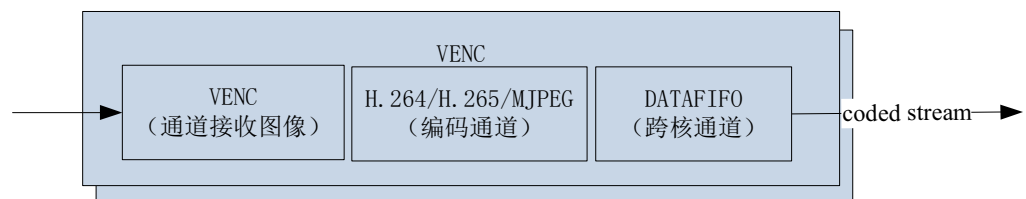
VENC 模块，即视频编码模块。本模块支持多路实时编码，且每路编码独立，编码协议和编码 profile 可以不同。模块包含码率控制和帧率控制以及其它高级编码属性设置。

VENC 模块的输入源当前仅支持 VPORT 输入。

## 4.2 功能描述

### 4.2.1 编码数据流程图

图4-1 VENC 的数据流程图



典型的编码流程包括了输入图像的接收、图像的编码、数据流的跨核传输以及码流的输出等过程。

VENC 模块由编码通道子模块（VENC）和编码协议子模块（H.264/H.265/JPEG/MJPEG）组成。

通道支持接收 YUV 格式图像输入，支持格式为 Semi-planar YUV 4:2:0 或 Semi-planar YUV 4:2:2，其中 H.264/H.265 只支持 Semi-planar YUV 4:2:0，JPEG/MJPEG 支持 Semi-planar YUV 4:2:0 或 Semi-planar YUV 4:2:2。通道模块当前仅支持接收 VPORT 模块输入的数据。



## 4.2.2 编码通道

编码通道作为基本容器，保存编码通道的多种用户设置和管理编码通道的多种内部资源。编码通道完成图像转化为码流的功能，具体由码率控制器和编码器协同完成。这里的编码器指的是狭义上的编码器，只完成编码功能。码率控制器提供了对编码参数的控制和调整，从而对输出码率进行控制。

## 4.2.3 码率控制

码率控制器实现对编码码率进行控制。

码率控制是针对连续的编码码流而言，所以，JPEG 协议编码通道不包括码率控制功能。

码率控制器分别提供了对 H.264/H.265/MJPEG 协议编码通道 CBR、VBR 两种码率控制模式，对图像质量和码率进行调节。

### 4.2.3.1 CBR

CBR (Constant Bit Rate) 固定比特率。即在码率统计时间内保证编码码率平稳。码率稳定主要由两个量来评估。

- 码率统计时间 `u32StatTime` 单位为秒(s)，码率统计时间越长，每帧图像的码率波动对于码率调节的影响越弱，码率的调节会更缓慢，图像质量的波动会更轻微；码率统计时间越短，每帧图像的码率波动对于码率调节的影响越强，图像码率的调节会更灵敏，图像质量的波动会更剧烈。
- 行级码率控制调节幅度 `u32RowQpDelta` 行级码率控制调节幅度是一帧内行级调节的最大范围，其中行级以宏块行为单位。调节幅度越大，允许行级调整的 QP 范围越大，码率越平稳。对于图像复杂度分布不均匀的场景，行级码率控制调节幅度设置过大会带来图像质量不均匀。

### 4.2.3.2 VBR

VBR (Variable Bit Rate) 可变比特率，即允许在码率统计时间内编码码率波动，从而保证编码图像质量平稳。以 H.264 编码为例，VENC 模块提供用户可设置 `MaxQp`，`MinQp`，`MaxBitrate` 和 `ChangePos`。`MaxBitrate` 可以在 API 接口中设置，`MaxQp`，`MinQp` 和 `ChangePos` 可以使用 PQTool 进行调节。`MaxQp`，`MinQp` 用于控制图像的质量范围，`MaxBitrate` 用于钳位码率统计时间内的最大编码码率，`ChangePos` 用于控制开始调整 Qp 的码率基准线。当编码码率大于 `MaxBitrate*ChangePos` 时，图像 qp 会逐步向 `MaxQp` 调整，如果图像 QP 达到 `MaxQp`，QP 会被钳位到最大值，`MaxBitrate` 的钳位效果失效，编码码率有可能会超出 `MaxBitrate`。当编码码率小于 `MaxBitrate*ChangePos` 时，图像 QP 会逐步向 `MinQp` 调整，如果图像 QP 达到 `MinQp`，此时编码的码率已经达到最大值，而且图像质量最好。

## 4.3 API 参考

视频编码模块主要提供视频编码通道的创建和销毁、视频编码通道的开启和停止接收图像、设置和获取编码通道属性、注册和解注册获取码流的回调函数等功能。

该功能模块提供以下 MAPI:



- [HI\\_MAPI\\_VEnc\\_Init](#): 初始化编码通道。
- [HI\\_MAPI\\_VEnc\\_DeInit](#): 去初始化编码通道。
- [HI\\_MAPI\\_VEnc\\_RegisterCallback](#): 注册获取回调函数。
- [HI\\_MAPI\\_VEnc\\_UnRegisterCallback](#): 解注册回调函数。
- [HI\\_MAPI\\_VEnc\\_Start](#): 启动编码通道。
- [HI\\_MAPI\\_VEnc\\_Stop](#): 停止编码通道。
- [HI\\_MAPI\\_VEnc\\_Bind\\_VProc](#): 绑定 VProc 输入端。
- [HI\\_MAPI\\_VEnc\\_UnBind\\_VProc](#): 解绑定 VProc。
- [HI\\_MAPI\\_VEnc\\_SetAttr](#): 设置编码通道参数。
- [HI\\_MAPI\\_VEnc\\_GetAttr](#): 获取编码通道参数。
- [HI\\_MAPI\\_VEnc\\_RequestIDR](#): 请求 IDR 帧。
- [HI\\_MAPI\\_VEnc\\_GetStreamHeadInfo](#): 获取码流头信息。
- [HI\\_MAPI\\_VEnc\\_SetThumbNailAttr](#): 设置缩略图参数。
- [HI\\_MAPI\\_VEnc\\_GetThumbNailAttr](#): 获取缩略图参数。

## HI\_MAPI\_VEnc\_Init

### 【描述】

初始化编码通道。

### 【语法】

```
HI_S32 HI_MAPI_VEnc_Init(HI_HANDLE hVencHdl, const HI_MPP_VENC_ATTR_S  
*pstVencAttr)
```

### 【参数】

参数名称	描述	输入/输出
hVencHdl	VENC 通道号。 取值范围: [0, HI_VENC_MAX_CHN_NUM)。	输入
pstVencAttr	VENC 通道属性指针。 静态属性。	输入

### 【返回值】

返回值	描述
0	成功。
非 0	失败, 其值为 <a href="#">错误码</a> 。

### 【芯片差异】



无。

【需求】

- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a

【注意】

- 调用该接口前需要先初始化 HI\_MAPI\_Sys\_Init()和 HI\_MAPI\_Media\_Init()成功，详见“系统控制”章节。
- 重复初始化返回成功。

【举例】

请参考 sample\_venc 代码。

【相关主题】

- [HI\\_MAPI\\_VEnc\\_DeInit](#)
- [HI\\_MPP\\_VENC\\_ATTR\\_S](#)

## HI\_MAPI\_VEnc\_DeInit

【描述】

编码通道去初始化。

【语法】

```
HI_S32 HI_MAPI_VEnc_DeInit(HI_HANDLE hVencHdl)
```

【参数】

参数名称	描述	输入/输出
hVencHdl	VENC 通道号。 取值范围：[0, HI_VENC_MAX_CHN_NUM)。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】



- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a

【注意】

无。

【举例】

无。

【相关主题】

[HI\\_MAPI\\_VEnc\\_Init](#)

## HI\_MAPI\_VEnc\_RegisterCallback

【描述】

注册编码通道回调函数，用于编码数据的获取。

【语法】

```
HI_S32 HI_MAPI_VEnc_RegisterCallback(HI_HANDLE hVencHdl,  
HI\_VENC\_CALLBACK\_S *pstVencCB)
```

【参数】

参数名称	描述	输入/输出
hVencHdl	VENC 通道号。 取值范围：[0, HI_VENC_MAX_CHN_NUM)。	输入
pstVencCB	编码器回调函数结构体指针。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】

- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a



【注意】

- 回调函数结构体指针不能为空。
- 结构体之内容完全一致时，才表示回调函数结构体相等。
- 每个通道最多支持五个回调函数。

【举例】

无。

【相关主题】

- [HI\\_MAPI\\_VEnc\\_UnRegisterCallback](#)
- [HI\\_VENC\\_CALLBACK\\_S](#)

## HI\_MAPI\_VEnc\_UnRegisterCallback

【描述】

解注册编码通道回调函数。

【语法】

```
HI_S32 HI_MAPI_VEnc_UnRegisterCallback(HI_HANDLE hVencHdl,  
HI\_VENC\_CALLBACK\_S *pstVencCB)
```

【参数】

参数名称	描述	输入/输出
hVencHdl	VENC 通道号。 取值范围：[0, HI_VENC_MAX_CHN_NUM)。	输入
pstVencCB	编码器回调函数结构体指针。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】

- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a



【注意】

回调函数结构体内容完全一致时才能解注册该回调函数。

【举例】

无。

【相关主题】

[HI\\_MAPI\\_VEnc\\_RegisterCallback](#)

## HI\_MAPI\_VEnc\_Start

【描述】

启动编码通道。

【语法】

```
HI_S32 HI_MAPI_VEnc_Start(HI_HANDLE hVencHdl, HI_S32 s32FrameCnt)
```

【参数】

参数名称	描述	输入/输出
hVencHdl	VENC 通道号。 取值范围：[0, HI_VENC_MAX_CHN_NUM)。	输入
s32FrameCnt	期望编码帧的个数。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】

- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a

【注意】

- 场景一：视频流编码场景下，s32FrameCnt 取值 [HI\\_VENC\\_LIMITLESS\\_FRAME\\_COUNT](#)。
- 场景二：单个或者多个数据编码的场景下，上一个编码完后才可以再次启动。





- 场景一切换场景二时，需要先调用 [HI\\_MAPI\\_VEnc\\_Stop](#) 接口。
- 场景二切换场景一时，可以先调用 [HI\\_MAPI\\_VEnc\\_Stop](#) 接口，也可以在编码完后再次启动编码。
- 编码完成的标记请查看 [HI\\_VENC\\_DATA\\_S](#) 中 bEndOfStream。

【举例】

无。

【相关主题】

[HI\\_MAPI\\_VEnc\\_Stop](#)

## HI\_MAPI\_VEnc\_Stop

【描述】

停止编码通道。

【语法】

```
HI_S32 HI_MAPI_VEnc_Stop(HI_HANDLE hVencHdl)
```

【参数】

参数名称	描述	输入/输出
hVencHdl	VENC 通道号。 取值范围：[0, HI_VENC_MAX_CHN_NUM)。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】

- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a

【注意】

无。

【举例】



无。

【相关主题】

[HI\\_MAPI\\_VEnc\\_Start](#)

## HI\_MAPI\_VEnc\_Bind\_VProc

【描述】

编码通道绑定输入源 VProc。

【语法】

```
HI_S32 HI_MAPI_VEnc_Bind_VProc(HI_HANDLE hVProcHdl, HI_HANDLE hVPortHdl,  
HI_HANDLE hVencHdl);
```

【参数】

参数名称	描述	输入/输出
hVProcHdl	VPORC 通道号。 取值范围：[0, 32)。	输入
hVPortHdl	VPORT 通道号 取值范围：[0, 12)。	输入
hVencHdl	VENC 通道号。 取值范围：[0, HI_VENC_MAX_CHN_NUM)。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】

- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a

【注意】

无。

【举例】



无。

【相关主题】

[HI\\_MAPI\\_VEnc\\_UnBind\\_VProc](#)

## HI\_MAPI\_VEnc\_UnBind\_VProc

【描述】

解绑定编码通道的输入源 VProc。

【语法】

```
HI_S32 HI_MAPI_VEnc_UnBind_VProc(HI_HANDLE hVProcHdl, HI_HANDLE hVPortHdl,  
HI_HANDLE hVencHdl)
```

【参数】

参数名称	描述	输入/输出
hVProcHdl	VPORC 通道号。 取值范围：[0, 32)。	输入
hVPortHdl	VPORT 通道号 取值范围：[0, 12)。	输入
hVencHdl	VENC 通道号。 取值范围：[0, HI_VENC_MAX_CHN_NUM)。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】

- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a

【注意】

无。

【举例】



无

【相关主题】

[HI\\_MAPI\\_VEnc\\_Bind\\_VProc](#)

## HI\_MAPI\_VEnc\_SetAttr

【描述】

设置编码通道属性参数。

【语法】

```
HI_S32 HI_MAPI_VEnc_SetAttr(HI_HANDLE hVencHdl, HI\_MPP\_VENC\_ATTR\_S
*pstStreamAttr)
```

【参数】

参数名称	描述	输入/输出
hVencHdl	VENC 通道号。 取值范围：[0, HI_VENC_MAX_CHN_NUM)。	输入
pstStreamAttr	编码通道属性参数指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】

- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a

【注意】

- 通道创建后才可以设置通道属性参数。
- 通道属性参数，仅可设置动态参数，使用时请注意查看参数结构体说明。

【举例】

无。



【相关主题】

- [HI\\_MAPI\\_VEnc\\_GetAttr](#)
- [HI\\_MPP\\_VENC\\_ATTR\\_S](#)

## HI\_MAPI\_VEnc\_GetAttr

【描述】

获取编码通道属性参数。

【语法】

```
HI_S32 HI_MAPI_VEnc_GetAttr(HI_HANDLE hVencHdl, HI\_MPP\_VENC\_ATTR\_S  
*pstStreamAttr);
```

【参数】

参数名称	描述	输入/输出
hVencHdl	VENC 通道号。 取值范围：[0, HI_VENC_MAX_CHN_NUM)。	输入
pstStreamAttr	编码通道属性参数指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】

- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a

【注意】

无。

【举例】

无。

【相关主题】

[HI\\_MAPI\\_VEnc\\_SetAttr](#)



## HI\_MAPI\_VEnc\_RequestIDR

### 【描述】

请求 IDR 帧。

### 【语法】

```
HI_S32 HI_MAPI_VEnc_RequestIDR(HI_HANDLE hVencHdl)
```

### 【参数】

参数名称	描述	输入/输出
hVencHdl	VENC 通道号。 取值范围：[0, HI_VENC_MAX_CHN_NUM)。	输入

### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

### 【芯片差异】

无。

### 【需求】

- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a

### 【注意】

编码启动后，调用该接口，请求 IDR 帧。

### 【举例】

无。

### 【相关主题】

- [HI\\_MAPI\\_VEnc\\_Start](#)
- [HI\\_MAPI\\_VEnc\\_Stop](#)

## HI\_MAPI\_VEnc\_GetStreamHeadInfo

### 【描述】

获取码流头信息。



### 【语法】

```
HI_S32 HI_MAPI_VEnc_GetStreamHeadInfo(HI_HANDLE hVencHdl,  
HI_VENC_HEAD_INFO_TYPE_E enType, HI_CHAR *pcHeadInfo, HI_U32  
*pu32HeadInfoLength)
```

### 【参数】

参数名称	描述	输入/输出
hVencHdl	VENC 通道号。 取值范围: [0, HI_VENC_MAX_CHN_NUM)。	输入
enType	码流头信息类型 HI_VENC_HEAD_INFO_TYPE_E	输入
pcHeadInfo	头信息指针	输出
pu32HeadInfoLength	头信息长度指针	输入/输出

### 【返回值】

返回值	描述
0	成功。
非 0	失败, 其值为错误码。

### 【芯片差异】

无。

### 【需求】

- 头文件: hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件: libhi3518e\_mapi\_venc.a

### 【注意】

- H.264 编码时支持获取 PPS、SPS 信息。
- H.265 编码时支持获取 VPS、PPS、SPS 信息。
- pu32HeadInfoLength 指向的值, 需要作为外部字符串数组的长度传入。
- 编码器启动后调用该接口, 如果获取失败返回值是 HI\_ERR\_MAPI\_VENC\_BUF\_EMPTY, 请等待几十毫秒后再次获取。

### 【举例】

无

### 【相关主题】



无

## HI\_MAPI\_VEnc\_SetThumbNailAttr

### 【描述】

设置编码通道缩略图参数。

### 【语法】

```
HI_S32 HI_MAPI_VEnc_SetThumbNailAttr(HI_HANDLE hVencHdl,  
HI_SNAP_SCREENNAIL_ATTR_S *pstThumbNailAttr);
```

### 【参数】

参数名称	描述	输入/输出
hVencHdl	VENC 通道号。 取值范围：[0, HI_VENC_MAX_CHN_NUM)。	输入
pstThumbNailAttr	缩略图参数指针。	输入

### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <b>错误码</b> 。

### 【芯片差异】

无。

### 【需求】

- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a

### 【注意】

- 该接口用于设置大于 160X120 的多级缩略图。
- 如果需要 160X120 的小缩略图，需要将 [HI\\_MPP\\_VENC\\_ATTR\\_JPEG\\_S](#) 中的 bEnableDCF 设置为 HI\_TRUE。

### 【举例】

无。

### 【相关主题】

[HI\\_MAPI\\_VEnc\\_GetThumbNailAttr](#)





## HI\_MAPI\_VEnc\_GetThumbNailAttr

### 【描述】

获取编码通道缩略图参数。

### 【语法】

```
HI_S32 HI_MAPI_VEnc_GetThumbNailAttr(HI_HANDLE hVencHdl,  
HI_SNAP_SCREENNAIL_ATTR_S* pstThumbNailAttr)
```

### 【参数】

参数名称	描述	输入/输出
hVencHdl	VENC 通道号。 取值范围：[0, HI_VENC_MAX_CHN_NUM)。	输入
pstThumbNailAttr	缩略图参数指针	输出

### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

### 【芯片差异】

无。

### 【需求】

- 头文件：hi\_mapi\_venc\_define.h、hi\_mapi\_venc.h
- 库文件：libhi3518e\_mapi\_venc.a

### 【注意】

无

### 【举例】

无。

### 【相关主题】

[HI\\_MAPI\\_VEnc\\_SetThumbNailAttr](#)



## 4.4 数据类型

视频编码相关数据类型定义如下：

- **HI\_VENC\_MAX\_CHN\_NUM**：定义视频编码器的最大个数。
- **HI\_VENC\_PACK\_NUM**：定义视频编码每个包的地址数量。
- **HI\_VENC\_MAX\_FRAME\_PACKCOUNT**：定义每帧数据中数据包的最大个数。
- **HI\_VENC\_LIMITLESS\_FRAME\_COUNT**：定义视频编码无限制数量。
- **VENC\_CHN\_REGISTER\_CALLBACK\_MAX\_COUNT**：定义视频编码每个通道注册的回调函数最大个数。
- **MEDIA\_VENC\_HEAD\_INFO\_BUFF\_LEN**：定义编码头信息 BUFF 最大长度。
- **HI\_MPP\_VENC\_PARAM\_REF\_S**：定义编码器高级跳帧参考结构体。
- **HI\_MPP\_VENC\_ATTR\_H264\_S**：定义 H.264 编码参数。
- **HI\_MPP\_VENC\_ATTR\_H265\_S**：定义 H.265 编码参数。
- **HI\_MPP\_VENC\_ATTR\_JPEG\_S**：定义 JPEG 编码参数。
- **HI\_MPP\_RESOLUTION\_S**：定义分辨率结构体。
- **HI\_MPP\_VENC\_ATTR\_CBR\_S**：定义固定码率参数结构体。
- **HI\_MPP\_VENC\_ATTR\_VBR\_S**：定义可变码率参数结构体。
- **HI\_MPP\_VENC\_ATTR\_S**：定义编码参数结构体。
- **HI\_VENC\_DATA\_H264E\_NALU\_TYPE\_E**：定义 H.264 的 NALU 类型。
- **HI\_VENC\_DATA\_H265E\_NALU\_TYPE\_E**：定义 H.265 的 NALU 类型。
- **HI\_VENC\_DATA\_JPEGE\_PACK\_TYPE\_E**：定义 JPEG 的包类型。
- **HI\_VENC\_DATA\_PACK\_S**：定义视频编码数据包类型。
- **HI\_REFSLICE\_TYPE\_E**：定义编码何种跳帧参考模式下的参考帧。
- **HI\_REF\_TYPE\_E**：定义码流的帧类型。
- **HI\_VENC\_STREAM\_INFO\_S**：定义视频编码码流信息结构体。
- **HI\_VENC\_DATA\_S**：定义视频编码数据结构体。
- **PFN\_VENC\_DataProc**：定义视频编码回调函数数据处理回调函数。
- **HI\_VENC\_CALLBACK\_S**：定义视频回调函数结构体。
- **HI\_VENC\_HEAD\_INFO\_TYPE\_E**：定义视频编码头信息类型。
- **HI\_SNAP\_SCREENNAIL\_ATTR\_S**：定义视频编码缩略图参数。

