AX VDEC API 文档

文档版本：**V2.2**

发布日期：**2024/7/15**

目 录

目 录 1
前 言 6
修订历史 7
1 概述 9
1.1 概述 9
1.2 功能描述 10
1.3 软件流程 16
2 API 简介 20
3 API 定义 22
AX_VDEC_Init 22
AX_VDEC_Deinit 24
AX_VDEC_CreateGrp 26
AX_VDEC_DestroyGrp 30
AX_VDEC_GetGrpAttr 31
AX_VDEC_SetGrpAttr 33
AX_VDEC_StartRecvStream 35
AX_VDEC_StopRecvStream 36
AX_VDEC_QueryStatus 37
AX_VDEC_ResetGrp 38
AX_VDEC_SetGrpParam 39
AX_VDEC_GetGrpParam 41
AX_VDEC_SelectGrp 43

AX_VDEC_SendStream 45
AX_VDEC_GetFrame 47
AX_VDEC_ReleaseFrame 49
AX_VDEC_GetUserData 51
AX_VDEC_ReleaseUserData 52
AX_VDEC_SetUserPic 53
AX_VDEC_EnableUserPic 55
AX_VDEC_DisableUserPic 57
AX_VDEC_SetDisplayMode 58
AX_VDEC_GetDisplayMode 60
AX_VDEC_AttachPool 61
AX_VDEC_DetachPool 63
AX_VDEC_JpegDecodeOneFrame 64
AX_VDEC_GetStreamBufInfo 66
AX_VDEC_ExtractStreamHeaderInfo 68
AX_VDEC_GetVuiParam 70
AX_VDEC_GetPicBufferSize 71
AX_JDEC_GetYuvBufferSize 72

4 数据结构 74

AX_VDEC_MAX_GRP_NUM 74
AX_VDEC_MAX_WIDTH 75
AX_VDEC_MAX_HEIGHT 76
AX_VDEC_MIN_WIDTH 77
AX_VDEC_MIN_HEIGHT 78
AX_JDEC_MAX_WIDTH 79
AX_JDEC_MAX_HEIGHT 80

AX_JDEC_MIN_WIDTH....81
AX_JDEC_MIN_HEIGHT....82
AX_MAX_VDEC_USER_DATA_SIZE....83
AX_MAX_JDEC_USER_DATA_SIZE....84
AX_MAX_VDEC_USER_DATA_CNT....85
AX_VDEC_GRP....86
AX_VDEC_MOD_ATTR_T....87
AX_VDEC_INPUT_MODE_E....88
AX_VDEC_GRP_ATTR_T....90
AX_VDEC_OUTPUT_ORDER_E....92
AX_VDEC_MODE_E....93
AX_VDEC_GRP_PARAM_T....94
AX_VDEC_RECV_PIC_PARAM_T....95
AX_VDEC_STREAM_T....96
AX_VDEC_DISPLAY_MODE_E....98
AX_VDEC_GRP_STATUS_T....99
AX_VDEC_DECODE_ERROR_T....101
AX_VDEC_GRP_SET_INFO_T....103
AX_VDEC_DEC_ONE_FRM_T....104
AX_VDEC_USERDATA_T....105
AX_VDEC_STREAM_BUF_INFO_T....106
AX_VDEC_VUI_PARAM_T....107
AX_VDEC_VUI_ASPECT_RATIO_T....108
AX_VDEC_VUI_TIME_INFO_T....110
AX_VDEC_VUI_VIDEO_SIGNAL_T....111
AX_VDEC_VUI_BITSTREAM_RESTRIC_T....113

AX_VDEC_USRPIC_T 114
AX_VDEC_BITSTREAM_INFO_T 115
5 错误码 116
6 调试信息 117
7 Dump Tool 124

权利声明

注意

您购买的产品、服务或特性等应受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。除非商业合同另有约定，本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。

前 言

适用产品

AX620E 系列产品（AX630C、AX620Q）

适读人群

- 软件开发工程师
- 技术支持工程师

符号与格式定义

符号/格式	说明
xxx	表示您可以执行的命令行。
☞ 说明/备注：	表示您在使用产品的过程中，我们向您说明的事项。
！ 注意：	表示您在使用产品的过程中，需要您特别注意的事项。