### HI\_VENC\_MAX\_CHN\_NUM

#### 【说明】

定义编码通道的最大个数。

#### 【定义】

```
#define HI_VENC_MAX_CHN_NUM 16
```



**【注意事项】**

无

**【相关数据类型及接口】**

无

## HI\_VENC\_PACK\_NUM

**【说明】**

定义编码数据包的数量。

**【定义】**

```
#define HI_VENC_PACK_NUM    2
```

**【注意事项】**

无

**【相关数据类型及接口】**

[PFN\\_VENC\\_DataProc](#)

## HI\_VENC\_MAX\_FRAME\_PACKCOUNT

**【说明】**

定义每帧数据中包的最大个数。

**【定义】**

```
#define HI_VENC_MAX_FRAME_PACKCOUNT  12
```

**【注意事项】**

无

**【相关数据类型及接口】**

无

## HI\_VENC\_LIMITLESS\_FRAME\_COUNT

**【说明】**

定义视频编码不限制数量。

**【定义】**

```
#define HI_VENC_LIMITLESS_FRAME_COUNT  -1
```

**【注意事项】**

无。



【相关数据类型及接口】

[HI\\_MAPI\\_VEnc\\_Start](#)

## VENC\_CHN\_REGISTER\_CALLBACK\_MAX\_COUNT

【说明】

定义一个视频编码通道下，可以注册的回调函数最大个数。

【定义】

```
#define VENC_CHN_REGISTER_CALLBACK_MAX_COUNT 5
```

【注意事项】

无。

【相关数据类型及接口】

- [HI\\_MAPI\\_VEnc\\_RegisterCallback](#)
- [HI\\_MAPI\\_VEnc\\_UnRegisterCallback](#)

## MEDIA\_VENC\_HEAD\_INFO\_BUFF\_LEN

【说明】

定义编码头信息 BUFF 的最大长度。

【定义】

```
#define MEDIA_VENC_HEAD_INFO_BUFF_LEN 128
```

【注意事项】

无。

【相关数据类型及接口】

[HI\\_MAPI\\_VEnc\\_GetStreamHeadInfo](#)

## HI\_MPP\_VENC\_PARAM\_REF\_S

【说明】

定义视频编码 H.264/H.265 的高级跳帧参考参数。

【定义】

```
typedef struct hiMPP_VENC_PARAM_REF_S
{
    HI_U32 u32Base;           /**< (0, +∞)*/
    HI_U32 u32Enhance;        /**< [0, 255]*/
    HI_BOOL bEnablePred;
} HI_MPP_VENC_PARAM_REF_S;
```



#### 【成员】

成员名称	描述
u32Base	base 层的周期。 取值范围：(0, +∞)。
u32Enhance	enhance 层的周期。 取值范围：[0, 255]。
bEnablePred	代表 base 层的帧是否被 base 层其他帧用作参考。当为 HI_FALSE 时，base 层的所有帧都参考 IDR 帧。

#### 【注意事项】

无

#### 【相关数据类型及接口】

- [HI\\_MPP\\_VENC\\_ATTR\\_H264\\_S](#)
- [HI\\_MPP\\_VENC\\_ATTR\\_H265\\_S](#)

## HI\_MPP\_VENC\_ATTR\_H264\_S

#### 【说明】

定义 H.264 编码参数。

#### 【定义】

```
typedef struct hiMPP_VENC_ATTR_H264_S
{
    HI_U32 u32Profile; /**< 0: baseline; 1:MP; 2:HP; 3: SVC-T [0,3]; */
    HI\_MPP\_VENC\_PARAM\_REF\_S stParamRef;
} HI_MPP_VENC_ATTR_H264_S;;
```

#### 【成员】

成员名称	描述
u32Profile	编码的等级。 取值范围：[0, 2]。 0: Baseline。 1: Main Profile。 2: High Profile。 静态属性。



成员名称	描述
stParamRef	高级跳帧参考参数。 静态属性。

【注意事项】

无。

【相关数据类型及接口】

- [HI\\_MPP\\_VENC\\_PARAM\\_REF\\_S](#)
- [HI\\_MAPI\\_VEnc\\_Init](#)

## HI\_MPP\_VENC\_ATTR\_H265\_S

【说明】

定义 H.265 编码参数。

【定义】

```
typedef struct hiMPP_VENC_ATTR_H265_S
{
    HI_U32  u32Profile;                /**< 0: MP */
    HI\_MPP\_VENC\_PARAM\_REF\_S stParamRef;
} HI_MPP_VENC_ATTR_H265_S;
```

【成员】

成员名称	描述
u32Profile	编码等级。 0: Main Profile。 静态属性。
stParamRef	高级跳帧参考参数。 静态属性。

【注意事项】

无。

【相关数据类型及接口】

- [HI\\_MPP\\_VENC\\_PARAM\\_REF\\_S](#)
- [HI\\_MAPI\\_VEnc\\_Init](#)



## HI\_MPP\_VENC\_ATTR\_JPEG\_S

### 【说明】

定义 JPEG 编码参数。

### 【定义】

```
typedef struct hiMPP_VENC_ATTR_JPEG_S
{
    HI_BOOL bEnableDCF;
    HI_U32 u32Qfactor;    /**< [1, 99]*/
} HI_MPP_VENC_ATTR_JPEG_S;
```

### 【成员】

成员名称	描述
bEnableDCF	是否支持 Jpeg 的缩略图。 静态属性。
u32Qfactor	图像质量。 取值范围[1, 99]。 静态属性。

### 【注意事项】

无。

### 【相关数据类型及接口】

[HI\\_MAPI\\_VEnc\\_Init](#)

## HI\_MPP\_RESOLUTION\_S

### 【说明】

定义分辨率结构体。

### 【定义】

```
typedef struct hiMPP_RESOLUTION_S
{
    HI_U32 u32Width;
    HI_U32 u32Height;
} HI_MPP_RESOLUTION_S;
```

### 【成员】



成员名称	描述
u32Width	宽度。 取值范围：[MIN_WIDTH, MAX_WIDTH]，以像素为单位。 必须是 MIN_ALIGN 的整数倍。 静态属性。
u32Height	高度。 取值范围：[MIN_HEIGHT, MAX_HEIGHT]，以像素为单位。 必须是 MIN_ALIGN 的整数倍。 静态属性。

【芯片差异】

无

【注意事项】

无

【相关数据类型及接口】

[HI\\_VENC\\_TYPE\\_ATTR\\_S](#)

## HI\_MPP\_VENC\_ATTR\_CBR\_S

【说明】

定义编码通道固定码率模式参数。

【定义】

```
typedef struct hiMPP_VENC_ATTR_CBR_S
{
    HI_U32    u32Gop;           /**< I frame interval [1, 65536]*/
    HI_U32    u32FrameRate;     /**< frame rate [1, 240]*/
    HI_U32    u32BitRate;       /**< encode bit rate, with Kbps as a
unit*/
} HI_MPP_VENC_ATTR_CBR_S;
```

【成员】

成员名称	描述
u32Gop	GOP 大小，取值范围：[1, 65536]。
u32FrameRate	帧率大小。取值范围：[1, 240]。
u32BitRate	平均 bitrate，以 kbps 为单位。 取值范围：[2, 102400]





#### 【注意事项】

无。

#### 【相关数据类型及接口】

[HI\\_MPP\\_VENC\\_RC\\_ATTR\\_S](#)

## HI\_MPP\_VENC\_ATTR\_VBR\_S

#### 【说明】

定义视频编码可变码率模式参数。

#### 【定义】

```
typedef struct hiMPP_VENC_ATTR_VBR_S
{
    HI_U32    u32Gop;           /**< I frame interval [1, 65536]*/
    HI_U32    u32FrameRate;     /**< frame rate [1, 240]*/
    HI_U32    u32MaxBitRate;    /**< encode max bit rate, with Kbps as a
unit*/
} HI_MPP_VENC_ATTR_VBR_S;
```

#### 【成员】

成员名称	描述
u32Gop	GOP 大小，取值范围：[1, 65536]。
u32FrameRate	帧率大小。取值范围：[1, 240]。
u32MaxBitRate	最大 bitrate，以 kbps 为单位。 取值范围：[2, 102400]

#### 【注意事项】

无

#### 【相关数据类型及接口】

[HI\\_MPP\\_VENC\\_RC\\_ATTR\\_S](#)

## HI\_MPP\_VENC\_ATTR\_S

#### 【说明】

定义视频编码参数。

#### 【定义】

```
typedef struct hiMPP_VENC_ATTR_S
```



```
{
    HI_VENC_TYPE_ATTR_S    stVencPloadTypeAttr;
    HI_MPP_VENC_RC_ATTR_S  stRcAttr;
} HI_MPP_VENC_ATTR_S;
```

#### 【成员】

成员名称	描述
stVencPloadTypeAttr	编码类型参数，动态属性。
stRcAttr	码率控制参数，动态属性。

#### 【注意事项】

无。

#### 【相关数据类型及接口】

- [HI\\_VENC\\_TYPE\\_ATTR\\_S](#)
- [HI\\_MPP\\_VENC\\_RC\\_ATTR\\_S](#)

## HI\_VENC\_DATA\_H264E\_NALU\_TYPE\_E

#### 【说明】

定义 H.264 码流 NALU 类型。

#### 【定义】

```
typedef enum hiVENC_DATA_H264E_NALU_TYPE_E
{
    HI_ENC_H264E_NALU_BSLICE = 0,          /**<BSLICE types*/
    HI_ENC_H264E_NALU_PSLICE = 1,          /**<PSLICE types*/
    HI_ENC_H264E_NALU_ISLICE = 2,          /**<ISLICE types*/
    HI_ENC_H264E_NALU_IDRSLICE = 5,        /**<IDRSLICE types*/
    HI_ENC_H264E_NALU_SEI    = 6,          /**<SEI types*/
    HI_ENC_H264E_NALU_SPS    = 7,          /**<SPS types*/
    HI_ENC_H264E_NALU_PPS    = 8,          /**<PPS types*/
    HI_ENC_H264E_NALU_BUTT
} HI_VENC_DATA_H264E_NALU_TYPE_E;
```

#### 【成员】

成员名称	描述
HI_ENC_H264E_NALU_BSLICE	BSLICE 类型。
HI_ENC_H264E_NALU_PSLICE	PSLICE 类型。



成员名称	描述
HI_ENC_H264E_NALU_ISLICE	ISLICE 类型。帧类型为 P 帧。
HI_ENC_H264E_NALU_IDRSLICE	ISLICE 类型。帧类型为 IDR 帧。
HI_ENC_H264E_NALU_SEI	SEI 类型。
HI_ENC_H264E_NALU_SPS	SPS 类型。
HI_ENC_H264E_NALU_PPS	PPS 类型。

【注意事项】

无

【相关数据类型及接口】

无

## HI\_VENC\_DATA\_H265E\_NALU\_TYPE\_E

【说明】

定义 H.265 码流 NALU 类型。

【定义】

```
typedef enum hiVENC_H265E_NALU_TYPE_E
{
    HI_ENC_H265E_NALU_BSLICE = 0,          /**<B SLICE types*/
    HI_ENC_H265E_NALU_PSLICE = 1,          /**<P SLICE types*/
    HI_ENC_H265E_NALU_ISLICE = 2,          /**<I SLICE types*/
    HI_ENC_H265E_NALU_IDRSLICE = 19,       /**<IDR SLICE types*/
    HI_ENC_H265E_NALU_VPS = 32,            /**<VPS types*/
    HI_ENC_H265E_NALU_SPS = 33,            /**<SPS types*/
    HI_ENC_H265E_NALU_PPS = 34,            /**<PPS types*/
    HI_ENC_H265E_NALU_SEI = 39,            /**<SEI types*/
    HI_ENC_H265E_NALU_BUTT
} HI_VENC_DATA_H265E_NALU_TYPE_E;
```

【成员】

成员名称	描述
HI_ENC_H265E_NALU_BSLICE	BSLICE 类型。
HI_ENC_H265E_NALU_PSLICE	PSLICE 类型。
HI_ENC_H266E_NALU_ISLICE	ISLICE 类型。帧类型为 P 帧。
HI_ENC_H266E_NALU_IDRSLICE	ISLICE 类型。帧类型为 IDR 帧。



成员名称	描述
HI_ENC_H265E_NALU_VPS	VPS 类型。
HI_ENC_H265E_NALU_SPS	SPS 类型。
HI_ENC_H265E_NALU_PPS	PPS 类型。
HI_ENC_H265E_NALU_SEI	SEI 类型。

【注意事项】

无

【相关数据类型及接口】

无

## HI\_VENC\_DATA\_JPEGE\_PACK\_TYPE\_E

【说明】

VI 通道信息结构体。

【定义】

```
typedef enum hiVENC_DATA_JPEG_PACK_TYPE_E
{
    HI_VENC_JPEGE_PACK_ECS = 5,
    HI_VENC_JPEGE_PACK_APP = 6,
    HI_VENC_JPEGE_PACK_VDO = 7,
    HI_VENC_JPEGE_PACK_PIC = 8,
    HI_VENC_JPEGE_PACK_BUTT
}HI_VENC_DATA_JPEGE_PACK_TYPE_E;
```

【成员】

成员名称	描述
HI_VENC_JPEGE_PACK_ECS	ECS 类型。
HI_VENC_JPEGE_PACK_APP	APP 类型。
HI_VENC_JPEGE_PACK_VDO	VDO 类型。
HI_VENC_JPEGE_PACK_PIC	PIC 类型。

【注意事项】

无



【相关数据类型及接口】

无

## HI\_VENC\_DATA\_TYPE\_S

## HI\_VENC\_DATA\_PACK\_S

【说明】

编码后的数据包类型。

【定义】

```
typedef struct hiVENC_DATA_PACK_S
{
    HI_U32 u32PhyAddr[HI_VENC_PACK_NUM];/**< the physics address of
stream*/
    HI_U8 *pu8Addr[HI_VENC_PACK_NUM];/**< the virtual address of stream*/
    HI_U32 au32Len[HI_VENC_PACK_NUM];
    HI_U64 u64PTS;
    HI_VENC_DATA_TYPE_S stDataType;
    HI_U32 u32Offset;
} HI_VENC_DATA_PACK_S;
```

【成员】

成员名称	描述
u32PhyAddr	数据包的物理地址。
pu8Addr	数据包的虚拟地址。
au32Len	数据包的长度。
u64PTS	时间戳。
stDataType	数据类型。
u32Offset	数据偏移。

【注意事项】

u32PhyAddr 和 pu8Addr 都有两个地址，地址 0 是有效的，地址 1 作为保留。

【相关数据类型及接口】

[HI\\_VENC\\_DATA\\_S](#)

## HI\_REFSLICE\_TYPE\_E

【说明】



定义编码何种跳帧参考模式下的参考帧。

#### 【定义】

```
typedef enum hiREFSLICE_TYPE_E
{
    HI_REFSLICE_FOR_1X = 1,
    HI_REFSLICE_FOR_2X = 2,
    HI_REFSLICE_FOR_4X = 5,
    HI_REFSLICE_FOR_BUTT
} HI_REFSLICE_TYPE_E;
```

#### 【成员】

成员名称	描述
HI_REFSLICE_FOR_1X	1 倍跳帧参考时的参考帧。
HI_REFSLICE_FOR_2X	2 倍跳帧参考时的参考帧或 4 倍跳帧参考时用于 2 倍跳帧参考的参考帧。
HI_REFSLICE_FOR_4X	4 倍跳帧参考时的参考帧。

#### 【芯片差异】

无

#### 【注意事项】

无

#### 【相关数据类型及接口】

[HI\\_VENC\\_STREAM\\_INFO\\_S](#)

## HI\_REF\_TYPE\_E

#### 【说明】

定义码流的帧类型。

#### 【定义】

```
typedef enum hiREF_TYPE_E
{
    HI_BASE_IDRSLICE = 0,          /**<the Idr frame at Base layer*/
    HI_BASE_PSLICE_REFTOIDR,       /**<the P frame at Base layer,
referenced by other frames at Base layer and reference to Idr frame */
    HI_BASE_PSLICE_REFBYBASE,      /**<the P frame at Base layer,
referenced by other frames at Base layer*/
    HI_BASE_PSLICE_REFBYENHANCE,   /**<the P frame at Base layer,
```



```
referenced by other frames at Enhance layer*/
    HI_ENHANCE_PSLICE_REFBYENHANCE,    /**<the P frame at Enhance layer,
referenced by other frames at Enhance layer*/
    HI_ENHANCE_PSLICE_NOTFORREF,        /**<the P frame at Enhance layer ,not
referenced*/
    HI_ENHANCE_PSLICE_BUTT
} HI_REF_TYPE_E;
```

#### 【成员】

成员名称	描述
HI_BASE_IDRSLICE	base 层中的 IDR 帧。
HI_BASE_PSLICE_REFTOIDR	base 层中的 P 帧，用于 base 层中其他帧的参考且只参考 IDR 帧。
HI_BASE_PSLICE_REFBYBASE	base 层中的 P 帧，用于 base 层中其他帧的参考。
HI_BASE_PSLICE_REFBYENHANCE	base 层中的 P 帧，用于 enhance 层中的帧的参考。
HI_ENHANCE_PSLICE_REFBYENHANCE	enhance 层中的 P 帧，用于 enhance 层中其他帧的参考。
HI_ENHANCE_PSLICE_NOTFORREF	enhance 层中的 P 帧，不用于参考。

#### 【芯片差异】

无

#### 【注意事项】

无

#### 【相关数据类型及接口】

[HI\\_VENC\\_STREAM\\_INFO\\_S](#)

## HI\_VENC\_STREAM\_INFO\_S

#### 【说明】

定义编码码流信息。

#### 【定义】

```
typedef struct hiVENC_STREAM_INFO_S
{
    HI_REFSLICE_TYPE_E enRefSliceType;
    HI_REF_TYPE_E enRefType;
```



```
HI_U32 u32StartQp;  
}HI_VENC_STREAM_INFO_S;
```

#### 【成员】

成员名称	描述
enRefSliceType	跳帧参考模式类型。
enRefType	帧类型以及参考属性。
u32StartQp	编码当前帧的 startqp 值。

#### 【芯片差异】

无

#### 【注意事项】

无

#### 【相关数据类型及接口】

无

## HI\_VENC\_DATA\_S

#### 【说明】

定义编码后的数据结构体。

#### 【定义】

```
typedef struct hiVENC_DATA_S  
{  
    HI_VENC_DATA_PACK_S astPack[HI_VENC_MAX_FRAME_PACKCOUNT];  
    /**<stream pack attribute*/  
    HI_U32 u32PackCount;    /**<the pack number of one frame stream*/  
    HI_U32 u32Seq;          /**<the list number of stream*/  
    HI_BOOL bEndOfStream;  
    HI_VENC_STREAM_INFO_S stStreamInfo;  
} HI_VENC_DATA_S;
```

#### 【成员】

成员名称	描述
astPack	数据包。
u32PackCount	包的数量。
u32Seq	序列号。





bEndOfStream	是否最后一帧数据包。用于表示拍照最后一帧数据。
stStreamInfo	码流信息。

【注意事项】

无。

【相关数据类型及接口】

- [PFN\\_VENC\\_DataProc](#)
- [HI\\_MAPI\\_VEnc\\_Start](#)

## PFN\_VENC\_DataProc

【说明】

定义编码数据回调函数。

【定义】

```
typedef HI_S32 (*PFN_VENC_DataProc)(HI_HANDLE VencHdl, HI_VENC_DATA_S*  
pVStreamData, HI_VOID *pPrivateData);
```

【成员】

成员名称	描述
VencHdl	编码通道句柄。
pVStreamData	数据指针。
pPrivateData	私有数据指针。

【注意事项】

无

【相关数据类型及接口】

- [HI\\_MAPI\\_VEnc\\_RegisterCallback](#)
- [HI\\_MAPI\\_VEnc\\_UnRegisterCallback](#)

## HI\_VENC\_CALLBACK\_S

【说明】

编码回调函数结构体。

【定义】

```
typedef struct hiVENC_CALLBACK_S  
{
```



```
    PFN_VENC_DataProc pfnDataCB;  
    HI_VOID *pPrivateData;  
} HI_VENC_CALLBACK_S;
```

#### 【成员】

成员名称	描述
pfnDataCB	回调处理函数。用于获取编码数据。
pPrivateData	私有数据指针。作为参数，在 <a href="#">PFN_VENC_DataProc</a> 中被调用。

#### 【注意事项】

- 该结构体被注册后，编码启动后，有编码数据时，pfnDataCB 函数会被调用。用户通过该函数获取编码数据。
- pPrivateData 为私有数据，用户可选用。

#### 【相关数据类型及接口】

- [HI\\_MAPI\\_VEnc\\_RegisterCallback](#)
- [HI\\_MAPI\\_VEnc\\_UnRegisterCallback](#)
- [PFN\\_VENC\\_DataProc](#)

## HI\_VENC\_HEAD\_INFO\_TYPE\_E

#### 【说明】

定义码流头信息类型。

#### 【定义】

```
typedef enum hiVENC_HEAD_INFO_TYPE_E  
{  
    HI_VENC_HEAD_INFO_TYPE_VPS,                /**<VPS types*/  
    HI_VENC_HEAD_INFO_TYPE_PPS,                /**<PPS types*/  
    HI_VENC_HEAD_INFO_TYPE_SPS,                /**<SPS types*/  
    HI_VENC_HEAD_INFO_TYPE_BUTT  
} HI_VENC_HEAD_INFO_TYPE_E;
```

#### 【成员】

成员名称	描述
HI_VENC_HEAD_INFO_TYPE_VPS	VPS 类型。
HI_VENC_HEAD_INFO_TYPE_PPS	PPS 类型。
HI_VENC_HEAD_INFO_TYPE_SPS	SPS 类型。



【注意事项】

无。

【相关数据类型及接口】

[HI\\_MAPI\\_VEnc\\_GetStreamHeadInfo](#)

## HI\_SNAP\_SCREENNAIL\_ATTR\_S

【说明】

定义缩略图参数。

【定义】

```
typedef struct hiSNAP_SCREENNAIL_ATTR_S
{
    HI_U32 u8ScreenNailNum; /**< Default:0, Range:[0,2]; 0: not support
screen nail*/
    HI\_MPP\_RESOLUTION\_S astScreenNailResolution[MAX_SCREEN_NAIL_COUNT];
    /**< The resolution of screen nail*/
}HI_SNAP_SCREENNAIL_ATTR_S;
```

【成员】

成员名称	描述
u8ScreenNailNum	缩略图个数。取值范围[0, 2]。
astScreenNailResolution	缩略图的分辨率。大于 160X120，小于等于原图分辨率。

【注意事项】

u8ScreenNailNum 这里的个数指分辨率大于 120\*160 的缩略图的数量。

【相关数据类型及接口】

- [HI\\_MAPI\\_VEnc\\_SetThumbNailAttr](#)
- [HI\\_MAPI\\_VEnc\\_GetThumbNailAttr](#)

## 4.5 错误码

视频输入 API 错误码如[表 4-1](#) 所示。



表4-1 视频编码 API 错误码

错误代码	宏定义	描述
0xA3038002	HI_ERR_MAPI_VENC_HANDLE_ILLEGAL	视频编码通道句柄无效
0xA3038003	HI_ERR_MAPI_VENC_ILLEGAL_PARAM	视频编码参数错误
0xA3038005	HI_ERR_MAPI_VENC_UNEXIST	视频编码不存在
0xA3038006	HI_ERR_MAPI_VENC_NULL_PTR	视频编码参数空指针错误
0xA3038009	HI_ERR_MAPI_VENC_NOT_PERM	操作不支持，或者试图修改静态参数
0xA303800C	HI_ERR_MAPI_VENC_NOMEM	内存申请失败
0xA303800E	HI_ERR_MAPI_VENC_BUF_EMPTY	缓冲区没数据，或者数据为空
0xA30380A0	HI_ERR_MAPI_VENC_NOT_INITED	编码通道没有初始化
0xA30380A2	HI_ERR_MAPI_VENC_BUSY	编码通道忙



## 目 录

<b>5 系统控制.....</b>	<b>5-1</b>
5.1 概述.....	5-1
5.2 功能描述.....	5-1
5.2.1 内存管理 .....	5-1
5.3 API 参考 .....	5-1
5.4 数据类型.....	5-5
5.4.1 基本数据类型.....	5-5
5.4.2 视频公共类型.....	5-9
5.5 错误码.....	5-10



## 表格目录

表 5-1 系统控制 API 错误码 .....	5-10
--------------------------	------



# 5 系统控制

## 5.1 概述

系统控制主要处理系统媒体相关各模块的初始化和去初始化工作。

- 应用程序启动 MPP（Media Process Platform 媒体处理平台）业务前，必须完成 MPP 系统初始化工作。
- 同理，应用程序退出 MPP 业务后，也要完成 MPP 系统去初始化工作，释放资源。

## 5.2 功能描述

### 5.2.1 内存管理

整个系统的内存分为 OS 管理的内存和媒体处理使用的内存。媒体处理使用的内存称为 MMZ，里面又包括视频 VB、码流 buffer、其他各功能模块占用的内存等。

视频 VB（video buffer）是媒体通路处理过程中，需要缓冲在内存中的一段数据，一般以帧大小为单位，可以通过参数配置文件

mapi/code/mediaserver/configs/sensor/hixxxx/xxx/sensor\_interface\_cfg\_params.c，修改系统需要的 VB 个数和大小。

## 5.3 API 参考

该功能模块提供以下 MPI：

- [HI\\_MAPI\\_Sys\\_Init](#)：初始化系统资源。
- [HI\\_MAPI\\_Sys\\_DeInit](#)：去初始化系统资源。
- [HI\\_MAPI\\_Media\\_Init](#)：初始化媒体相关资源。
- [HI\\_MAPI\\_Media\\_DeInit](#)：去初始化媒体相关资源。



## HI\_MAPI\_Sys\_Init

### 【描述】

初始化系统资源，建立多 CPU 间消息通信管道。为了建立多个 CPU 之间的连接，每个 OS 上运行的业务在初始化过程中需要先调用这个接口建立连接，然后才能做多 CPU 间的通信。默认 Huawei LiteOS 端已经在 app\_init.c 文件中调用了这个函数，用户不需要再显式调用。

### 【语法】

```
HI_S32 HI_MAPI_Sys_Init();
```

### 【参数】

无。

### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值参见 <a href="#">错误码</a> 。

### 【需求】

- 头文件：hi\_mapi\_sys.h
- 库文件：libhi3518e\_mapi\_sys.a

### 【注意】

只有在 MPP 整个系统处于未初始化状态，才可调用此函数配置 MPP 系统，否则会配置失败。

### 【举例】

无。

### 【相关主题】

无。

## HI\_MAPI\_Sys\_DeInit

### 【描述】

去初始化系统资源，断开多 CPU 间通信管道。请明确需要断开各 CPU 间的通信时，再调用该接口。

### 【语法】

```
HI_S32 HI_MAPI_Sys_DeInit();
```

### 【参数】





无。

【返回值】

返回值	描述
0	成功。
非 0	失败，其值参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_sys.h
- 库文件：libhi3518e\_mapi\_sys.a

【注意】

- 调用之前要保证系统已经初始化。
- 仅在确认需要断开多 CPU 间通信管道的情况下，调用该接口。

【举例】

无

【相关主题】

无

## HI\_MAPI\_Media\_Init

【描述】

初始化媒体相关各模块的资源。读取 sensor，VB 的相关配置文件，然后做媒体业务的初始化工作。配置文件存放在 mapi/code/mediaserver/configs 目录下面。

【语法】

```
HI_S32 HI_MAPI_Media_Init();
```

【参数】

无。

【返回值】

返回值	描述
0	成功。
非 0	失败，其值参见 <a href="#">错误码</a> 。

【需求】



- 头文件: hi\_mapi\_sys.h
- 库文件: libhi3518e\_mapi\_sys.a

**【注意】**

无

**【举例】**

无

**【相关主题】**

无

## HI\_MAPI\_Media\_DeInit

**【描述】**

去初始化媒体相关各模块的资源。

**【语法】**

```
HI_S32 HI_MAPI_Media_DeInit();
```

**【参数】**

无。

**【返回值】**

返回值	描述
0	成功。
非 0	失败，其值参见 <a href="#">错误码</a> 。

**【需求】**

- 头文件: hi\_mapi\_sys.h
- 库文件: libhi3518e\_mapi\_sys.a

**【注意】**

无

**【举例】**

无

**【相关主题】**

无



## 5.4 数据类型

### 5.4.1 基本数据类型

基本数据类型定义如下：

#### 公共数据类型

```
typedef unsigned char      HI_U8;
typedef unsigned short     HI_U16;
typedef unsigned int       HI_U32;

typedef signed char        HI_S8;
typedef short              HI_S16;
typedef int                HI_S32;

#ifndef _M_IX86
    typedef unsigned long long HI_U64;
    typedef long long          HI_S64;
#else
    typedef __int64            HI_U64;
    typedef __int64            HI_S64;
#endif

typedef char                HI_CHAR;
#define HI_VOID              void

/*-----*
 * const defination          *
 *-----*/
typedef enum {
    HI_FALSE = 0,
    HI_TRUE  = 1,
} HI_BOOL;

#ifndef NULL
    #define NULL    0L
#endif

#define HI_NULL      0L
#define HI_SUCCESS    0
#define HI_FAILURE    (-1)
```



## HI\_MPP\_RESOLUTION\_S

### 【说明】

定义图像分辨率宽高的结构体。

### 【定义】

```
typedef struct hiMPP_RESOLUTION_S
{
    HI_U32 u32Width;
    HI_U32 u32Height;
} HI_MPP_RESOLUTION_S;
```

### 【成员】

成员名称	描述
u32Width	分辨率的宽。
u32Height	分辨率的高。

### 【注意事项】

无。

### 【相关数据类型及接口】

无。

## HI\_MPP\_RECT\_S

### 【说明】

定义矩形区域信息结构体。

### 【定义】

```
typedef struct hiMPP_RECT_S
{
    HI_U32 u32X;
    HI_U32 u32Y;
    HI_U32 u32Width;
    HI_U32 u32Height;
} HI_MPP_RECT_S;
```

### 【成员】

成员名称	描述
s32X	横坐标。



成员名称	描述
s32Y	纵坐标。
u32Width	宽度。
u32Height	高度。

**【注意事项】**

无。

**【相关数据类型及接口】**

无。

## HI\_MPP\_MOD\_ID\_E

**【说明】**

定义模块 ID 枚举类型。

**【定义】**

```
typedef enum hiMPP_MOD_ID_E
{
    HI_MPP_MOD_SYS = 0,
    HI_MPP_MOD_VCAP,
    HI_MPP_MOD_VPROC,
    HI_MPP_MOD_VENC,
    HI_MPP_MOD_ACAP,
    HI_MPP_MOD_AENC,
    HI_MPP_MOD_ADEC,
    HI_MPP_MOD_AO,
    HI_MPP_MOD_DISP,
    HI_MPP_MOD_VDEC,
    HI_MPP_MOD_BUTT,
} HI_MPP_MOD_ID_E;
```

**【成员】**

无。

**【注意事项】**

无。

**【相关数据类型及接口】**

无。



## HI\_MPP\_PIXEL\_FORMAT\_E

### 【说明】

定义像素格式的枚举。

### 【定义】

```
typedef enum hiMPP_PIXEL_FORMAT_E
{
    HI_MPP_PIXEL_FORMAT_420 = 0,
    HI_MPP_PIXEL_FORMAT_422,
    HI_MPP_PIXEL_FORMAT_RGB_1555,
    HI_MPP_PIXEL_FORMAT_RGB_8888,
    HI_MPP_PIXEL_FORMAT_RGB_BAYER_8BPP,
    HI_MPP_PIXEL_FORMAT_RGB_BAYER_10BPP,
    HI_MPP_PIXEL_FORMAT_RGB_BAYER_12BPP,
    HI_MPP_PIXEL_FORMAT_RGB_BAYER_14BPP,
    HI_MPP_PIXEL_FORMAT_RGB_BAYER,
    HI_MPP_PIXEL_FORMAT_BUTT
} HI_MPP_PIXEL_FORMAT_E;
```

### 【成员】

无。

### 【注意事项】

无。

### 【相关数据类型及接口】

无。

## HI\_MPP\_PAYLOAD\_TYPE\_E

### 【说明】

定义音视频编码协议类型枚举。

### 【定义】

```
typedef enum hiMPP_PAYLOAD_TYPE_E
{
    HI_MPP_PAYLOAD_TYPE_H264,
    HI_MPP_PAYLOAD_TYPE_H265,
    HI_MPP_PAYLOAD_TYPE_MJPEG,
    HI_MPP_PAYLOAD_TYPE_JPEG,
    HI_MPP_PAYLOAD_TYPE_AAC,
    HI_MPP_PAYLOAD_TYPE_BUTT
} HI_MPP_PAYLOAD_TYPE_E;
```



**【成员】**

无。

**【注意事项】**

无。

**【相关数据类型及接口】**

无。

## HI\_FRAME\_DATA\_TYPE\_E

**【说明】**

定义帧数据类型枚举。

**【定义】**

```
typedef enum hiFRAME_DATA_TYPE_E
{
    HI_FRAME_DATA_TYPE_RAW,
    HI_FRAME_DATA_TYPE_YUV,
    HI_FRAME_DATA_TYPE_BUTT,
} HI_FRAME_DATA_TYPE_E;
```

**【成员】**

无。

**【注意事项】**

无。

**【相关数据类型及接口】**

无。

## 5.4.2 视频公共类型

视频公共数据类型定义如下：

**HI\_FRAME\_DATA\_S**：定义视频帧信息结构体。

## HI\_FRAME\_DATA\_S

**【说明】**

定义视频帧信息结构体。

**【定义】**

```
typedef struct hiFRAME_DATA_S
{
    HI_FRAME_DATA_TYPE_E enFrameDataType;
```



```
HI_U32          u32Width;  
HI_U32          u32Height;  
HI_MPP_PIXEL_FORMAT_E enPixelFormat;  
HI_U32          u32PhyAddr[3];  
HI_VOID*        pVirAddr[3];  
HI_U32          u32Stride[3];  
HI_U64          u64pts;  
} HI_FRAME_DATA_S;
```

#### 【成员】

成员名称	描述
enFrameDataType	视频帧类型。
u32Width	图像宽。
u32Height	图像高。
enPixelFormat	像素格式。
u32PhyAddr	图像存储在内存中的物理地址。
pVirAddr	图像存储在内存中的虚拟地址。
u32Stride	图像存储在内存中每行的长度。
u64pts	当前帧的时间戳。

#### 【注意事项】

无。

#### 【相关数据类型及接口】

无。

## 5.5 错误码

系统控制 API 错误码如表 5-1 所示。

表5-1 系统控制 API 错误码

错误代码	宏定义	描述
0xA3008003	HI_MAPI_ERR_SYS_ILLEGAL_PARAM	参数设置无效
0xA3008006	HI_MAPI_ERR_SYS_NULL_PTR	空指针错误
0xA3008009	HI_MAPI_ERR_SYS_NOT_PERM	操作不允许





错误代码	宏定义	描述
0xA3008010	HI_MAPI_ERR_SYS_NOTREADY	系统控制属性未配置
0xA3008012	HI_MAPI_ERR_SYS_BUSY	系统忙
0xA300800C	HI_MAPI_ERR_SYS_NOMEM	分配内存失败，如系统内存不足



## 目 录

6 显示.....	6-1
6.1 概述.....	6-1
6.2 重要概念.....	6-1
6.3 API 参考 .....	6-2
6.4 数据类型.....	6-22
6.5 错误码.....	6-32



## 插图目录

图 6-1 视频层属性的相关概念示意.....	6-27
-------------------------	------



## 表格目录

表 6-1 芯片支持的显示设备、图形层和鼠标层情况 .....	6-1
表 6-2 芯片设备规格 .....	6-1
表 6-3 芯片视频层规格 .....	6-2
表 6-4 视频输出 API 错误码 .....	6-32



# 6 显示

## 6.1 概述

DISP 模块即视频显示模块，其封装了 SDK 的 VOU 模块。VOU（Video Output Unit）模块主动从内存相应位置读取视频和图形数据，并通过相应的显示设备输出。芯片支持的显示设备、图形层和鼠标层情况如表 6-1 所示。

表6-1 芯片支持的显示设备、图形层和鼠标层情况

芯片	支持的显示设备		图形层	鼠标层
	HD 设备	SD 设备		
Hi3516C V200	-	支持 1 个标清设备 (DSD)	1 个图形层 (G0)	-
	标清视频层最多支持 8 画面			-

## 6.2 重要概念

- Disp 设备  
Disp 显示设备为 SD 设备，支持 1 个视频层和 1 个图形层。芯片支持显示设备的情况参见表 6-1。芯片的功能参考表 6-2。视频层的功能参考表 6-3 所示。
- 窗口  
显示窗口就是通道，一个显示窗口对应一个画面。NDK 将通道归属于视频层管理。对于一个视频层，它上面的通道都是独立的。

表6-2 芯片设备规格

芯片		最大输出时序	输出接口	叠加拼接显示
Hi3516CV200	SD	1080P@30fps	BT656/LCD	VGS 叠加



表6-3 芯片视频层规格

芯片		动态绑定能力	缩放能力	支持通道数		CSC 调节
				软件支持	硬件支持	
Hi3516CV200	VSD	不支持 (固定绑定在对应的 SD 设备上)	不支持	8	1	支持

- 通道优先级
  - 标清显示设备支持多个通道同时输出显示，按照优先级顺序对输出图像进行叠加，当各个通道的画面有重叠区域时，优先级高的图像显示在上层，如果各个通道优先级一致，则通道号越大的默认优先级越高。
- 分辨率