修订历史

文档版本	发布时间	修订说明
V1.0	2023/08/29	文档初版
V1.1	2023/10/25	删除 AX_VIDEO_FRAME_T 结构描述
V1.2	2023/11/24	● 删除 AX_PAYLOAD_TYPE_E 结构描述 ● 增加 AX_VDEC_GetPicBufferSize 描述
V1.3	2024/01/12	1、修改结构 AX_VDEC_MOD_ATTR_T 2、修改 JDEC 最大解码分辨率
V1.4	2024/01/31	**第 3 章 API 定义** 1、修改部分函数返回值描述 2、调整函数顺序
V1.5	2024/2/1	**第 3 章 API 定义** 修改部分函数返回值描述 **第 7 章 Dump Tool** 修改使用方法；
V1.6	2024/2/4	**第 4 章 数据结构** 修改 PTS 描述 **第 6 章 调试信息** 修改调试信息描述；
V1.7	2024/2/5	**第 3 章 API 定义** 新增 AX_VDEC_ExtractStreamHeaderInfo API 说明
V1.8	2024/2/19	**第 3 章 API 定义** AX_VDEC_Init、AX_VDEC_Deinit【注意事项】涉及修改，支持内部引用计数重复调用。
V1.9	2024/3/25	**第 3 章 API 定义**

文档版本	发布时间	修订说明
		1、修改 AX_VDEC_Init、AX_VDEC_Deinit 注意事项 2、修改 AX_VDEC_CreateGrp 注意事项
V2.0	2024/6/4	**第 4 章 数据结构** AX_VDEC_BITSTREAM_INFO_T 中增加解码 buf 个数描述。
V2.1	2024/6/28	**第 3 章 API 定义** 新增 AX_JDEC_GetYuvBufferSize 接口，针对 YUV444 等格式，增加 JDEC 解码输出 YUV buffer 大小计算说明。
V2.2	2024/7/15	**第 3 章 API 定义** 新增 AX_VDEC_CreateGrpEx 接口

1 概述

1.1 概述

视频解码(video decoder)模块简称 VDEC，负责对 H.264/JPEG/MJPEG 格式的数字压缩视频进行解压缩处理，从而得到 YUV 格式的图像。该模块支持多路解码。本文档主要介绍 VDEC 模块的工作流程，以及相关 API 接口的使用方法。

不同型号的芯片支持不同的解码规格。芯片支持的解码规格如下表所示。

表1-1 芯片解码规格

H.264					
Baseline Profile, Levels 1~5. 1	Main Profile, Levels 1~5. 1	High Profile, Levels 1-5.1	High 10 Profile, Levels 1-5.1	最大分辨率	最小分辨率
支持	支持	支持	支持	1920x1920	48x48

JEPG			
JPEG Baseline	Motion JPEG	最大分辨率	最小分辨率
支持	支持	16384x16384	48x48

Video 解码器 x1，支持 H.264，最大 1080P@60fps。

JPEG 解码器 x1，支持 MJPEG，最大 4k@30fps。

Video 和 JPEG 使用同一个解码器，分时复用完成各自解码。

Video 解码器不支持 slice 解码，不支持解码 interlace 格式码流，不支持 VP8/VP9/H265，不支持 MPEG4。

1.2 功能描述

码流提交方式

VDEC 模块目前支持几种码流发送方式：

- 帧模式，按帧发送方式(VIDEO_MODE_FRAME)：用户每次调用 API 向解码器发送一帧压缩码流。该方式所需的帧缓存少，解码延迟小，较适合实时解码的场景。如无特别需求，一般建议使用该模式解码。

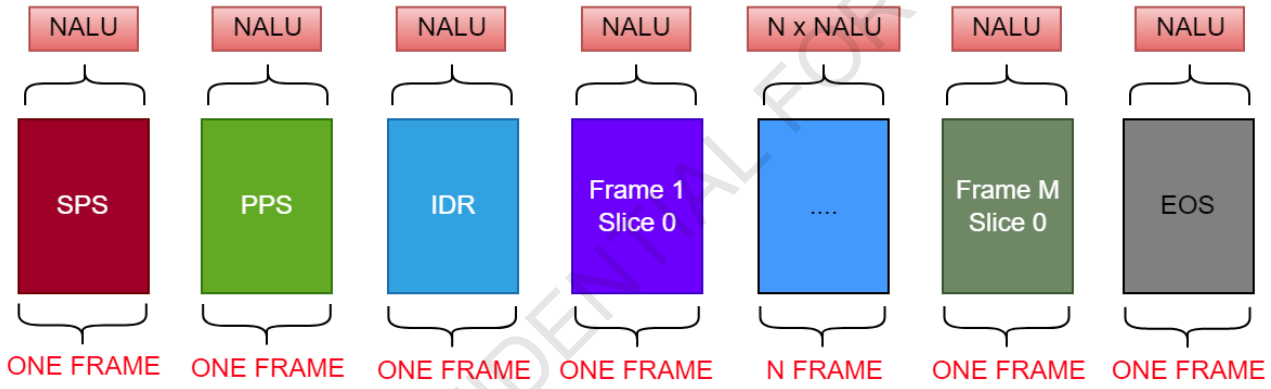


图1-1 按帧发送示意图

- 流模式，任意长度提交方式(INPUT_MODE_STREAM)：用户每次调用 API 向解码器提交任意长度的码流，由驱动软件完成对 NAL 边界的解析，最终在硬件上解码时仍是以帧为处理单位。