分辨率主要有以下 3 种概念：

  - 设备分辨率指该设备的输出有效像素点数，由设备时序决定。
  - 显示分辨率指画面在显示设备上的有效显示区域。
  - 图像分辨率指图像本身的有效像素点数。

## 6.3 API 参考

视频输出（DISP）实现启用视频输出设备或窗口、发送视频数据到输出窗口等功能。

该功能模块提供以下 API：

- 设备操作：
  - [HI\\_MAPI\\_Disp\\_Init](#)：初始化视频输出设备。
  - [HI\\_MAPI\\_Disp\\_DeInit](#)：去初始化视频输出设备。
  - [HI\\_MAPI\\_Disp\\_Start](#)：启动视频输出设备。
  - [HI\\_MAPI\\_Disp\\_Stop](#)：停止视频输出设备。
- 窗口操作：
  - [HI\\_MAPI\\_Disp\\_Window\\_SetAttr](#)：设置视频输出窗口属性。
  - [HI\\_MAPI\\_Disp\\_Window\\_GetAttr](#)：获取视频输出窗口属性。
  - [HI\\_MAPI\\_Disp\\_Window\\_Start](#)：启动视频输出窗口。
  - [HI\\_MAPI\\_Disp\\_Window\\_Stop](#)：停止视频输出窗口。
  - [HI\\_MAPI\\_Disp\\_Bind\\_VProc](#)：绑定指定视频输出设备的窗口到指定 Vproc 的 Vport。



- [HI\\_MAPI\\_Disp\\_UnBind\\_VProc](#): 解绑定指定视频输出设备的窗口到指定 Vproc 的 Vport。
- [HI\\_MAPI\\_Disp\\_SendFrame](#): 将视频图像送入指定视频输出窗口显示。
- 高级接口:
  - [HI\\_MAPI\\_Disp\\_SetAttrEx](#): 设置视频输出设备的图像效果。
  - [HI\\_MAPI\\_Disp\\_GetAttrEx](#): 获取视频输出设备的图像效果。
  - [HI\\_MAPI\\_Disp\\_SetReg](#): 设置指定寄存器的值。
  - [HI\\_MAPI\\_Disp\\_GetReg](#): 获取指定寄存器的值。

## HI\_MAPI\_Disp\_Init

### 【描述】

初始化视频输出设备。

### 【语法】

```
HI_S32 HI_MAPI_Disp_Init(HI_HANDLE DispHdl, HI\_MPP\_DISP\_ATTR\_S*  
pstDispAttr);
```

### 【参数】

参数名称	描述	输入/输出
DispHdl	视频输出 handle 号。	输入
pstDispAttr	视频输出设备的属性。	输入

### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

### 【需求】

- 头文件: [hi\\_mapi\\_disp\\_define.h](#)、[hi\\_mapi\\_disp.h](#)
- 库文件: [libhi3518e\\_mapi\\_disp.a](#)

### 【注意】

- 在使用视频输出功能前必须先进行设备初始化操作。
- 视频输出设备属性为静态属性，必须对设备属性 [pstDispAttr](#) 进行赋值。
- 初始化参数一致时，可以重复初始化，直接返回成功；如果参数不同，返回错误。



- DispHdl 的取值范围为[0, VO\_MAX\_DEV\_NUM)。Hi3516 只能为 0。

#### 【举例 1】

```
HI_HANDLE DispHdl = 0;
HI_S32 s32Ret = HI_SUCCESS;
    HI_MPP_DISP_ATTR_S stDispAttr;
    stDispAttr.u32BgColor = 0xFF00;

    stDispAttr.enIntfType = DISP_INTF_BT1120;
    stDispAttr.enIntfSync = DISP_SYNC_1080P30;

    s32Ret = HI_MAPI_Dispatch_Init(DispHdl, &stDispAttr);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Dispatch_Init fail s32Ret:%x\n", s32Ret);
        return HI_FAILURE;
    }

    s32Ret = HI_MAPI_Dispatch_Start(DispHdl);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Dispatch_Start fail s32Ret:%x\n", s32Ret);
        return HI_FAILURE;
    }

    s32Ret = HI_MAPI_Dispatch_Stop(DispHdl);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Dispatch_Stop fail s32Ret:%x\n", s32Ret);
        return HI_FAILURE;
    }

    s32Ret = HI_MAPI_Dispatch_DeInit(DispHdl);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Dispatch_DeInit fail s32Ret:%x\n", s32Ret);
        return s32Ret;
    }
```

#### 【举例 2】

用户自定义 Disp 属性

```
HI_HANDLE s_hVoHdl = 0;
HI_S32 s32Ret = HI_SUCCESS;
HI_MPP_DISP_ATTR_S stCSC = {0};
```





```
stCSC.stVideoCSC.enCscMatrix = HI_MPP_CSC_MATRIX_BT709_TO_RGB_PC;
stCSC.stVideoCSC.u32Contrast = 50;
stCSC.stVideoCSC.u32Hue = 50;
stCSC.stVideoCSC.u32Luma = 50;
stCSC.stVideoCSC.u32Satuature = 50;

HI_MAPI_Disp_SetAttrEx(s_hVoHdl, &stCSC);

vo_attr.u32BgColor = 0x00;
vo_attr.enIntfType = DISP_INTF_LCD_8BIT;
vo_attr.enIntfSync = DISP_SYNC_USER;

vo_attr.stSyncInfo.bSynm=1;
vo_attr.stSyncInfo.bIop=1;
vo_attr.stSyncInfo.u8Intfb=16;

vo_attr.stSyncInfo.u16Vact=272;
vo_attr.stSyncInfo.u16Vbb=8;
vo_attr.stSyncInfo.u16Vfb=8;

vo_attr.stSyncInfo.u16Hact=480;
vo_attr.stSyncInfo.u16Hbb=40;
vo_attr.stSyncInfo.u16Hfb=56;
vo_attr.stSyncInfo.u16Hmid=1;

vo_attr.stSyncInfo.u16Bvact=272;
vo_attr.stSyncInfo.u16Bvbb=8;
vo_attr.stSyncInfo.u16Bvfb=8;

vo_attr.stSyncInfo.u16Hpw=1;
vo_attr.stSyncInfo.u16Vpw=1;

vo_attr.stSyncInfo.bIdv=0;
vo_attr.stSyncInfo.bIhs=1;
vo_attr.stSyncInfo.bIvs=1;

s32Ret = HI_MAPI_Disp_Init(s_hVoHdl, &vo_attr);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_Disp_Init fail, 0x%x\n", s32Ret);
}
```



```
        return HI_FALSE;
    }

    s32Ret = HI_MAPI_Disp_Start(s_hVoHdl);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Disp_Start fail, 0x%x\n", s32Ret);
        return HI_FALSE;
    }
    s32Ret = HI_MAPI_Disp_Stop(DispHdl);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Disp_Stop fail s32Ret:%x\n",s32Ret);
        return HI_FAILURE;
    }

    s32Ret = HI_MAPI_Disp_DeInit(DispHdl);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Disp_DeInit fail s32Ret:%x\n", s32Ret);
        return s32Ret;
    }
}
```

#### 【相关主题】

[HI\\_MAPI\\_Disp\\_DeInit](#)

## HI\_MAPI\_Disp\_DeInit

#### 【描述】

去初始化视频输出设备。

#### 【语法】

```
HI_S32 HI_MAPI_Disp_DeInit(HI_HANDLE DispHdl);
```

#### 【参数】

参数名称	描述	输入/输出
DispHdl	视频输出 handle 号。	输入

#### 【返回值】



返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

#### 【注意】

- 设备去初始化前必须先初始化视频输出设备。
- 调用设备初始化接口后，如果未调用该接口进行去初始化，则 DISP 设备将一直保持使能状态，并且下次以不同参数重新初始化设备时不会生效。

#### 【举例】

请参见 [HI\\_MAPI\\_Disp\\_Init](#) 的举例。

#### 【相关主题】

[HI\\_MAPI\\_Disp\\_Init](#)

## HI\_MAPI\_Disp\_Start

#### 【描述】

启动视频输出设备。

#### 【语法】

```
HI_S32 HI_MAPI_Disp_Start(HI_HANDLE DispHdl);
```

#### 【参数】

参数名称	描述	输入/输出
DispHdl	视频输出 handle 号。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。



#### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

#### 【注意】

- 启动视频输出设备必须在 HI\_MAPI\_Disp\_Init 前配置。
- 参数一致时重复 start 直接返回成功。

#### 【举例】

请参见 [HI\\_MAPI\\_Disp\\_Init](#) 的举例。

#### 【相关主题】

[HI\\_MAPI\\_Disp\\_Stop](#)

## HI\_MAPI\_Disp\_Stop

#### 【描述】

停止视频输出设备。

#### 【语法】

```
HI_S32 HI_MAPI_Disp_Stop(HI_HANDLE DispHdl);
```

#### 【参数】

参数名称	描述	输入/输出
DispHdl	视频输出 handle 号。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

#### 【注意】

- 在停止视频输出设备前，必须先启动视频输出设备。
- 启动视频输出设备后，可以不必停止视频输出设备，再启动视频输出设备，直接返回成功。



#### 【举例】

请参见 [HI\\_MAPI\\_Disp\\_Init](#) 的举例。

#### 【相关主题】

[HI\\_MAPI\\_Disp\\_Start](#)

## HI\_MAPI\_Disp\_Window\_SetAttr

#### 【描述】

配置指定视频输出窗口的属性。

#### 【语法】

```
HI_S32 HI_MAPI_Disp_Window_SetAttr(HI_HANDLE DispHdl, HI_HANDLE WndHdl,  
HI_MPP_DISP_WINDOW_ATTR_S* pstWndAttr);
```

#### 【参数】

参数名称	描述	输入/输出
DispHdl	视频输出 handle 号。	输入
WndHdl	设备窗口 handle 号。	输入
pstWndAttr	设备窗口属性指针。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

#### 【注意】

- 标清设备中，属性中的优先级，数值越大优先级越高。  
优先级有 0~VO\_MAX\_CHN\_NUM-1，当多个通道有重叠的显示区域时，优先级高的通道图像将覆盖优先级低的通道。优先级相同的各通道有重叠时，默认通道号大的图像将覆盖通道号小的通道图像。
- 窗口显示区域不能超过视频输出设备的大小。
- HD 设备的通道属性 stRect 需 2 对齐，且宽高必须不小于 32。在隔行时序且像素格式为 SEMIPLANAR\_420 时，stRect 的高度必须 4 对齐。**起始点的纵坐标建议按**



**照 4 对齐配置，否则可能会出现不可预料的图像质量问题。HD 的宽高必须不小于 32。**

- 如果 HD 有视频层放大的情况，stRect 是放大前视频层上的通道原始的起始位置和宽高，放大后的通道的显示起始位置和宽高会按视频层放大的比例偏移或放大。
- SD 设备的通道属性 stRect 需 2 对齐。

#### 【举例】

请参见 [HI\\_MAPI\\_Disp\\_Window\\_Start](#) 的举例。

#### 【相关主题】

[HI\\_MAPI\\_Disp\\_Window\\_GetAttr](#)

## HI\_MAPI\_Disp\_Window\_GetAttr

#### 【描述】

获取指定视频输出窗口的属性。

#### 【语法】

```
HI_S32 HI_MAPI_Disp_Window_GetAttr(HI_HANDLE DispHdl, HI_HANDLE WndHdl,  
HI_MPP_DISP_WINDOW_ATTR_S* pstWndAttr);
```

#### 【参数】

参数名称	描述	输入/输出
DispHdl	视频输出 handle 号。	输入
WndHdl	设备窗口 handle 号。	输入
pstWndAttr	设备窗口属性指针。	输出

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

#### 【注意】



设备窗口属性指针不能为空。

#### 【举例】

请参见 [HI\\_MAPI\\_Disp\\_Window\\_Start](#) 的举例。

#### 【相关主题】

[HI\\_MAPI\\_Disp\\_Window\\_SetAttr](#)

## HI\_MAPI\_Disp\_Window\_Start

#### 【描述】

启用指定的视频输出窗口。

#### 【语法】

```
HI_S32 HI_MAPI_Disp_Window_Start(HI_HANDLE DispHdl, HI_HANDLE WndHdl);
```

#### 【参数】

参数名称	描述	输入/输出
DispHdl	视频输出 handle 号。	输入
WndHdl	设备窗口 handle 号。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

#### 【注意】

- 调用前必须先启动视频输出设备。
- 窗口启动前必须进行窗口属性配置，否则返回通道未配置的错误。
- 允许重复启动同一视频输出窗口，直接返回成功。

#### 【举例】

```
HI_HANDLE DispHdl = 0;  
HI_S32 s32Ret = HI_SUCCESS;  
HI_MPP_DISP_ATTR_S stDispAttr;
```



```
stDispAttr.u32BgColor = 0xFF00;

stDispAttr.enIntfType = DISP_INTF_BT1120;
stDispAttr.enIntfSync = DISP_SYNC_1080P30;

s32Ret = HI_MAPI_Disp_Init(DispHdl, &stDispAttr);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_Disp_Init fail s32Ret:%x\n",s32Ret);
    return HI_FAILURE;
}

s32Ret = HI_MAPI_Disp_Start(DispHdl);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_Disp_Start fail s32Ret:%x\n",s32Ret);
    return HI_FAILURE;
}

s32Ret = HI_MAPI_Disp_Window_SetAttr(DispHdl, WndHdl, &stWndAttr);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_Disp_Window_SetAttr fail s32Ret:%x\n",s32Ret);
    return HI_FAILURE;
}

s32Ret = HI_MAPI_Disp_Window_Start(DispHdl, WndHdl);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_Disp_Window_Start fail s32Ret:%x\n",s32Ret);
    return HI_FAILURE;
}

s32Ret = HI_MAPI_Disp_Window_Stop(DispHdl, WndHdl);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_Disp_Window_Stop fail s32Ret:%x\n", s32Ret);
    return s32Ret;
}
```

#### 【相关主题】

[HI\\_MAPI\\_Disp\\_Window\\_Stop](#)

## HI\_MAPI\_Disp\_Window\_Stop

#### 【描述】

停止指定的视频输出窗口工作。





#### 【语法】

```
HI_S32 HI_MAPI_Disp_Window_Stop(HI_HANDLE DispHdl, HI_HANDLE WndHdl);
```

#### 【参数】

参数名称	描述	输入/输出
DispHdl	视频输出 handle 号。	输入
WndHdl	设备窗口 handle 号。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

#### 【注意】

当高清设备的窗口绑定 VPROC 的 Vport 时，建议先调用本接口停止窗口后，再解绑定 DISP 窗口与 VPROC 通道的绑定关系，否则，可能出现 HI\_MAPI\_ERR\_DISP\_BUSY 的超时返回错误。

#### 【举例】

请参见 [HI\\_MAPI\\_Disp\\_Window\\_Start](#) 的举例。

#### 【相关主题】

[HI\\_MAPI\\_Disp\\_Window\\_Start](#)

## HI\_MAPI\_Disp\_Bind\_VProc

#### 【描述】

绑定指定视频输出设备的窗口到指定 Vproc 的 Vport。

#### 【语法】

```
HI_S32 HI_MAPI_Disp_Bind_VProc(HI_HANDLE VProcHdl, HI_HANDLE VPortHdl,  
HI_HANDLE DispHdl, HI_HANDLE WndHdl);
```

#### 【参数】



参数名称	描述	输入/输出
VProcHdl	Vproc 的 handle 号。	输入
VPortHdl	Vport 的 handle 号。	输入
DispHdl	视频输出 handle 号。	输入
WndHdl	设备窗口 handle 号。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

#### 【注意】

允许重复绑定，直接返回成功。

#### 【举例】

```
HI_HANDLE DispHdl = 0;
HI_S32 s32Ret = HI_SUCCESS;
    HI_MPP_DISP_ATTR_S stDispAttr;
    stDispAttr.u32BgColor = 0xFF00;

    stDispAttr.enIntfType = DISP_INTF_BT1120;
    stDispAttr.enIntfSync = DISP_SYNC_1080P30;

    s32Ret = HI_MAPI_Dispatch_Init(DispHdl, &stDispAttr);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Dispatch_Init fail s32Ret:%x\n",s32Ret);
        return HI_FAILURE;
    }

    s32Ret = HI_MAPI_Dispatch_Start(DispHdl);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Dispatch_Start fail s32Ret:%x\n",s32Ret);
```



```
        return HI_FAILURE;
    }

    s32Ret = HI_MAPI_Disp_Window_SetAttr(DispHdl, WndHdl, &stWndAttr);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Disp_Window_SetAttr fail s32Ret:%x\n", s32Ret);
        return HI_FAILURE;
    }

    s32Ret = HI_MAPI_Disp_Window_Start(DispHdl, WndHdl);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Disp_Window_Start fail s32Ret:%x\n", s32Ret);
        return HI_FAILURE;
    }

    s32Ret = HI_MAPI_Disp_Bind_VProc(vprocHdl1, vportHdl1, DispHdl,
    WndHdl);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Disp_Bind_VProc fail s32Ret:%x\n", s32Ret);
        return HI_FAILURE;
    }

    s32Ret = HI_MAPI_Disp_Window_Stop(DispHdl, WndHdl);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Disp_Window_Stop fail s32Ret:%x\n", s32Ret);
        return s32Ret;
    }

    s32Ret = HI_MAPI_Disp_UnBind_VProc(vprocHdl1, vportHdl1, DispHdl,
    WndHdl);
    if(HI_SUCCESS != s32Ret)
    {
        printf("HI_MAPI_Disp_UnBind_VProc fail s32Ret:%x\n", s32Ret);
        return s32Ret;
    }
}
```

#### 【相关主题】

[HI\\_MAPI\\_Disp\\_UnBind\\_VProc](#)



## HI\_MAPI\_Disp\_UnBind\_VProc

### 【描述】

解除绑定指定视频输出设备的窗口到指定 Vproc 的 Vport。

### 【语法】

```
HI_S32 HI_MAPI_Disp_UnBind_VProc(HI_HANDLE VProcHdl, HI_HANDLE VPortHdl,  
HI_HANDLE DispHdl, HI_HANDLE WndHdl);
```

### 【参数】

参数名称	描述	输入/输出
VProcHdl	Vproc 的 handle 号。	输入
VPortHdl	Vport 的 handle 号。	输入
DispHdl	视频输出 handle 号。	输入
WndHdl	设备窗口 handle 号。	输入

### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

### 【注意】

允许重复解除绑定，直接返回成功。

### 【举例】

请参见 [HI\\_MAPI\\_Disp\\_Bind\\_VProc](#) 的举例。

### 【相关主题】

[HI\\_MAPI\\_Disp\\_Bind\\_VProc](#)

## HI\_MAPI\_Disp\_SendFrame

### 【描述】

将视频图像送入指定视频输出窗口显示。



### 【语法】

```
HI_S32 HI_MAPI_Dispatch_SendFrame(HI_HANDLE DispHdl, HI_HANDLE WndHdl,  
HI_FRAME_DATA_S *pstFramedata);
```

### 【参数】

参数名称	描述	输入/输出
DispHdl	视频输出 handle 号。	输入
WndHdl	设备窗口 handle 号。	输入
pstFramedata	视频数据信息指针。	输入

### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

### 【注意】

- 调用该接口前必须保证已经启动窗口。
- 该接口默认为非阻塞发送。
- 输入视频数据信息要符合 VO 数据的要求。宽和高需要与实际图像宽高相符，且均不能小于 32，宽高要求以 2 对齐。像素格式为 SPYCbCr420、SPYCbCr422 或者单分量格式。压缩模式为非压缩模式或者是 256 段压缩模式。视频格式只支持 LINEAR 格式。

### 【举例】

无

### 【相关主题】

无

## HI\_MAPI\_Dispatch\_SetAttrEx

### 【描述】

设置视频输出图像效果。



#### 【语法】

```
HI_S32 HI_MAPI_Dispatch_SetAttrEx(HI_HANDLE DispatchHdl, HI_MPP_DISP_ATTREX_S*  
pstDispatchAttrEx);
```

#### 【参数】

参数名称	描述	输入/输出
DispatchHdl	视频输出 handle 号。	输入
pstDispatchAttrEx	disp 结构体属性。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

#### 【注意】

- 该接口可以用于调节图像的输出生效果，包括亮度、对比度、色调、饱和度，其取值范围均为[0, 100]。
- 禁止视频层时会把亮度、对比度、色调、饱和度的值恢复为默认值，CSC 转换矩阵恢复为默认值。
- 由于 LCD 接口输出数据为 RGB 数据而默认输出数据为 YUV，因此当输出接口为 LCD，需要调用该接口，选择 HI\_MPP\_CSC\_MATRIX\_BT601\_TO\_RGB\_PC 或 HI\_MPP\_CSC\_MATRIX\_BT709\_TO\_RGB\_PC 转换矩阵将输出数据转换为 RGB 数据。
- 该接口可以用于调节视频图像的旋转，支持旋转 0 度、90 度、270 度、180 度。

#### 【举例】

无。

#### 【相关主题】

[HI\\_MAPI\\_Dispatch\\_GetAttrEx](#)

## HI\_MAPI\_Dispatch\_GetAttrEx

#### 【描述】



获取设备输出图像效果。

#### 【语法】

```
HI_S32 HI_MAPI_Disp_GetAttrEx(HI_HANDLE DispHdl, HI_MPP_DISP_CMD_E enCMD,  
HI_VOID *pAttr, HI_U32 u32Len);
```

#### 【参数】

参数名称	描述	输入/输出
DispHdl	视频输出 handle 号。	输入
pstDispAttrEx	disp 结构体属性。	输出

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

#### 【注意】

- 该接口可以用于获取图像的输出效果，包括亮度、对比度、色调、饱和度，其取值范围均为[0, 100]。
- 该接口可以用于获取视频图像的旋转属性，即旋转角度。

#### 【举例】

无

#### 【相关主题】

[HI\\_MAPI\\_Disp\\_SetAttrEx](#)

## HI\_MAPI\_Disp\_SetReg

#### 【描述】

设置指定寄存器的值。

#### 【语法】

```
HI_S32 HI_MAPI_Disp_SetReg(HI_U32 u32Addr, HI_U32 u32Value);
```

#### 【参数】



参数名称	描述	输入/输出
u32Addr	寄存器地址	输入
u32Value	指定寄存器的值	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a
- 在用户时序下，可以通过此接口配置屏的时钟频率。

#### 【举例】

```
HI_S32 s32Ret = HI_SUCCESS;
HI_U32 u32VdpCrgAddr = 0x12010044;
HI_U32 u32LcdCrgAddr = 0x12010048;

HI_U32 u32VdpCrgValue = 0x000187f4;
HI_U32 u32LcdCrgValue = 0x08090c54;

s32Ret = HI_MAPI_Disp_SetReg (u32VdpCrgAddr, u32VdpCrgValue);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_Disp_SetReg fail s32Ret:%x\n",s32Ret);
    return HI_FAILURE;
}

s32Ret = HI_MAPI_Disp_SetReg (u32LcdCrgAddr, u32LcdCrgValue);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_Disp_SetReg fail s32Ret:%x\n",s32Ret);
    return HI_FAILURE;
}
```

#### 【相关主题】

[HI\\_MAPI\\_Disp\\_GetReg](#)





## HI\_MAPI\_Disp\_GetReg

### 【描述】

获取指定寄存器的值。

### 【语法】

```
HI_S32 HI_MAPI_Disp_GetReg(HI_U32 u32Addr, HI_U32 *pu32Value);
```

### 【参数】

参数名称	描述	输入/输出
u32Addr	寄存器地址	输入
pu32Value	指定寄存器的值	输出

### 【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

### 【需求】

- 头文件：hi\_mapi\_disp\_define.h、hi\_mapi\_disp.h
- 库文件：libhi3518e\_mapi\_disp.a

### 【举例】

```
HI_S32 s32Ret = HI_SUCCESS;
HI_U32 u32VdpCrgAddr = 0x12010044;
HI_U32 u32LcdCrgAddr = 0x12010048;

HI_U32 u32VdpCrgValue = 0x0;
HI_U32 u32LcdCrgValue = 0x0;

s32Ret = HI_MAPI_Disp_GetReg (u32VdpCrgAddr, &u32VdpCrgValue);
if(HI_SUCCESS != s32Ret)
{
    printf("HI_MAPI_Disp_GetReg fail s32Ret:%x\n",s32Ret);
    return HI_FAILURE;
}

s32Ret = HI_MAPI_Disp_GetReg (u32LcdCrgAddr, &u32LcdCrgValue);
if(HI_SUCCESS != s32Ret)
```



```
{  
    printf("HI_MAPI_Disp_GetReg fail s32Ret:%x\n",s32Ret);  
    return HI_FAILURE;  
}
```

#### 【相关主题】

[HI\\_MAPI\\_Disp\\_SetReg](#)

## 6.4 数据类型

视频输出相关数据类型定义如下：

- [HI\\_HANDLE](#)：定义设备 handle 号。
- [HI\\_MPP\\_DISP\\_ROTATE\\_E](#)：定义旋转枚举类型。
- [HI\\_MPP\\_DISP\\_ATTR\\_S](#)：定义视频输出设备属性结构体。
- [HI\\_MPP\\_DISP\\_WINDOW\\_ATTR\\_S](#)：定义视频输出通道属性结构体。
- [HI\\_MPP\\_DISP\\_CSCATTREX\\_S](#)：定义图像输出效果 csc 结构体。
- [HI\\_MPP\\_DISP\\_ROTATEATTREX\\_S](#)：定义视频图像旋转结构体。
- [HI\\_MPP\\_CSC\\_S](#)：定义图像输出效果 CSC 转换结构体。
- [HI\\_MPP\\_CSC\\_MATRIX\\_E](#)：定义 CSC 转换矩阵。
- [HI\\_MPP\\_DISP\\_CMD\\_E](#)：定义高级属性命令类型。

### HI\_HANDLE

#### 【说明】

定义设备 handle 号。

#### 【定义】

```
typedef HI_S32 HI_HANDLE;
```

#### 【成员】

芯片	描述
Hi3516CV200	视频输出模块有 1 个视频输出设备，如下定义： 0：DSD0 设备，即标清显示设备 0。

#### 【注意事项】

无

#### 【相关数据类型及接口】

无



## HI\_MPP\_DISP\_ROTATE\_E

### 【说明】

定义旋转枚举类型。

### 【定义】

```
typedef enum hiMPP_DISP_ROTATE_E
{
    HI_MPP_ROTATE_NONE = 0,
    HI_MPP_ROTATE_90   = 1,
    HI_MPP_ROTATE_180  = 2,
    HI_MPP_ROTATE_270  = 3,
    HI_MPP_ROTATE_BUTT
} HI_MPP_DISP_ROTATE_E;
```

### 【成员】

成员名称	描述
HI_MPP_ROTATE_NONE	不旋转。
HI_MPP_ROTATE_90	旋转 90 度。
HI_MPP_ROTATE_180	旋转 180 度。
HI_MPP_ROTATE_270	旋转 270 度。

### 【注意事项】

无

### 【相关数据类型及接口】

无

## HI\_MPP\_DISP\_ATTR\_S

### 【说明】

定义视频输出公共属性结构体。

### 【定义】

```
typedef struct hiMPP_DISP_ATTR_S
{
    HI_U32 u32BgColor;
    HI_MPP_DISP_INTF_TYPE_E enIntfType;
    HI_MPP_DISP_INTF_SYNC_E enIntfSync;
    HI_MPP_DISP_SYNC_INFO_S stSyncInfo;
    HI_MPP_DISP_ROTATE_E enDispRorate;
```



```
} HI_MPP_DISP_ATTR_S;
```

### 【成员】

成员名称	描述
u32BgColor	设备背景色，表示方法 RGB888。
enIntfType	接口类型典型配置，原型定义： <pre>typedef HI_S32 HI_MPP_DISP_INTF_TYPE_E; #define DISP_INTF_CVBS      (0x01L&lt;&lt;0) #define DISP_INTF_BT656     (0x01L&lt;&lt;3) #define DISP_INTF_BT1120    (0x01L&lt;&lt;4) #define DISP_INTF_HDMI      (0x01L&lt;&lt;5) #define DISP_INTF_LCD_6BIT  (0x01L&lt;&lt;9) #define DISP_INTF_LCD_8BIT  (0x01L&lt;&lt;10) #define DISP_INTF_LCD_16BIT (0x01L&lt;&lt;11) #define DISP_INTF_LCD_24BIT (0x01L&lt;&lt;12)</pre>
enIntfSync	接口时序典型配置，原型定义： <pre>typedef enum HiMPP_DISP_INTF_SYNC_E {     DISP_SYNC_PAL = 0,     DISP_SYNC_NTSC,      DISP_SYNC_1080P24,     DISP_SYNC_1080P25,     DISP_SYNC_1080P30,      DISP_SYNC_720P50,     DISP_SYNC_720P60,     DISP_SYNC_1080I50,     DISP_SYNC_1080I60,     DISP_SYNC_1080P50,     DISP_SYNC_1080P60,      DISP_SYNC_576P50,     DISP_SYNC_480P60,      DISP_SYNC_800x600_60, /* VESA 800 x 600 at 60 Hz (non-interlaced) */     DISP_SYNC_1024x768_60, /* VESA 1024 x 768 at 60 Hz (non-interlaced) */     DISP_SYNC_1280x1024_60, /* VESA 1280 x 1024 at 60 Hz (non-interlaced) */     DISP_SYNC_1366x768_60, /* VESA 1366 x 768 at 60</pre>



成员名称	描述
	<pre>Hz (non-interlaced) */     DISP_SYNC_1440x900_60,          /* VESA 1440 x 900 at 60 Hz (non-interlaced) CVT Compliant */     DISP_SYNC_1280x800_60,          /* 1280*800@60Hz VGA@60Hz*/     DISP_SYNC_1600x1200_60,          /* VESA 1600 x 1200 at 60 Hz (non-interlaced) */     DISP_SYNC_1680x1050_60,          /* VESA 1680 x 1050 at 60 Hz (non-interlaced) */     DISP_SYNC_1920x1200_60,          /* VESA 1920 x 1600 at 60 Hz (non-interlaced) CVT (Reduced Blanking)*/     DISP_SYNC_640x480_60,            /* VESA 640 x 480 at 60 Hz (non-interlaced) CVT */     DISP_SYNC_960H_PAL,              /* ITU-R BT.1302 960 x 576 at 50 Hz (interlaced)*/     DISP_SYNC_960H_NTSC,            /* ITU-R BT.1302 960 x 480 at 60 Hz (interlaced)*/     DISP_SYNC_320X240_60,            /* For ota5182 at 60 Hz (8bit) */     DISP_SYNC_320X240_50,            /* For ili9342 at 50 Hz (6bit) */     DISP_SYNC_240X320_50,            /* For ili9341 at 50 Hz (6bit) */     DISP_SYNC_240X320_60,            /* For ili9341 at 60 Hz (16bit) */     DISP_SYNC_800X600_50,            /* For LCD at 50 Hz (24bit) */     DISP_SYNC_USER,     DISP_SYNC_BUTT  } HI_MPP_DISP_INTF_SYNC_E;</pre>
stSyncInfo	<p>接口时序结构体，原型定义：</p> <pre>typedef struct HiMPP_DISP_SYNC_INFO_S {     HI_BOOL bSynm;     HI_BOOL bIop;     HI_U8 u8Intfb;     HI_U16 u16Vact ;     HI_U16 u16Vbb;     HI_U16 u16Vfb;     HI_U16 u16Hact;</pre>



成员名称	描述
	<pre>HI_U16 u16Hbb; HI_U16 u16Hfb; HI_U16 u16Hmid; HI_U16 u16Bvact; HI_U16 u16Bvbb; HI_U16 u16Bvfb; HI_U16 u16Hpw; HI_U16 u16Vpw; HI_BOOL bIdv; HI_BOOL bIhs; HI_BOOL bIvs; } HI_MPP_DISP_SYNC_INFO_S;</pre>
enDispRorate	<p>视频输出旋转结构体，原型定义：</p> <pre>typedef enum hiMPP_DISP_ROTATE_E {     HI_MPP_ROTATE_NONE = 0,     HI_MPP_ROTATE_90   = 1,     HI_MPP_ROTATE_180  = 2,     HI_MPP_ROTATE_270  = 3,     HI_MPP_ROTATE_BUTT } HI_MPP_DISP_ROTATE_E;</pre>

#### 【注意事项】

- 当接口时序配置为 DISP\_OUTPUT\_USER 时，stSyncInfo 定义的时序结构才会生效，表示用户自定义的时序结构。
- 同时使用多个接口类型示意：enIntfSync = DISP\_INTF\_BT1120 | DISP\_INTF\_HDMI。
- 接口类型配置为 DISP\_INTF\_CVBS 时，enIntfSync 的取值范围为[0, 1]。
- 接口类型配置为 DISP\_INTF\_BT1120 时，enIntfSync 的取值范围为[2, 12]。
- 接口类型配置为 DISP\_INTF\_HDMI、DISP\_INTF\_VGA 时，enIntfSync 的取值范围为[2, 22]。
- 接口类型配置为 DISP\_INTF\_BT656 时，enIntfSync 的取值范围为[0, 1]。
- 接口类型配置为 DISP\_INTF\_LCD\_8BIT 时，enIntfSync 的取值为 25。
- 接口类型配置为 DISP\_INTF\_LCD\_6BIT 时，enIntfSync 的取值范围为[26, 27]。
- 接口类型配置为 DISP\_INTF\_LCD\_16BIT 时，enIntfSync 的取值为 28。
- 配置设备属性在设备下一次打开时生效。
- 用户自定义时序时，需要保证时序的有效性，并且需要自行配置 VDP 的 CRG 寄存器。CRG 的寄存器配置请参考《Hi35xx 专业型 HD IP Camera SoC 用户指南》的系统章节。



- 不支持 DISP\_OUTPUT\_960H\_PAL、DISP\_OUTPUT\_960H\_NTSC、DISP\_OUTPUT\_1600x1200\_60 和 DISP\_OUTPUT\_1920x1200\_60。

【相关数据类型及接口】

[HI\\_MAPI\\_Disp\\_Init](#)

## HI\_MPP\_DISP\_WINDOW\_ATTR\_S

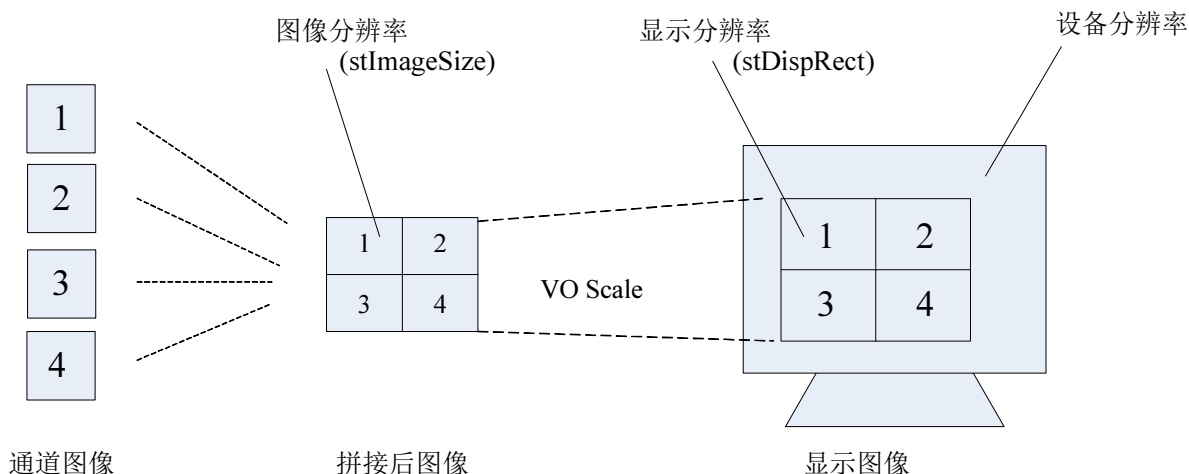
【说明】

定义视频输出通道属性。

在视频层属性中存在三个概念，即设备分辨率、显示分辨率和图像分辨率。每种分辨率的概念可以从图 6-1 中看出。

- 图像分辨率指放置各个通道图像的画布大小。
- 显示分辨率是把图像分辨率中描述的画布经过 VO 放大后的显示区域。
- 设备分辨率与设备时序一致，即如果时序为 1920 x 1080，那设备分辨率就为 1920 x 1080。

图6-1 视频层属性的相关概念示意



【定义】

```
typedef struct hiMPP_DISP_WINDOW_ATTR_S
{
    HI_MPP_RECT_S stRect;
    HI_U32 u32Priority;
} HI_MPP_DISP_WINDOW_ATTR_S;
```

【成员】



成员名称	描述
u32Priority	视频通道叠加优先级，优先级高的在上层。 该属性只针对标清设备。 标清设备的优先级取值范围：[0, VO_MAX_CHN_NUM-1]。 动态属性。
stRect	通道矩形显示区域。以屏幕的左上角为原点。其取值必须是 2 对齐，且该矩形区域必须在屏幕范围之内。 <b>注意：当高清设备是隔行时序且 SPYCbCr420 显示时，高度必须 4 对齐。</b> 动态属性。

#### 【说明】

芯片	描述
Hi3516CV200	<ul style="list-style-type: none"><li>标清设备的优先级为 0~VO_MAX_CHN_NUM-1</li><li>标清设备的通道宽高必须不小于 32</li></ul>

#### 【注意事项】

- 属性中的优先级，数值越大优先级越高。  
标清设备：优先级有 0~VO\_MAX\_CHN\_NUM-1，当多个通道有重叠的显示区域时，优先级高的通道图像将覆盖优先级低的通道。优先级相同的各通道有重叠时，默认通道号大的图像将覆盖通道号小的通道图像。
- 通道显示区域不能超过视频层属性中设定的画布大小(stImageSize 大小)。
- HD 设备的通道属性 stRect 需 2 对齐，在隔行时序且像素格式为 SEMIPLANAR\_420 时，stRect 的高度必须 4 对齐。HD 的宽高必须大于等于 32。
- 如果 HD 有视频层放大的情况，stRect 是放大前视频层上的起始位置和宽高，放大后显示的起始位置和宽高会按视频层放大的比例偏移或放大。
- SD 设备的通道属性 stRect 需 2 对齐。

#### 【相关数据类型及接口】

[HI\\_MAPI\\_Disp\\_Window\\_SetAttr](#)

## HI\_MPP\_DISP\_ROTATEATTREX\_S

#### 【说明】

定义图像旋转结构体。

#### 【定义】

```
typedef struct hiMPP_DISP_ROTATEATTREX_S
{
    HI\_MPP\_DISP\_ROTATE\_E enDispRorate;
```





```
} HI_MPP_DISP_ROTATEATTREX_S;
```

#### 【成员】

成员名称	描述
enDispRorate	旋转角度。

#### 【注意事项】

无

#### 【相关数据类型及接口】

[HI\\_MAPI\\_Dispatch\\_SetAttrEx](#)

## HI\_MPP\_DISP\_CSCATTREX\_S

#### 【说明】

定义图像输出效果 csc 结构体。

#### 【定义】

```
typedef struct hiMPP_DISP_CSCATTREX_S  
{  
    HI_MPP_CSC_S stVideoCSC;  
} HI_MPP_DISP_CSCATTREX_S;
```

#### 【成员】

成员名称	描述
stVideoCSC	CSC 矩阵结构体。

#### 【注意事项】

无

#### 【相关数据类型及接口】

[HI\\_MAPI\\_Dispatch\\_SetAttrEx](#)