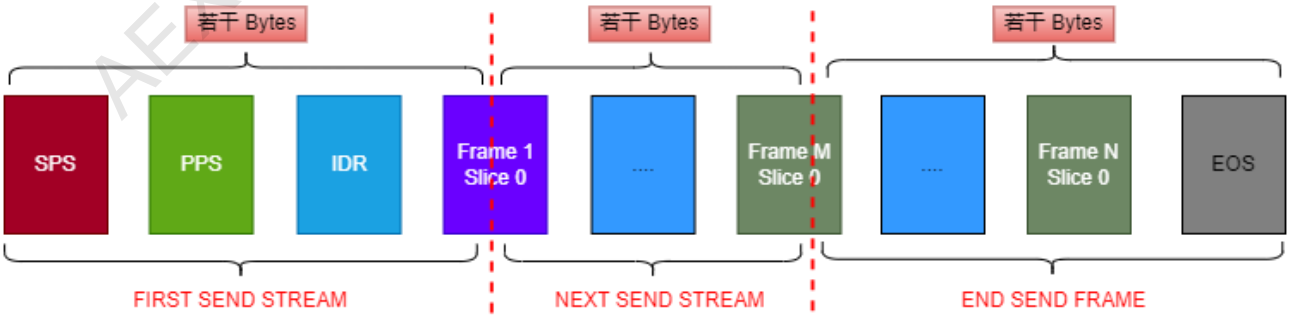


图1-2 按任意长度发送码流示意图

用户在创建解码通道的同时设置码流提交方式，详情可参考 AX_VDEC_CreateGrp()的接

口说明。

需要注意流模式送入码流时每次发送的数据帧数必须小于输入 fifo 深度的帧数(SDK 内部输入 fifo 深度是 32)，否则 AX_VDEC_SendStream()会返回失败。

解码输入通道组（Group）

VDEC 模块支持多个码流通过分时复用硬件的方式同时解码。对于解码器来说，每路输入码流称为一个通道(Group)，每个 Group 输入的码流经解码后可以输出同分辨率的 YUV 流。

播放时间戳(Presentation Timestamp, PTS)

在模式 VIDEO_MODE_FRAME 下发送码流时，解码输出的图像时间戳 PTS 为发送码流接口(AX_VDEC_SendStream)中用户送入的 PTS，解码器不会更改此值，解码输出 PTS 会和输入帧时一一对应。

！ 注意：

1、开始解码一个序列时，首先送入解码器的一般都是 SPS/PPS/SEI 等非数据帧，这些数据将被解码器内部消化，不会有对应的图像帧输出，自然也不会有 PTS 输出。如果用户对送帧和收帧进行计数，则可能出现收帧的次数少于送帧次数的情况。

2、不能出现 **PTS** 值为 **0** 和非 **0** 混合的情况。

图像输出顺序

当码流中存在 B 帧时，码流的解码顺序与图像的显示顺序可能是不一样的，所以 VDEC 模块支持两种图像输出方式：

- 按解码的先后顺序输出
- 按图像的正常显示顺序输出

默认按图像帧显示顺序输出，码流的输入（解码）帧顺序与图像帧的显示顺序不一样时，SDK 内部不会修改 PTS 值，送帧的时候需要保证 PTS 的准确性。

取帧方式

解码器的用户可选阻塞或非阻塞的方式接收解码图像。阻塞方式的典型用法是用户为每路码流创建一个取帧线程，在线程中调用 AX_VDEC_GetFrame()接口取帧。当解码路数很多时，阻塞方式线程开销较大，此时效率更高的方法是用一个线程处理所有解码通道的数据，此方法需要使用 AX_VDEC_SelectGrp()接口配合 AX_VDEC_GetFrame()接口使用，其中 AX_VDEC_SelectGrp()接口承担阻塞功能，并监控所有解码通道(非 link 模式的解码通道)的活动情况。当一个或多个解码通道有数据可用时，AX_VDEC_SelectGrp()立即退出阻塞并返回待处理的通道列表，然后遍历通道列表，调用 AX_VDEC_GetFrame()接口以非阻塞方式完成取帧。