## HI\_MPP\_CSC\_S

#### 【说明】

定义图像输出效果 CSC 转换结构体。

#### 【定义】

```
typedef struct hiMPP_CSC_S  
{
```



```
HI_MPP_CSC_MATRIX_E enCscMatrix;
HI_U32 u32Luma;          /* luminance:  0 ~ 100 default: 50 */
HI_U32 u32Contrast;       /* contrast :  0 ~ 100 default: 50
*/
HI_U32 u32Hue;            /* hue      :  0 ~ 100 default: 50 */
HI_U32 u32Satuature;      /* satuature: 0 ~ 100 default: 50
*/
} HI_MPP_CSC_S;
```

【成员】

成员名称	描述
enCscMatrix	CSC 矩阵选择。
u32Luma	亮度值。 取值范围：[0,100]。
u32Contrast	对比度值。 取值范围：[0,100]。
u32Hue	色调值。 取值范围：[0,100]。
u32Satuature	饱和度值。 取值范围：[0,100]。

【注意事项】

无

【相关数据类型及接口】

[HI\\_MAPI\\_Disp\\_SetAttrEx](#)

HI\_MPP\_CSC\_MATRIX\_E

【说明】

定义 CSC 转换矩阵。

【定义】

```
typedef enum hiMPP_CSC_MATRIX_E
{
    HI_MPP_CSC_MATRIX_IDENTITY = 0,

    HI_MPP_CSC_MATRIX_BT601_TO_BT709,
    HI_MPP_CSC_MATRIX_BT709_TO_BT601,
```



```
HI_MPP_CSC_MATRIX_BT601_TO_RGB_PC,  
HI_MPP_CSC_MATRIX_BT709_TO_RGB_PC,  
  
HI_MPP_CSC_MATRIX_RGB_TO_BT601_PC,  
HI_MPP_CSC_MATRIX_RGB_TO_BT709_PC,  
  
HI_MPP_CSC_MATRIX_BUTT  
} HI_MPP_CSC_MATRIX_E;
```

#### 【成员】

成员名称	描述
HI_MPP_CSC_MATRIX_IDENTITY	单位转换矩阵。
HI_MPP_CSC_MATRIX_BT601_TO_BT709	BT.601 到 BT.709 色彩空间的 CSC 矩阵。
HI_MPP_CSC_MATRIX_BT709_TO_BT601	BT.709 到 BT.601 色彩空间的 CSC 矩阵。
HI_MPP_CSC_MATRIX_BT601_TO_RGB_PC	BT.601 到 RGB 色彩空间的 CSC 矩阵。
HI_MPP_CSC_MATRIX_BT709_TO_RGB_PC	BT.709 到 RGB 色彩空间的 CSC 矩阵。
HI_MPP_CSC_MATRIX_RGB_TO_BT601_PC	RGB 到 BT.601 色彩空间的 CSC 矩阵
HI_MPP_CSC_MATRIX_RGB_TO_BT709_PC	RGB 到 BT.709 色彩空间的 CSC 矩阵

#### 【注意事项】

无

#### 【相关数据类型及接口】

[HI\\_MAPI\\_Disp\\_SetAttrEx](#)

## HI\_MPP\_DISP\_CMD\_E

#### 【说明】

定义高级属性命令类型。

#### 【定义】

```
typedef enum hiMPP_DISP_CMD_E  
{  
    HI_MPP_DISP_CMD_SETCSC,          /**< HI_MPP_DISP_ATTREX_S*/
```



```

HI_MPP_DISP_CMD_GETCSC,          /**< HI_MPP_DISP_ATTREX_S*/
HI_MPP_DISP_CMD_SETROTATE,       /**< HI_MPP_DISP_ROTATE_E*/
HI_MPP_DISP_CMD_GETROTATE,       /**< HI_MPP_DISP_ROTATE_E*/
HI_MPP_DISP_CMD_BUTT
}HI_MPP_DISP_CMD_E;

```

#### 【成员】

成员名称	描述
HI_MPP_DISP_CMD_SETCSC	设置 csc 属性命令。
HI_MPP_DISP_CMD_GETCSC	获取 csc 属性命令。
HI_MPP_DISP_CMD_SETROTATE	设置旋转属性命令。
HI_MPP_DISP_CMD_GETROTATE	获取旋转属性命令。

#### 【注意事项】

无

#### 【相关数据类型及接口】

- [HI\\_MAPI\\_Disp\\_SetAttrEx](#)
- [HI\\_MAPI\\_Disp\\_GetAttrEx](#)

## 6.5 错误码

视频输出 API 错误码如表 6-4 所示。

表6-4 视频输出 API 错误码

错误代码	宏定义	描述
0xA3088001	HI_MAPI_ERR_DISP_INVALID_DEVID	设备 ID 超出合法范围
0xA3088002	HI_MAPI_ERR_DISP_INVALID_CHNID	通道 ID 超出合法范围
0xA3088003	HI_MAPI_ERR_DISP_ILLEGAL_PARAM	参数超出合法范围
0xA3088006	HI_MAPI_ERR_DISP_NULL_PTR	函数参数中有空指针
0xA3088008	HI_MAPI_ERR_DISP_NOT_SUPPORT	不支持的操作
0xA3088009	HI_MAPI_ERR_DISP_NOT_PERM	操作不允许
0xA308800C	HI_MAPI_ERR_DISP_NOMEM	内存不足
0xA3088010	HI_MAPI_ERR_DISP_SYS_NOTREADY	系统未初始化



错误代码	宏定义	描述
0xA3088012	HI_MAPI_ERR_DISP_BUSY	资源忙
0xA3088040	HI_MAPI_ERR_DISP_DEV_NOT_CONFIG	设备未配置
0xA3088041	HI_MAPI_ERR_DISP_DEV_NOT_ENABLE	设备未使能
0xA3088042	HI_MAPI_ERR_DISP_DEV_HAS_ENABLE D	设备已使能
0xA3088043	HI_MAPI_ERR_DISP_DEV_HAS_BINDED	设备已被绑定
0xA3088044	HI_MAPI_ERR_DISP_DEV_NOT_BINDED	设备未被绑定
0xA3088045	HI_MAPI_ERR_DISP_VIDEO_NOT_ENAB LE	视频层未使能
0xA3088046	HI_MAPI_ERR_DISP_VIDEO_NOT_DISAB LE	视频层未禁止
0xA3088047	HI_MAPI_ERR_DISP_VIDEO_NOT_CONFI G	视频层未配置
0xA3088048	HI_MAPI_ERR_DISP_CHN_NOT_DISABL E	通道未禁止
0xA3088049	HI_MAPI_ERR_DISP_CHN_NOT_ENABLE	通道未使能
0xA308804A	HI_MAPI_ERR_DISP_CHN_NOT_CONFIG	通道未配置
0xA308804B	HI_MAPI_ERR_DISP_CHN_NOT_ALLOC	通道未分配资源
0xA308804C	HI_MAPI_ERR_DISP_INVALID_PATTERN	无效样式
0xA308804D	HI_MAPI_ERR_DISP_INVALID_POSITION	无效级联位置
0xA308804E	HI_MAPI_ERR_DISP_WAIT_TIMEOUT	等待超时
0xA308804F	HI_MAPI_ERR_DISP_INVALID_VFRAME	无效视频帧
0xA3088050	HI_MAPI_ERR_DISP_INVALID_RECT_PA RA	无效矩形参数
0xA3088068	HI_MAPI_ERR_DISP_GFX_INVALID_ID	图形层 ID 超出范围
0xA308806b	HI_MAPI_ERR_DISP_CHN_AREA_OVERL AP	VO 通道区域重叠



## 目 录

7 音频采集.....	7-1
7.1 概述.....	7-1
7.2 功能描述.....	7-1
7.2.1 音频采集原理.....	7-1
7.2.2 输入时序 .....	7-1
7.3 API 参考 .....	7-3
7.4 数据类型.....	7-10
7.5 错误码.....	7-15



## 插图目录

图 7-1 音频采集原理示意图.....	7-1
图 7-2 I <sup>2</sup> S 1/2/4/8/16 路接收 .....	7-3



## 表格目录

表 7-1 音频采集 API 错误码 .....	7-15
--------------------------	------





# 7 音频采集

## 7.1 概述

音频采集(Audio Capture, 简称 ACAP)用于接收 Audio Codec 转换之后的音频数字信号采集到内存, 完成录音, 用于送给音频编码或音频输出模块。

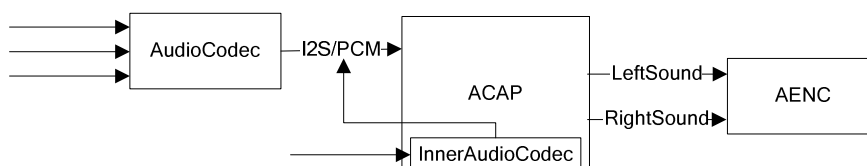
## 7.2 功能描述

### 7.2.1 音频采集原理

原始音频信号以模拟信号的形式给出后, 通过 Audio Codec, 按一定采样率和采样精度转换为数字信号。Audio Codec 以 I<sup>2</sup>S 时序或 PCM 时序的方式, 将数字信号传输给音频采集单元。芯片利用 DMA 将采集到的音频数据搬移到内存中, 完成录音操作。

目前音频采集单元默认支持一个内部 Codec 单元, 同时支持对接外部 Codec。

图7-1 音频采集原理示意图



### 7.2.2 输入时序

音频接口支持标准的 I<sup>2</sup>S 接口时序模式和 PCM 接口时序模式, 并提供灵活的配置以支持与多种 Audio Codec 对接。详细的时序支持情况请参考对应芯片的用户指南。

为实现成功对接, 需要对 I<sup>2</sup>S 或者 PCM 协议以及对接的 Audio Codec 时序支持情况有足够了解, 这里只简单介绍下 I<sup>2</sup>S 及 PCM 接口时序的几个特性:

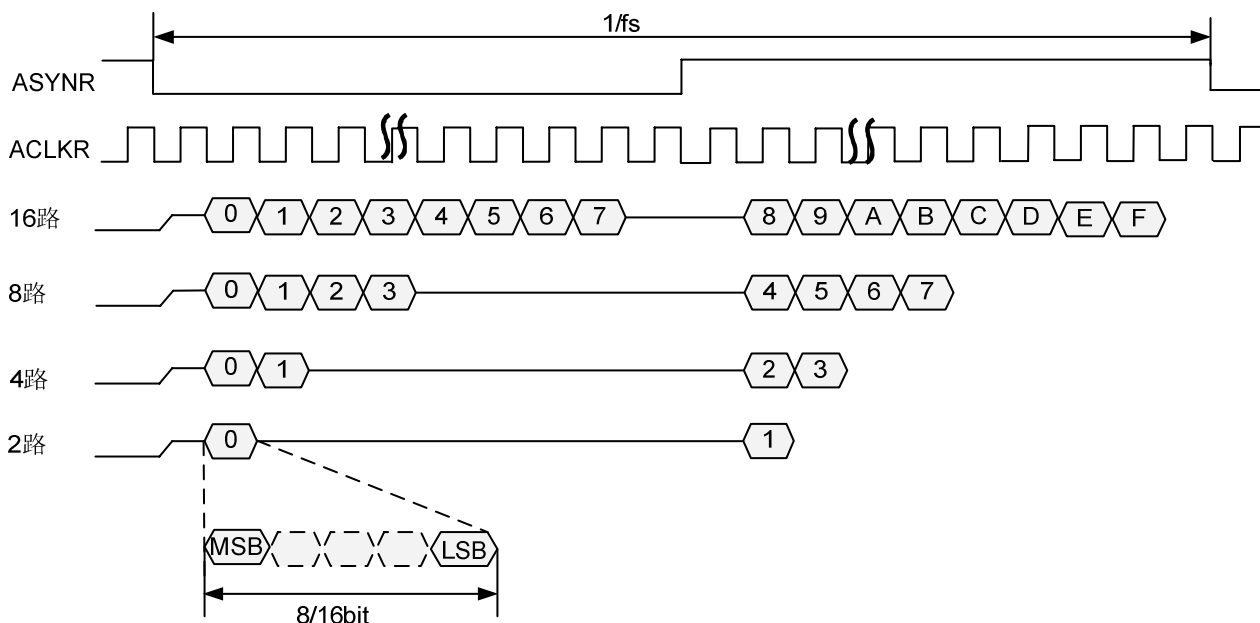
- 按照标准 I<sup>2</sup>S 或 PCM 协议, 总是先传送最高有效位 MSB, 后传送最低有效位 LSB, 即按照从高位到低位的顺序传输串行数据。



- ACAP 支持扩展的多路接收的 I<sup>2</sup>S 及 PCM 接口时序。对接时，Codec 的时序模式选择、同步时钟、采样位宽等配置必须与 ACAP 的配置保持一致，否则可能采集不到正确的数据。
- ACAP 支持主模式和从模式，主模式即 ACAP 提供时钟，从模式即 Audio Codec 提供时钟。主模式时，如果 ACAP 与 AO 同时对接同一个 Codec，则时钟供输入和输出共同使用，其他情况没有此限制。而从模式时的输入输出时钟可以分别由外围 Audio Codec 提供。
- 由于时序的问题，在 ACAP/AO 从模式下，建议用户先配置好对接的 Codec，再配置 ACAP 或 AO 单元；而在 ACAP/AO 主模式下，建议用户先配置好 ACAP 或 AO 单元，再配置对接的 Codec。
- ACAP/AO 选择主模式时，有些 ACAP/AO 只提供用于时序同步的帧同步时钟和位流时钟，不提供 MCLK，这时如果 Audio Codec 使用外接的晶振作为工作时钟，这样可能导致声音失真，因此推荐使用从模式或者使用位流时钟产生 Codec 内部工作主时钟。
- 当 ACAP/AO 为主模式时，对于向外提供了 MCLK 的 ACAP/AO，MCLK 的设定为：
  - 采样率为 96k/48k/24k/12k 时，提供 12.288MHz 的主时钟。
  - 采样率为 32k/16k/8k（32k 采样位宽不是 256bits，8k 要求采样位宽不是 16bits）时，提供 12.288MHz 的主时钟。
  - 采样率为 64k/32k/16k/8k（32k 采样位宽为 256bits 或 8k 采样位宽为 16bits）时，提供 8.192MHz 的主时钟。
  - 采样率为 44.1k/22.05k/11.025 时，提供 11.2896MHz 的主时钟。
- ACAP/AO 支持标准 PCM 模式和自定义 PCM 模式。PCM 模式下只支持单声道模式，因此在 PCM 模式下，需要关闭对接 Codec 的右声道输出，否则声音会有杂音。根据 PCM 标准模式协议，当工作模式为标准 PCM 主模式或标准 PCM 从模式时，音频输入输出的数据相对帧同步信号延迟 1 个位流时钟（BCLK）周期。如果在标准 PCM 主模式或标准 PCM 从模式下对接外置的 Codec，需要确保外置 Codec 的数据相对帧同步信号延迟的位流时钟（BCLK）周期数大于音频输入输出数据相对帧同步信号延迟的位流时钟周期数（1 个 BCLK），否则声音会有杂音。



图7-2 I<sup>2</sup>S 1/2/4/8/16 路接收



## 7.3 API 参考

该功能模块为用户提供以下 MAPI:

- [HI\\_MAPI\\_ACap\\_Init](#): 初始化音频采集单元。
- [HI\\_MAPI\\_ACap\\_DeInit](#): 销毁音频采集单元。
- [HI\\_MAPI\\_ACap\\_Start](#): 启动 ACAP, 开始接收音频帧数据。
- [HI\\_MAPI\\_ACap\\_Stop](#): 停止 ACAP。
- [HI\\_MAPI\\_ACap\\_SetVolume](#): 设置音频输入的音量。
- [HI\\_MAPI\\_ACap\\_GetVolume](#): 获取音频输入的音量。
- [HI\\_MAPI\\_ACap\\_Mute](#): 静音音频输入。
- [HI\\_MAPI\\_ACap\\_UnMute](#): 取消静音。

### HI\_MAPI\_ACap\_Init

#### 【描述】

初始化音频采集单元。

#### 【语法】

```
HI_S32 HI_MAPI_ACap_Init(HI_HANDLE ACapHdl, HI_MPP_ACAP_ATTR_S*  
pstACapAttr);
```

#### 【参数】



参数名称	描述	输入/输出
ACapHdl	ACap 的 Handle 号。 取值范围：0。	输入
pstACapAttr	ACap 属性指针。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_acap\_define.h、hi\_mapi\_acap.h、hi\_mapi\_comm\_define.h
- 库文件：libhi3518e\_mapi\_acap.a

#### 【注意】

- 若外接 AudioCodec 时，必须保证 enSampleRate 与 Codec 配置的采样率保持一致。
- 其他对接外部 Codec 的注意事项请参看 7.2.2 “[输入时序](#)” 章节。
- 在参数没发生变化的情况下重复初始化，直接返回成功。若参数发生变化，则必须调用 [HI\\_MAPI\\_ACap\\_Stop](#) 将采集单元停止之后重新初始化。

#### 【举例】

无

#### 【相关主题】

无

## HI\_MAPI\_ACap\_DeInit

#### 【描述】

去初始化音频采集单元。

#### 【语法】

```
HI_S32 HI_MAPI_ACap_DeInit(HI_HANDLE ACapHdl);
```

#### 【参数】



参数名称	描述	输入/输出
ACapHdl	ACap 的 Hhandle 号。 取值范围：0。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_acap\_define.h、hi\_mapi\_acap.h、hi\_mapi\_comm\_define.h
- 库文件：libhi3518e\_mapi\_acap.a

【注意】

无

【举例】

无

【相关主题】

[HI\\_MAPI\\_ACap\\_Init](#)

## HI\_MAPI\_ACap\_Start

【描述】

启动 ACap。

【语法】

```
HI_S32 HI_MAPI_ACap_Start(HI_HANDLE ACapHdl);
```

【参数】

参数名称	描述	输入/输出
ACapHdl	ACap 的 Handle 号。 取值范围：0。	输入

【返回值】



返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_acap\_define.h、hi\_mapi\_acap.h、hi\_mapi\_comm\_define.h
- 库文件：libhi3518e\_mapi\_acap.a

【注意】

- ACap 必须已经初始化。
- 可重复启动，如果已经启动，则直接返回成功。

【举例】

无

【相关主题】

[HI\\_MAPI\\_ACap\\_Init](#)

## HI\_MAPI\_ACap\_Stop

【描述】

停止 ACap。

【语法】

```
HI_S32 HI_MAPI_ACap_Stop(HI_HANDLE ACapHdl);
```

【参数】

参数名称	描述	输入/输出
ACapHdl	ACap 的 Handle 号。 取值范围：0。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】



- 头文件: hi\_mapi\_acap\_define.h、hi\_mapi\_acap.h、hi\_mapi\_comm\_define.h
- 库文件: libhi3518e\_mapi\_acap.a

【注意】

可重复停止，如果已经停止，则直接返回成功。

【举例】

无

【相关主题】

[HI\\_MAPI\\_ACap\\_Start](#)

## HI\_MAPI\_ACap\_SetVolume

【描述】

设置音频输入的音量。

【语法】

```
HI_S32 HI_MAPI_ACap_SetVolume (HI_HANDLE ACapHdl, HI\_MPP\_AUDIO\_GAIN\_S\*  
pstVol);
```

【参数】

参数名称	描述	输入/输出
ACapHdl	ACap 的 Handle 号。 取值范围：0。	输入
pstVol	音频输入音量结构体。 取值范围：[-78,80]，推荐值为：[19,50]。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件: hi\_mapi\_acap\_define.h、hi\_mapi\_acap.h、hi\_mapi\_comm\_define.h
- 库文件: libhi3518e\_mapi\_acap.a

【注意】



音频采集单元中采用了声音质量增强算法自适应，会对音质进行调节，当音量设置为 20 时，调节效果最优，因此，建议设置音量为 20。

【举例】

无

【相关主题】

无

## HI\_MAPI\_ACap\_GetVolume

【描述】

获取音频输入的音量。

【语法】

```
HI_S32 HI_MAPI_ACap_GetVolume(HI_HANDLE ACapHdl, HI_MPP_AUDIO_GAIN_S*  
pstVol);
```