当解码器内部没有申请到可用的输出帧缓存时，解码过程会暂停，当有空闲输出帧缓存可用时会自动恢复解码。如果用户为解码器配置的帧缓存总数不满足解码所需的最小数量要求，或者用户取帧后没有及时归还帧缓存，都可能导致解码器暂停工作。

解码能力(Specification)

VDEC 解码器的输出图像支持如下格式：

1. Color depth: 8-bit
2. Plane: NV12

JDEC 解码器的输出图像支持如下格式：

1. Color depth: 8-bit
2. Plane: NV12、YUV400、YUV422_SEMIPLANAR、YUV444_SEMIPLANAR

备注：JDEC 输出的 YUV 格式取决于输入的 jpeg 采样格式。如果输入的 jpeg 是 YUV444 采样，则输出的 YUV 格式就是 YUV444_SEMIPLANAR，输出 YUV 内存需要按照 width*height*3 申请。用户需确保整个业务过程中输出 YUV 内存配置满足最大需求场景。

链接(Link)与非链接(Unlink)

媒体业务的实际使用中，有用户主动控制数据流转和 SDK 软件内部自动数据流转这两种

使用场景的需求。

在用户主动控制数据流转的使用场景中，该模块的输入由上层应用软件负责推送，该模块的输出也由上层应用软件负责取出，这种场景我们称之为 Unlink 模式。

在 SDK 软件内部自动数据流转的使用场景中，是要把多种媒体模块的输入输出端彼此链接(Link)，前级输出的数据给后级作为输入，软件需维护好上下级数据链接的流转序列和流转性能，这种场景我们称之为 Link 模式。

Unlink 和 Link 两种使用场景的数据流控制方式不一样，涉及到的具体需求和控制逻辑也就不同。给 VDEC 输入的码流，无论是从网络传入，还是从存档文件传入，都由上层应用软件负责管理。上层应用软件通过 AX_VDEC_SendStream()接口推送码流到 VDEC 输入码流缓存。Unlink 使用场景下，VDEC 输出的帧，由上层应用软件通过 AX_VDEC_GetFrame() 接口负责取出，上层应用软件还须通过 AX_VDEC_ReleaseFrame()接口控制 VDEC 回收已被取出帧所占资源。

若前后级模块软件建立好 Link 关系，那么前后级就形成了计算机领域中生产者和消费者的逻辑关系；前级模块作为数据生产者不停的生产数据，后级模块作为消费者不停的消费数据。例如 VDEC Link VO，无需上层应用软件控制 VDEC 输出数据到后级，VDEC 解码出的帧会自动投递给 VO。

可以使用 AX_VDEC_SetDisplayMode()来使能 VDEC 阻塞式输出或丢帧式输出。

用户在创建解码通道的同时设置 Unlink 或 Link 模式，不支持两种模式并存，详情可参考 AX_VDEC_CreateGrp()的接口说明。

解码参数配置

- 各个结构体参数建议用户将不使用的值默认清零，或者在配置前通过获取相应参数的接口获取其值，SDK 内部会检查各参数是否满足合法范围或依赖条件。
- 输入码流缓存大小：最小必须能容纳码流中最占内存的一帧码流，否则解码会失败。建议最小设置为待解码码流一帧 YUV 帧缓存大小，以 YUV NV12 格式为例，此大小为 Width * Height * 3 / 2。
- 解码图像宽高必须是 16 像素对齐，比如 1920x1080 实际的解码 VB 是按照 1920x1088 申请

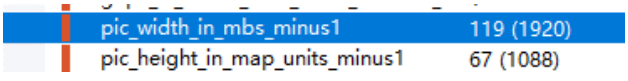
- 输入队列深度：无需用户设置。取决于输入码流缓存大小能容纳的帧数和 SDK 内部 FIFO 深度。
- 解码输出顺序：enOutputOrder 设为 AX_VDEC_OUTPUT_ORDER_DEC 时，解码顺序与输出显示顺序一致；而解码含 B 帧码流时，enOutputOrder 应设为 AX_VDEC_OUTPUT_ORDER_DISP，解码顺序与输出显示顺序不一致
- 输出帧缓存个数：帧缓存最小个数为码流 DPB 个数+1；若后接 VO 或 IVPS 等模块，最小个数需在前述基础上再+1。
- 输出帧缓存个数最大不超过 AX_VDEC_MAX_FRAME_BUF_CNT。

软件多进程

1. 支持多进程(一个通道只能被一个进程管理)，不支持跨进程读写同一个通道。
2. 每个进程都不允许调用 AX_POOL_Exit，也不需要调用。
3. 每个进程使用 VDEC 的 group 必须不同，否则不能使用 VDEC 多进程, vdec_sample 提供了“--uStartGrpId”作为指定不同起始 group id，方便测试。
4. common pool 属于所有进程共享，多进程情况下只有一个进程可以创建 common pool，其他进程不允许创建，一定要所有其他进程都退出，创建 common pool 的进程才能重启。
5. 推荐多进程内存足够情况下使用 private pool 和 user pool，如果内存有限，可以独立一个进程创建 common pool，其他进程可直接使用 common pool。

通道解码能力

1. AX_VDEC_GRP_ATTR_T 中 u32PicWidth 与 u32PicHeight 配置指当前解码通道所能解码的最大分辨率，用于分配参考帧缓存的计算。H264 协议编码是按照整数个 macro block 解码，所以码流解码出来也是整数个宏块像素，即宽高是 16 像素对齐，配置时需要注意对齐。



例如，用户创建一个 u32PicWidth 为 1920，u32PicHeight 为 1088 的解码通道，内存足够的情况下，那么当前通道可以解码 1920x1088 内的任何码流，此时注意输出 buffer 配置也

足够。

2. 当前 SDK 内部分配解码 VB 是按照码流实际的大小进行分配，当 u32PicWidth 与 u32PicHeight 设置与实际的码流分辨率不一致时，只要 VB pool 能够申请到 VB 也能正常解码。