【参数】

参数名称	描述	输入/输出
ACapHdl	ACap 的 Hhandle 号。 取值范围：0。	输入
pstVol	音频输入音量结构体。	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_acap\_define.h、hi\_mapi\_acap.h、hi\_mapi\_comm\_define.h
- 库文件：libhi3518e\_mapi\_acap.a

【注意】

无

【举例】

无

【相关主题】





无

## HI\_MAPI\_ACap\_Mute

### 【描述】

设置音频输入为静音。

### 【语法】

```
HI_S32 HI_MAPI_ACap_Mute(HI_HANDLE ACapHdl);
```

### 【参数】

参数名称	描述	输入/输出
ACapHdl	ACap 的 Handle 号。 取值范围：0。	输入

### 【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

### 【需求】

- 头文件：hi\_mapi\_acap\_define.h、hi\_mapi\_acap.h、hi\_mapi\_comm\_define.h
- 库文件：libhi3518e\_mapi\_acap.a

### 【注意】

- 当设置为静音时，输出为静音数据。
- 当恢复静音时，音量保持为静音之前的音量大小。

### 【举例】

无

### 【相关主题】

[HI\\_MAPI\\_ACap\\_UnMute](#)

## HI\_MAPI\_ACap\_UnMute

### 【描述】

取消音频输入的静音。

### 【语法】



```
HI_S32 HI_MAPI_ACap_UnMute(HI_HANDLE ACapHdl);
```

#### 【参数】

参数名称	描述	输入/输出
ACapHdl	ACap 的 Handle 号。 取值范围：0。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

#### 【需求】

- 头文件：hi\_mapi\_acap\_define.h、hi\_mapi\_acap.h、hi\_mapi\_comm\_define.h
- 库文件：libhi3518e\_mapi\_acap.a

#### 【注意】

无

#### 【举例】

无

#### 【相关主题】

[HI\\_MAPI\\_ACap\\_Mute](#)

## 7.4 数据类型

音频采集模块相关数据类型定义如下：

- [HI\\_MPP\\_ACAP\\_ATTR\\_S](#)：定义 ACap 的属性。
- [HI\\_MPP\\_AUDIO\\_SAMPLE\\_RATE\\_E](#)：音频采样率。
- [HI\\_MPP\\_AUDIO\\_BITWIDTH\\_E](#)：音频位宽。
- [HI\\_MPP\\_AUDIO\\_SOUND\\_MODE\\_E](#)：音频声道数。
- [HI\\_MPP\\_AUDIO\\_MODE\\_E](#)：音频接口时序模式。
- [HI\\_MPP\\_AUDIO\\_GAIN\\_S](#)：音频增益。



## HI\_MPP\_ACAP\_ATTR\_S

### 【说明】

定义 ACap 属性的结构体。

### 【定义】

```
typedef struct hiMPP_ACAP_ATTR_S
{
    HI_MPP_AUDIO_SAMPLE_RATE_E enSampleRate;
    HI_MPP_AUDIO_BITWIDTH_E enBitwidth;
    HI_MPP_AUDIO_SOUND_MODE_E enSoundMode;
    HI_MPP_AUDIO_MODE_E enAudioMode;
    HI_U32 u32PtNumPerFrm;
} HI_MPP_ACAP_ATTR_S;
```

### 【成员】

成员名称	描述
enSampleRate	音频采样率。
enBitwidth	音频采样精度。
enSoundMode	音频输入声道模式。
enAudioMode	音频接口时序模式。
u32PtNumPerFrm	每帧的采样点个数。

### 【注意事项】

无

### 【相关数据类型及接口】

- [HI\\_MAPI\\_ACap\\_Init](#)
- [HI\\_MPP\\_AUDIO\\_SAMPLE\\_RATE\\_E](#)
- [HI\\_MPP\\_AUDIO\\_BITWIDTH\\_E](#)
- [HI\\_MPP\\_AUDIO\\_SOUND\\_MODE\\_E](#)
- [HI\\_MPP\\_AUDIO\\_MODE\\_E](#)

## HI\_MPP\_AUDIO\_SAMPLE\_RATE\_E

### 【说明】

定义音频采样率。

### 【定义】

```
typedef enum hiMPP_AUDIO_SAMPLE_RATE_E
```



```
{
    HI_MPP_AUDIO_SAMPLE_RATE_8      = 8000,    /* 8K Sample rate */
    HI_MPP_AUDIO_SAMPLE_RATE_11025 = 11025,    /* 11.025K Sample rate*/
    HI_MPP_AUDIO_SAMPLE_RATE_16     = 16000,    /* 16K Sample rate */
    HI_MPP_AUDIO_SAMPLE_RATE_22050 = 22050,    /* 22.050K Sample rate*/
    HI_MPP_AUDIO_SAMPLE_RATE_24     = 24000,    /* 24K Sample rate */
    HI_MPP_AUDIO_SAMPLE_RATE_32     = 32000,    /* 32K Sample rate */
    HI_MPP_AUDIO_SAMPLE_RATE_441    = 44100,    /* 44.1K Sample rate */
    HI_MPP_AUDIO_SAMPLE_RATE_48     = 48000,    /* 48K Sample rate */
    HI_MPP_AUDIO_SAMPLE_RATE_BUTT
} HI_MPP_AUDIO_SAMPLE_RATE_E;
```

#### 【成员】

成员名称	描述
HI_MPP_AUDIO_SAMPLE_RATE_8	8kHz 采样率。
HI_MPP_AUDIO_SAMPLE_RATE_11025	11.025kHz 采样率。
HI_MPP_AUDIO_SAMPLE_RATE_16	16kHz 采样率。
HI_MPP_AUDIO_SAMPLE_RATE_22050	22.050kHz 采样率。
HI_MPP_AUDIO_SAMPLE_RATE_24	24kHz 采样率。
HI_MPP_AUDIO_SAMPLE_RATE_32	32kHz 采样率。
HI_MPP_AUDIO_SAMPLE_RATE_441	44.1kHz 采样率。
HI_MPP_AUDIO_SAMPLE_RATE_48	48kHz 采样率。

#### 【注意事项】

- 这里枚举值不是从 0 开始，而是与实际的采样率值相同。
- 目前仅支持 8kHz、16kHz 和 48kHz 采样率。

#### 【相关数据类型及接口】

[HI\\_MPP\\_ACAP\\_ATTR\\_S](#)

## HI\_MPP\_AUDIO\_BITWIDTH\_E

#### 【说明】

定义音频位宽。

#### 【定义】

```
typedef enum hiMPP_AUDIO_BITWIDTH_E
{
    HI_MPP_AUDIO_BITWIDTH_8 = 0,    /* Bit width is 8 bits */

```



```
HI_MPP_AUDIO_BITWIDTH_16 = 1, /* Bit width is 16 bits */
HI_MPP_AUDIO_BITWIDTH_24 = 2, /* Bit width is 24 bits */
HI_MPP_AUDIO_BITWIDTH_BUTT
} HI_MPP_AUDIO_BITWIDTH_E;
```

#### 【成员】

成员名称	描述
HI_MPP_AUDIO_BITWIDTH_8	采样精度为 8bit 位宽。
HI_MPP_AUDIO_BITWIDTH_16	采样精度为 16bit 位宽。
HI_MPP_AUDIO_BITWIDTH_24	采样精度为 24bit 位宽。

#### 【注意事项】

目前只支持 16bit 位宽。

#### 【相关数据类型及接口】

[HI\\_MPP\\_ACAP\\_ATTR\\_S](#)

## HI\_MPP\_AUDIO\_SOUND\_MODE\_E

#### 【说明】

定义音频声道模式。

#### 【定义】

```
typedef enum hiMPP_AUDIO_SOUND_MODE_E
{
    HI_MPP_AUDIO_SOUND_MODE_LEFT = 0,
    HI_MPP_AUDIO_SOUND_MODE_RIGHT = 1,
    HI_MPP_AUDIO_SOUND_MODE_STEREO = 2,
    HI_MPP_AUDIO_SOUND_MODE_BUTT
} HI_MPP_AUDIO_SOUND_MODE_E;
```

#### 【成员】

成员名称	描述
HI_MPP_AUDIO_SOUND_MODE_LEFT	左声道。
HI_MPP_AUDIO_SOUND_MODE_RIGHT	右声道。
HI_MPP_AUDIO_SOUND_MODE_STEREO	立体声。

#### 【注意事项】



无

【相关数据类型及接口】

[HI\\_MPP\\_ACAP\\_ATTR\\_S](#)

## HI\_MPP\_AUDIO\_MODE\_E

【说明】

定义音频输入时序模式。

【定义】

```
typedef enum hiMPP_AUDIO_MODE_E
{
    HI_MPP_AUDIO_MODE_I2S_MASTER = 0,
    HI_MPP_AUDIO_MODE_I2S_SLAVE,
    HI_MPP_AUDIO_MODE_PCM_SLAVE_STD,
    HI_MPP_AUDIO_MODE_PCM_SLAVE_NSTD,
    HI_MPP_AUDIO_MODE_PCM_MASTER_STD,
    HI_MPP_AUDIO_MODE_PCM_MASTER_NSTD,
    HI_MPP_AUDIO_MODE_BUTT
} HI_MPP_AUDIO_MODE_E;
```

【成员】

成员名称	描述
HI_MPP_AUDIO_MODE_I2S_MASTER	I <sup>2</sup> S 主模式。
HI_MPP_AUDIO_MODE_I2S_SLAVE	I <sup>2</sup> S 从模式。
HI_MPP_AUDIO_MODE_PCM_SLAVE_STD	PCM 从模式（标准协议）。
HI_MPP_AUDIO_MODE_PCM_SLAVE_NSTD	PCM 从模式（自定义协议）。
HI_MPP_AUDIO_MODE_PCM_MASTER_STD	PCM 主模式（标准协议）。
HI_MPP_AUDIO_MODE_PCM_MASTER_NSTD	PCM 主模式（自定义协议）。

【注意事项】

在对接内置 Codec 时，仅支持 I2S 主模式。

【相关数据类型及接口】

[HI\\_MPP\\_ACAP\\_ATTR\\_S](#)

## HI\_MPP\_AUDIO\_GAIN\_S

【说明】



定义音频增益。

#### 【定义】

```
typedef struct hiMPP_AUDIO_GAIN_S
{
    HI_S32 s32Gain;
}HI_MPP_AUDIO_GAIN_S;
```

#### 【成员】

成员名称	描述
s32Gain	音频增益。

#### 【注意事项】

无

#### 【相关数据类型及接口】

- [HI\\_MAPI\\_ACap\\_SetVolume](#)
- [HI\\_MAPI\\_ACap\\_GetVolume](#)

## 7.5 错误码

音频采集 API 错误码如表 7-1 所示。

表7-1 音频采集 API 错误码

错误代码	宏定义	描述
0xA3048003	HI_ERR_MAPI_ACAP_ILLEGAL_PARA	参数设置无效
0xA3048006	HI_ERR_MAPI_ACAP_NULL_PTR	参数空指针错误
0xA3048008	HI_ERR_MAPI_ACAP_NOTSUPPORT	操作不支持
0xA3048040	HI_ERR_MAPI_ACAP_INVALID_FD	非法 FD
0xA3048041	HI_ERR_MAPI_ACAP_IOCTL_FAIL	IOCTL 错误
0xA3048042	HI_ERR_MAPI_ACAP_OVERRANGE	参数越界
0xA3048043	HI_ERR_MAPI_ACAP_NOT_INITED	音频采集模块未初始化
0xA3048044	HI_ERR_MAPI_ACAP_NOT_STARTED	音频采集模块未启动



## 目 录

8 音频编码.....	8-1
8.1 概述.....	8-1
8.2 功能描述.....	8-1
8.3 API 参考 .....	8-1
8.4 数据类型.....	8-11
8.5 错误码.....	8-16





## 表格目录

表 8-1 音频编码 API 错误码 .....	8-16
--------------------------	------



# 8 音频编码

## 8.1 概述

音频编码主要实现创建编码通道、绑定音频输入通道及获取编码码流等功能。

## 8.2 功能描述

当前仅支持 AAC 编码。

## 8.3 API 参考

音频编码功能模块提供以下 MAPI：

- [HI\\_MAPI\\_AEnc\\_Init](#)：初始化编码通道。
- [HI\\_MAPI\\_AEnc\\_DeInit](#)：去初始化编码通道。
- [HI\\_MAPI\\_AEnc\\_Start](#)：启动编码通道。
- [HI\\_MAPI\\_AEnc\\_Stop](#)：停止编码通道。
- [HI\\_MAPI\\_AEnc\\_RegisterCallback](#)：注册编码回调函数。
- [HI\\_MAPI\\_AEnc\\_UnRegisterCallback](#)：解注册编码回调函数。
- [HI\\_MAPI\\_AEnc\\_BindACap](#)：绑定 ACap。
- [HI\\_MAPI\\_AEnc\\_UnBindACap](#)：解绑定 ACap。
- [HI\\_MAPI\\_Register\\_ExtAudioEncoder](#)：注册外部音频编码器。
- [HI\\_MAPI\\_UnRegister\\_ExtAudioEncoder](#)：解注册外部音频编码器。

### HI\_MAPI\_AEnc\_Init

#### 【描述】

初始化音频编码通道。

#### 【语法】



```
HI_S32 HI_MAPI_AEnc_Init(HI_HANDLE AEncHdl, HI_MPP_AENC_ATTR_S*  
pstAencAttr);
```

#### 【参数】

参数名称	描述	输入/输出
AEncHdl	音频编码通道句柄。 取值范围：[0, <a href="#">HI_AENC_MAX_CHN_NUM</a> )。	输入
pstAencAttr	编码通道参数指针。 静态属性。	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】

无。

#### 【需求】

- 头文件：hi\_mapi\_aenc\_define.h、hi\_mapi\_aenc.h
- 库文件：libhi3518e\_mapi\_aenc.a

#### 【注意】

无

#### 【举例】

请参考 sample\_aenc.c

#### 【相关主题】

无

## HI\_MAPI\_AEnc\_DeInit

#### 【描述】

去初始化音频编码通道。

#### 【语法】

```
HI_S32 HI_MAPI_AEnc_DeInit(HI_HANDLE AEncHdl)
```



【参数】

参数名称	描述	输入/输出
AEncHdl	音频编码通道句柄。 取值范围：[0, <a href="#">HI_AENC_MAX_CHN_NUM</a> )。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】

- 头文件：hi\_mapi\_aenc\_define.h、hi\_mapi\_aenc.h
- 库文件：libhi3518e\_mapi\_aenc.a

【注意】

无

【举例】

无

【相关主题】

[HI\\_MAPI\\_AEnc\\_Init](#)

## HI\_MAPI\_AEnc\_Start

【描述】

启动音频编码通道。

【语法】

```
HI_S32 HI_MAPI_AEnc_Start(HI_HANDLE AEncHdl)
```

【参数】

参数名称	描述	输入/输出
AEncHdl	音频编码通道句柄。 取值范围：[0, <a href="#">HI_AENC_MAX_CHN_NUM</a> )。	输入



【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无

【需求】

- 头文件：hi\_mapi\_aenc\_define.h、hi\_mapi\_aenc.h
- 库文件：libhi3518e\_mapi\_aenc.a

【注意】

无

【举例】

无

【相关主题】

[HI\\_MAPI\\_AEnc\\_Stop](#)

## HI\_MAPI\_AEnc\_Stop

【描述】

停止音频编码通道。

【语法】

```
HI_S32 HI_MAPI_AEnc_Stop(HI_HANDLE AEncHdl)
```

【参数】

参数名称	描述	输入/输出
AEncHdl	音频编码通道句柄。 取值范围：[0, <a href="#">HI_AENC_MAX_CHN_NUM</a> )。	输入

【返回值】

返回值	描述
0	成功。



返回值	描述
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】

- 头文件：hi\_mapi\_aenc\_define.h、hi\_mapi\_aenc.h
- 库文件：libhi3518e\_mapi\_aenc.a

【注意】

无

【举例】

无

【相关主题】

[HI\\_MAPI\\_AEnc\\_Start](#)

## HI\_MAPI\_AEnc\_RegisterCallback

【描述】

注册音频编码回调函数。

【语法】

```
HI_S32 HI_MAPI_AEnc_RegisterCallback(HI_HANDLE AEncHdl,  
HI\_AENC\_CALLBACK\_S *pstAencCB);
```

【参数】

参数名称	描述	输入/输出
AEncHdl	音频编码通道句柄。 取值范围：[0, <a href="#">HI_AENC_MAX_CHN_NUM</a> )。	输入
pstAencCB	音频编码回调函数结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。



#### 【芯片差异】

无。

#### 【需求】

- 头文件：hi\_mapi\_aenc\_define.h、hi\_mapi\_aenc.h
- 库文件：libhi3518e\_mapi\_aenc.a

#### 【注意】

- 音频编码通道最大支持五个不同的回调函数。
- 回调函数结构体内容相等时则判定为相同。

#### 【举例】

无

#### 【相关主题】

[HI\\_MAPI\\_AEnc\\_UnRegisterCallback](#)

## HI\_MAPI\_AEnc\_UnRegisterCallback

#### 【描述】

解注册回调函数结构体。

#### 【语法】

```
HI_S32 HI_MAPI_AEnc_UnRegisterCallback(HI_HANDLE AEncHdl,  
HI\_AENC\_CALLBACK\_S *pstAencCB)
```

#### 【参数】

参数名称	描述	输入/输出
AEncHdl	音频编码通道句柄。 取值范围：[0, <a href="#">HI_AENC_MAX_CHN_NUM</a> )。	输入
pstAencCB	音频编码回调函数结构体指针	输入

#### 【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

#### 【芯片差异】



无

【需求】

- 头文件：hi\_mapi\_aenc\_define.h、hi\_mapi\_aenc.h
- 库文件：libhi3518e\_mapi\_aenc.a

【注意】

重复解注册回调函数，返回失败。

【举例】

无

【相关主题】

[HI\\_MAPI\\_AEnc\\_RegisterCallback](#)

## HI\_MAPI\_AEnc\_BindACap

【描述】

音频编码通道绑定输入源 ACap。

【语法】

```
HI_S32 HI_MAPI_AEnc_BindACap(HI_HANDLE ACapHdl, HI_HANDLE AEncHdl);
```

【参数】

参数名称	描述	输入/输出
ACapHdl	音频采集通道号。 取值范围：[0, 2)	输入
AEncHdl	音频编码通道句柄。 取值范围：[0, <a href="#">HI_AENC_MAX_CHN_NUM</a> )。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无

【需求】





- 头文件：hi\_mapi\_aenc\_define.h、hi\_mapi\_aenc.h
- 库文件：libhi3518e\_mapi\_aenc.a

【注意】

无

【举例】

无

【相关主题】

[HI\\_MAPI\\_AEnc\\_UnBindACap](#)

## HI\_MAPI\_AEnc\_UnBindACap

【描述】

解绑定音频编码通道输入源 ACap。

【语法】

```
HI_S32 HI_MAPI_AEnc_UnBindACap(HI_HANDLE ACapHdl, HI_HANDLE AEncHdl);
```

【参数】

参数名称	描述	输入/输出
ACapHdl	音频采集通道号。 取值范围：[0, 2)	输入
AEncHdl	音频编码通道句柄。 取值范围：[0, <a href="#">HI_AENC_MAX_CHN_NUM</a> )。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】

- 头文件：hi\_mapi\_aenc\_define.h、hi\_mapi\_aenc.h
- 库文件：libhi3518e\_mapi\_aenc.a



【注意】

无

【举例】

无

【相关主题】

[HI\\_MAPI\\_AEnc\\_BindACap](#)