1.3 软件流程

VDEC 的软件解码处理流程图如下所示。

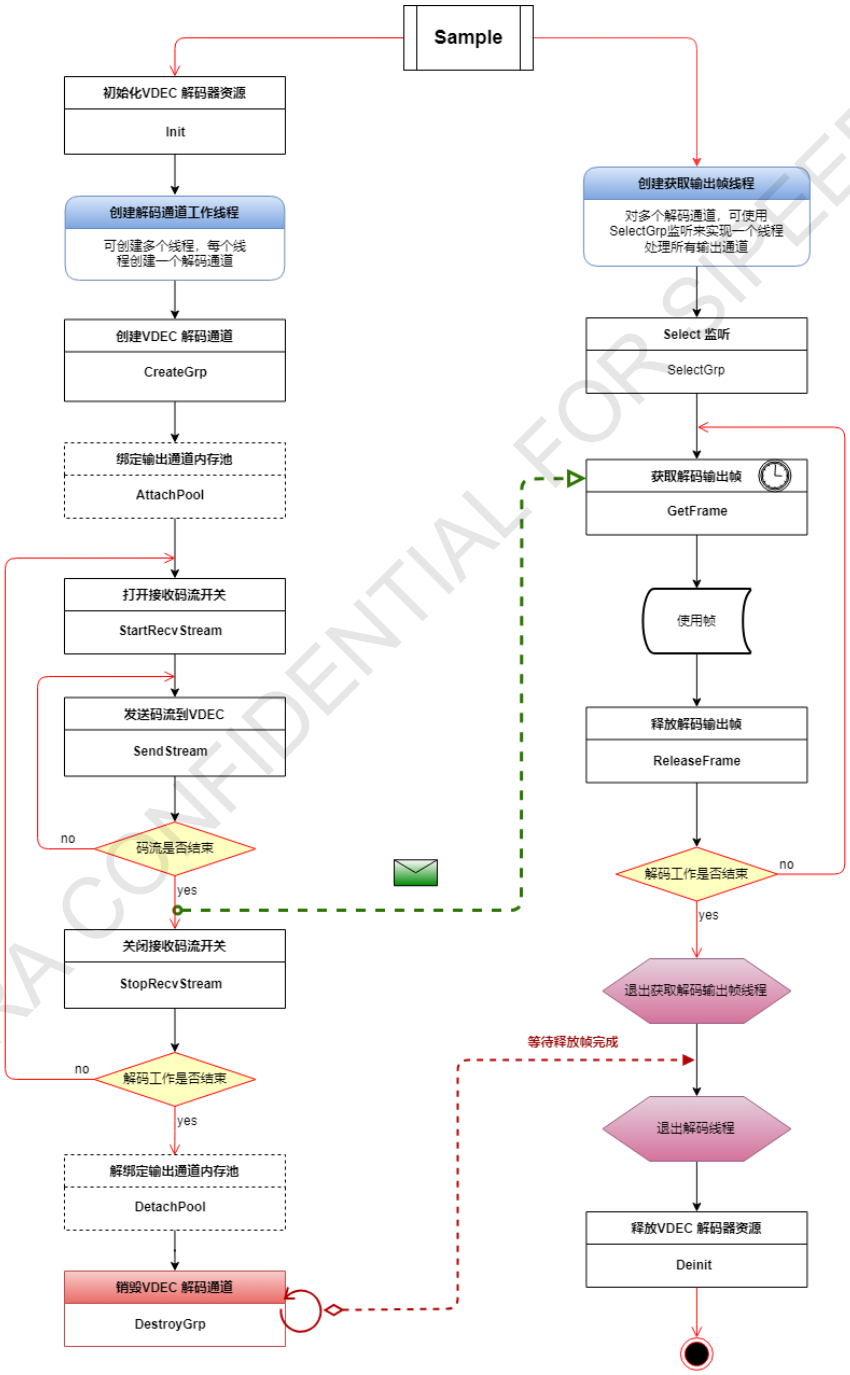


图1-3 Unlink 模式的软件流程

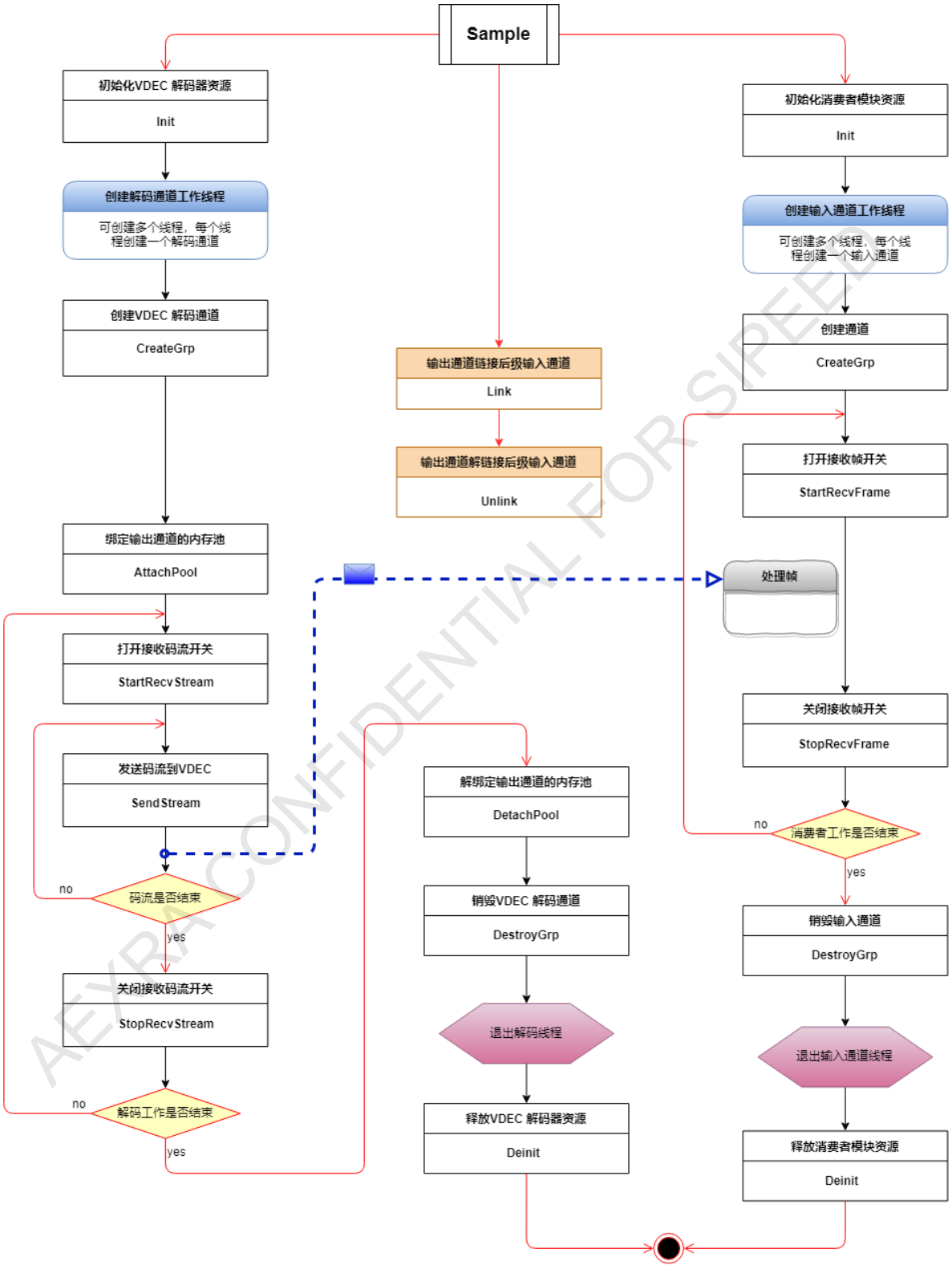


图1-4 Link 模式的软件流程

本芯片 VDEC 硬件不支持输入码流缓存环绕(Ring buffer)，SDK 支持用户配置输入码流缓存大小。在创建通道时，输入码流缓存大小为 u32StreamBufSize。VDEC 硬件不支持 Slice 解码，所以 u32StreamBufSize 最小必须能容纳码流中最占内存的一帧码流，否则解码会失败。u32StreamBufSize 最小值请参看上文参数配置章节。

VDEC 使用帧模式送流的处理流程图如下所示。

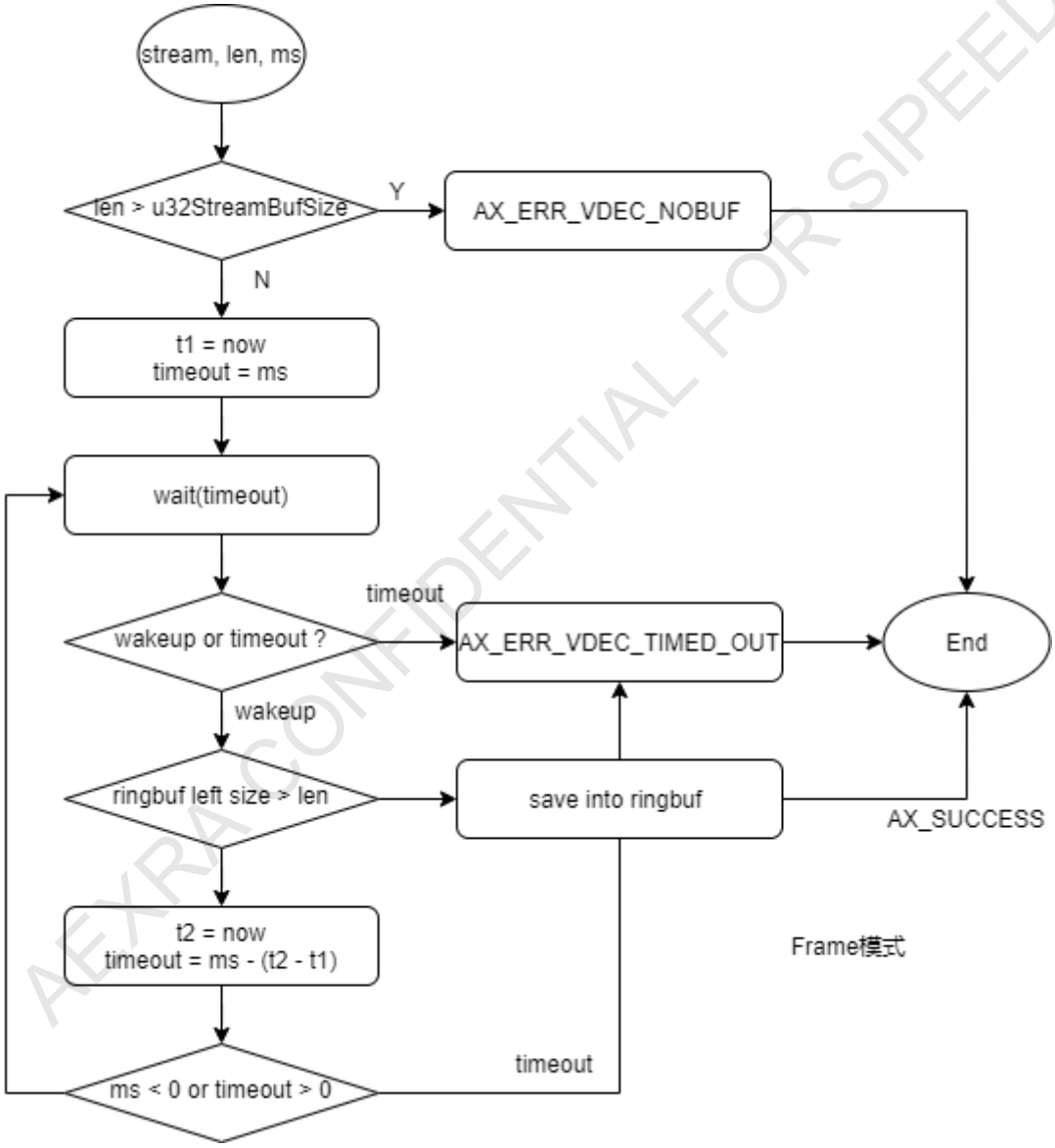


图1-5 帧模式送流的软件流程

VDEC 通道的软件状态流程图如下所示。典型解码流程包括：初始化解码器硬件及模块公共软件资源，创建解码实例，配置解码通道参数，配置协议参数，设置通道模式，开始发送码