## HI\_MAPI\_Register\_ExtAudioEncoder

【描述】

注册外部音频编码器。

【语法】

```
HI_S32 HI_MAPI_Register_ExtAudioEncoder(HI\_MPP\_AENC\_ENCODER\_S\*  
pstAencEncoder, HI_HANDLE* ps32AEncoderHdl)
```

【参数】

参数名称	描述	输入/输出
pstAencEncoder	外部音频编码器参数	输入
ps32AEncoderHdl	外部音频编码器句柄指针，范围[0,20)	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无。

【需求】

- 头文件：[hi\\_mapi\\_aenc\\_define.h](#)、[hi\\_mapi\\_aenc.h](#)
- 库文件：[libhi3518e\\_mapi\\_aenc.a](#)

【注意】

- 编码器注册成功后，返回编码器句柄。
- 句柄大于等于零有效。



【举例】

请参见 [HI\\_MAPI\\_AEnc\\_Init](#) 的举例。

【相关主题】

[HI\\_MAPI\\_UnRegister\\_ExtAudioEncoder](#)

## HI\_MAPI\_UnRegister\_ExtAudioEncoder

【描述】

解注册外部编码器。

【语法】

```
HI_S32 HI_MAPI_UnRegister_ExtAudioEncoder(HI_HANDLE s32AEncoderHdl);
```

【参数】

参数名称	描述	输入/输出
ps32AEncoderHdl	外部音频编码器句柄指针，范围[0,20)	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为 <a href="#">错误码</a> 。

【芯片差异】

无

【需求】

- 头文件：hi\_mapi\_aenc\_define.h、hi\_mapi\_aenc.h
- 库文件：libhi3518e\_mapi\_aenc.a

【注意】

无

【举例】

无

【相关主题】

[HI\\_MAPI\\_Register\\_ExtAudioEncoder](#)



## 8.4 数据类型

音频编码相关数据类型定义如下：

- [HI\\_AENC\\_MAX\\_CHN\\_NUM](#)：定义音频编码通道的最大个数。
- [AENC\\_CHN\\_REGISTER\\_CALLBACK\\_MAX\\_COUNT](#)：定义音频编码通道注册回调函数的最大个数。
- [HI\\_MPP\\_AUDIO\\_FORMAT\\_E](#)：定义音频格式。
- [HI\\_MPP\\_AENC\\_ATTR\\_S](#)：定义音频编码参数结构体。
- [HI\\_MPP\\_AENC\\_STREAM\\_S](#)：定义音频编码码流信息。
- [PFN\\_AENC\\_DataProc](#)：定义音频编码数据处理函数。
- [HI\\_AENC\\_CALLBACK\\_S](#)：定义音频编码回调函数结构体。
- [HI\\_MPP\\_AENC\\_ENCODER\\_S](#)：定义音频编码器结构体。

### HI\_AENC\_MAX\_CHN\_NUM

**【说明】**

定义音频编码通道的最大个数。

**【定义】**

```
#define HI_AENC_MAX_CHN_NUM      2
```

**【注意事项】**

无

**【相关数据类型及接口】**

无

### AENC\_CHN\_REGISTER\_CALLBACK\_MAX\_COUNT

**【说明】**

定义音频编码通道注册回调函数的最大个数。

**【定义】**

```
#define AENC_CHN_REGISTER_CALLBACK_MAX_COUNT 5
```

**【注意事项】**

无

**【相关数据类型及接口】**

[HI\\_MAPI\\_AEnc\\_RegisterCallback](#)

### HI\_MPP\_AUDIO\_FORMAT\_E

**【说明】**



定义音频格式。

#### 【定义】

```
typedef enum hiMPP_AUDIO_FORMAT_E
{
    HI_MPP_AUDIO_FORMAT_G711A    = 1,    /* G.711 A          */
    HI_MPP_AUDIO_FORMAT_G711Mu   = 2,    /* G.711 Mu         */
    HI_MPP_AUDIO_FORMAT_ADPCM    = 3,    /* ADPCM            */
    HI_MPP_AUDIO_FORMAT_G726     = 4,    /* G.726            */
    HI_MPP_AUDIO_FORMAT_AMR      = 5,    /* AMR encoder format */
    HI_MPP_AUDIO_FORMAT_AMRDTX   = 6,    /* AMR encoder formant and VAD1
enable */
    HI_MPP_AUDIO_FORMAT_AAC      = 7,    /* AAC encoder       */
    HI_MPP_AUDIO_FORMAT_WAV      = 8,    /* WAV encoder       */
    HI_MPP_AUDIO_FORMAT_MP3      = 9,    /* MP3 encoder       */
    HI_MPP_AUDIO_FORMAT_BUTT
}HI_MPP_AUDIO_FORMAT_E;
```

#### 【成员】

无

#### 【注意事项】

无

#### 【相关数据类型及接口】

[HI\\_MAPI\\_AEnc\\_Init](#)

## HI\_MPP\_AENC\_ATTR\_S

#### 【说明】

定义音频编码参数结构体。

#### 【定义】

```
typedef struct hiMPP_AENC_ATTR_S
{
    HI_MPP_AUDIO_FORMAT_E enAencFormat; /**< audio encode format type*/
    HI_U32 u32PtNumPerFrm; /**<sampling point number per frame*/
    HI_VOID *pValue; /**pointer of Aenc attr*/
    HI_U32 u32Len; /**Aenc attr struct len*/
} HI_MPP_AENC_ATTR_S;
```

#### 【成员】

成员名称	描述
enAencFormat	编码类型。



成员名称	描述
u32PtNumPerFrm	采样点。
pValue	私有数据。
u32Len	私有数据长度。

【注意事项】

无

【相关数据类型及接口】

[HI\\_MAPI\\_AEnc\\_Init](#)[HI\\_MAPI\\_AEnc\\_Init](#)

## HI\_MPP\_AENC\_STREAM\_S

【说明】

定义音频编码码流信息。

【定义】

```
typedef struct hiMPP_AENC_STREAM_S
{
    HI_U8* pu8Addr; /**< the virtual address of stream */
    HI_U32 u32PhyAddr; /**< the physics address of stream */
    HI_U32 u32Len; /**< stream length, by bytes */
    HI_U64 u64TimeStamp; /**< frame time stamp*/
    HI_U32 u32Seq; /**< frame sequels,if stream is not a valid
frame,u32Seq is 0*/
} HI_MPP_AENC_STREAM_S;
```

【成员】

成员名称	描述
pu8Addr	码流数据虚拟地址。
u32PhyAddr	码流数据物理地址。
u32Len	码流数据长度。
u64TimeStamp	码流数据时间戳。
u32Seq	码流数据序列号。

【注意事项】

无



【相关数据类型及接口】

[PFN\\_AENC\\_DataProc](#)

## PFN\_AENC\_DataProc

【说明】

定义音频编码数据处理函数。

【定义】

```
typedef HI_S32 (*PFN_AENC_DataProc)(HI_HANDLE AencHdl,  
HI\_MPP\_AENC\_STREAM\_S* pStreamData, HI_VOID *pPrivateData);
```

【成员】

成员名称	描述
AencHdl	编码通道句柄。
pStreamData	编码码流数据。
pPrivateData	私有数据。

【注意事项】

无

【相关数据类型及接口】

[HI\\_AENC\\_CALLBACK\\_S](#)

## HI\_AENC\_CALLBACK\_S

【说明】

定义音频编码回调函数结构体。

【定义】

```
typedef struct hiAENC_CALLBACK_S  
{  
    PFN\_AENC\_DataProc pfnDataCB;  
    HI_VOID *pPrivateData;  
} HI_AENC_CALLBACK_S;
```

【成员】

成员名称	描述
pfnDataCB	数据处理函数。
pPrivateData	私有数据指针。



#### 【注意事项】

无。

#### 【相关数据类型及接口】

- [HI\\_MAPI\\_AEnc\\_RegisterCallback](#)
- [HI\\_MAPI\\_AEnc\\_UnRegisterCallback](#)

## HI\_MPP\_AENC\_ENCODER\_S

#### 【说明】

定义音频编码器结构体。

#### 【定义】

```
typedef struct hiMPP_AENC_ENCODER_S
{
    HI_MPP_AUDIO_FORMAT_E enType;
    HI_U32 u32MaxFrmLen;
    HI_CHAR aszName[16]; /* encoder type, be used to print proc
information */
    HI_S32 (*pfnOpenEncoder) (HI_VOID *pEncoderAttr, HI_VOID **ppEncoder);
/* pEncoder is the handle to control the encoder */
    HI_S32 (*pfnEncodeFrm) (HI_VOID *pEncoder, const HI_MPP_AUDIO_FRAME_S
*pstData, HI_U8 *pu8Outbuf, HI_U32 *pu32OutLen);
    HI_S32 (*pfnCloseEncoder) (HI_VOID *pEncoder);
} HI_MPP_AENC_ENCODER_S;
```

#### 【成员】

成员名称	描述
enType	编码类型。
u32MaxFrmLen	最大帧长度。
aszName	编码器名称。
pfnOpenEncoder	编码器打开回调函数。
pfnEncodeFrm	编码回调函数。
pfnCloseEncoder	编码器关闭回调函数。

#### 【注意事项】

无





【相关数据类型及接口】

[HI\\_MAPI\\_Register\\_ExtAudioEncoder](#)

8.5 错误码

音频编码 API 错误码如表 8-1 所示。

表8-1 音频编码 API 错误码

错误代码	宏定义	描述
0xA3058002	HI_ERR_MAPI_AENC_HANDLE_ILLEGAL	音频编码句柄无效
0xA3058003	HI_ERR_MAPI_AENC_ILLEGAL_PARAM	音频编码参数无效
0xA3058006	HI_ERR_MAPI_AENC_NULL_PTR	音频编码空指针错误
0xA3058010	HI_ERR_MAPI_AENC_NOT_INITED	音频编码通道未初始化



## 目 录

9 音频输出.....	9-1
9.1 概述.....	9-1
9.2 功能描述.....	9-1
9.3 API 参考 .....	9-1
9.4 数据类型.....	9-8
9.5 错误码.....	9-10



## 表格目录

表 9-1 AO API 错误码.....	9-10
-----------------------	------



# 9 音频输出

## 9.1 概述

AO (audio output) 用于解码回放时，将音频解码后的音频帧送给 AO，然后 AO 通过 codec 将数字信号转换为模拟信号输出声音。

## 9.2 功能描述

用户可以通过 AO 的接口设置输出音量大小，或者设置是否静音。音频解码和 AO 之间的通路连接。

## 9.3 API 参考

该功能模块为用户提供以下 MAPI：

- [HI\\_MAPI\\_AO\\_Init](#)：初始化一个 AO 处理单元。
- [HI\\_MAPI\\_AO\\_DeInit](#)：销毁一个 AO 处理单元。
- [HI\\_MAPI\\_AO\\_Start](#)：启动 AO，开始接收音频帧数据。
- [HI\\_MAPI\\_AO\\_Stop](#)：停止 AO 接收。
- [HI\\_MAPI\\_AO\\_SetVolume](#)：设置音频输出的音量。
- [HI\\_MAPI\\_AO\\_GetVolume](#)：获取音频输出的音量。
- [HI\\_MAPI\\_AO\\_Mute](#)：静音音频输出。
- [HI\\_MAPI\\_AO\\_unMute](#)：取消静音。

### HI\_MAPI\_AO\_Init

#### 【描述】

初始化一个 AO 处理单元。

#### 【语法】

```
HI_S32 HI_MAPI_AO_Init(HI_HANDLE AoHdl, HI_MPP_AO_ATTR_S* pstAoAttr);
```



【参数】

参数名称	描述	输入/输出
AoHdl	AO handle 号。 取值范围：0。	输入
pstAoAttr	AO 属性指针。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_ao\_define.h、hi\_mapi\_ao.h
- 库文件：libhi3518e\_mapi\_ao.a

【注意】

无

【举例】

无

【相关主题】

无

## HI\_MAPI\_AO\_DeInit

【描述】

去初始化一个 AO。

【语法】

```
HI_S32 HI_MAPI_AO_DeInit(HI_HANDLE AoHdl);
```

【参数】

参数名称	描述	输入/输出
AoHdl	AO handle 号。 取值范围：0。	输入



【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_ao\_define.h、hi\_mapi\_ao.h
- 库文件：libhi3518e\_mapi\_ao.a

【注意】

无

【举例】

无

【相关主题】

无

## HI\_MAPI\_AO\_Start

【描述】

启动一个 AO。

【语法】

```
HI_S32 HI_MAPI_AO_Start(HI_HANDLE AoHdl);
```

【参数】

参数名称	描述	输入/输出
AoHdl	AO handle 号。 取值范围：0。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，参见 <a href="#">错误码</a> 。

【需求】



- 头文件：hi\_mapi\_ao\_define.h、hi\_mapi\_ao.h
- 库文件：libhi3518e\_mapi\_ao.a

**【注意】**

AO 必须已初始化。

**【举例】**

无

**【相关主题】**

无

## HI\_MAPI\_AO\_Stop

**【描述】**

停止 AO。

**【语法】**

```
HI_S32 HI_MAPI_AO_Stop(HI_HANDLE AoHdl);
```

**【参数】**

参数名称	描述	输入/输出
AoHdl	AO handle 号。 取值范围：0。	输入

**【返回值】**

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

**【需求】**

- 头文件：hi\_mapi\_ao\_define.h、hi\_mapi\_ao.h
- 库文件：libhi3518e\_mapi\_ao.a

**【注意】**

AO 必须已经启动。

**【举例】**

无



【相关主题】

无

## HI\_MAPI\_AO\_SetVolume

【描述】

设置 AO 输出的音量。

【语法】

```
HI_S32 HI_MAPI_AO_SetVolume(HI_HANDLE AoHdl, HI_MPP_AUDIO_GAIN_S* pstVol);
```

【参数】

参数名称	描述	输入/输出
AoHdl	AO handle 号。 取值范围：0。	输入
pstVol	音频输出音量结构体。具体描述请参考“音频采集”章节。 S32Gain 取值范围：[-121, 6]。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_ao\_define.h、hi\_mapi\_ao.h
- 库文件：libhi3518e\_mapi\_ao.a

【注意】

无

【举例】

无

【相关主题】

无





## HI\_MAPI\_AO\_GetVolume

### 【描述】

获取 AO 输出的音量。

### 【语法】

```
HI_S32 HI_MAPI_AO_GetVolume(HI_HANDLE AoHdl, HI_MPP_AUDIO_GAIN_S* pstVol);
```

### 【参数】

参数名称	描述	输入/输出
AoHdl	AO handle 号。 取值范围：0。	输入
pstVol	音频输出音量结构体。	输出

### 【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

### 【需求】

- 头文件：hi\_mapi\_ao\_define.h、hi\_mapi\_ao.h
- 库文件：libhi3518e\_mapi\_ao.a

### 【注意】

无

### 【举例】

无

### 【相关主题】

无

## HI\_MAPI\_AO\_Mute

### 【描述】

设置音频输出静音。

### 【语法】

```
HI_S32 HI_MAPI_AO_Mute(HI_HANDLE AoHdl);
```



【参数】

参数名称	描述	输入/输出
AoHdl	AO handle 号。 取值范围：0。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_ao\_define.h、hi\_mapi\_ao.h
- 库文件：libhi3518e\_mapi\_ao.a

【注意】

无

【举例】

无

【相关主题】

无

## HI\_MAPI\_AO\_unMute

【描述】

取消音频输出的静音。

【语法】

```
HI_S32 HI_MAPI_AO_unMute(HI_HANDLE AoHdl);
```

【参数】

参数名称	描述	输入/输出
AoHdl	AO handle 号。 取值范围：0。	输入

【返回值】



返回值	描述
0	成功。
非 0	失败，请参见 <a href="#">错误码</a> 。

【需求】

- 头文件：hi\_mapi\_ao\_define.h、hi\_mapi\_ao.h
- 库文件：libhi3518e\_mapi\_ao.a

【注意】

无

【举例】

无

【相关主题】

[HI\\_MAPI\\_AO\\_Mute](#)

## 9.4 数据类型

AO 模块相关数据类型定义如下：

- [HI\\_MPP\\_AO\\_ATTR\\_S](#)：定义 AO 属性结构体。
- [HI\\_MPP\\_AO\\_INTF\\_TYPE\\_E](#)：定义 AO 输出接口类型枚举。

### HI\_MPP\_AO\_ATTR\_S

【说明】

定义 AO 属性的结构体。

【定义】

```
typedef struct hiMPP_AO_ATTR_S
{
    HI_MPP_AUDIO_SAMPLE_RATE_E  enSampleRate;
    HI_MPP_AUDIO_BITWIDTH_E     enBitwidth;
    HI_MPP_AUDIO_SOUND_MODE_E   enSoundMode;
    HI_MPP_AUDIO_MODE_E         enAudioMode;
    HI_U32                       u32PtNumPerFrm;
    HI\_MPP\_AO\_INTF\_TYPE\_E enAoIntfType;
} HI_MPP_AO_ATTR_S;
```

【成员】



成员名称	描述
enSampleRate	音频采样率。
enBitwidth	音频采样精度。
enSoundMode	音频输出声道模式。
enAudioMode	音频工作模式。
u32PtNumPerFrm	每帧的采样点个数。
enAoIntfType	音频输出接口类型。

【注意事项】

enSampleRate、enBitwidth、enSoundMode、enAudioMode 的具体描述请参考第 7 章“音频采集”章节。

【相关数据类型及接口】

[HI\\_MAPI\\_AO\\_Init](#)

## HI\_MPP\_AO\_INTF\_TYPE\_E

【说明】

定义 AO 输出接口类型枚举。

【定义】

```
typedef enum hiMPP_AO_INTF_TYPE_E
{
    HI_MPP_AO_INTF_TYPE_HDMI = 0,
    HI_MPP_AO_INTF_TYPE_ACODEC = 1,
    HI_MPP_AO_INTF_TYPE_BUTT
}HI_MPP_AO_INTF_TYPE_E;
```

【成员】

成员名称	描述
HI_MPP_AO_INTF_TYPE_HDMI	输出到 HDMI 接口。
HI_MPP_AO_INTF_TYPE_ACODEC	输出到 audio codec。

【注意事项】

无

【相关数据类型及接口】



## HI\_MPP\_AO\_ATTR\_S

### 9.5 错误码

音频输出 API 错误码如表 9-1 所示。

表9-1 AO API 错误码

错误代码	宏定义	描述
0xA3078001	HI_MAPI_ERR_AO_INVALID_DEVICE	音频输出设备号无效
0xA3078002	HI_MAPI_ERR_AO_INVALID_CHANNEL	音频输出通道号无效
0xA3078003	HI_MAPI_ERR_AO_ILLEGAL_PARAMETER	音频输出参数设置无效
0xA3078005	HI_MAPI_ERR_AO_NOT_ENABLED	音频输出设备或通道没使能
0xA3078006	HI_MAPI_ERR_AO_NULL_PTR	输出空指针错误
0xA3078007	HI_MAPI_ERR_AO_NOT_CONFIG	音频输出设备属性未设置
0xA3078008	HI_MAPI_ERR_AO_NOT_SUPPORT	操作不被支持
0xA3078009	HI_MAPI_ERR_AO_NOT_PERM	操作不允许
0xA307800C	HI_MAPI_ERR_AO_NOMEM	系统内存不足
0xA307800D	HI_MAPI_ERR_AO_NOBUF	音频输出缓存不足
0xA307800E	HI_MAPI_ERR_AO_BUF_EMPTY	音频输出缓存为空
0xA307800F	HI_MAPI_ERR_AO_BUF_FULL	音频输出缓存为满
0xA3078010	HI_MAPI_ERR_AO_SYS_NOTREADY	音频输出系统未初始化
0xA3078012	HI_MAPI_ERR_AO_BUSY	音频输出系统忙