流解码，解码完成停止发送码流，注销解码实例。

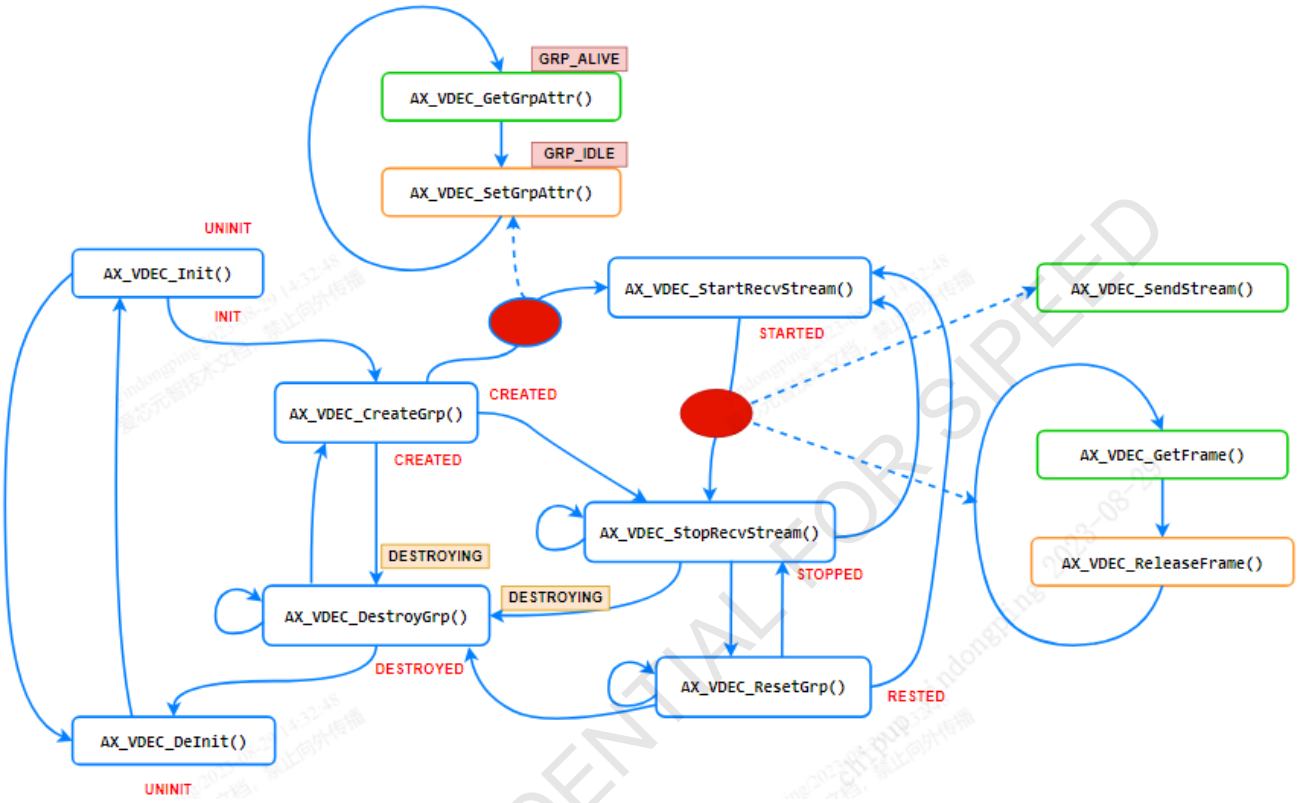


图1-6 解码通道状态流转

2 API 简介

解码模块提供的 API 接口如下：

- AX_VDEC_Init：初始化解码硬件资源。
- AX_VDEC_Deinit：解码结束释放解码硬件资源。
- AX_VDEC_CreateGrp：创建解码输入通道。
- AX_VDEC_DestroyGrp：销毁解码通道。
- AX_VDEC_GetGrpAttr：获取解码通道属性。
- AX_VDEC_SetGrpAttr：设置解码通道属性。
- AX_VDEC_StartRecvStream：触发解码通道开始接收码流数据。
- AX_VDEC_StopRecvStream：解码通道停止接收码流数据。
- AX_VDEC_QueryStatus：查询解码通道解码工作相关数据。
- AX_VDEC_ResetGrp：复位解码通道。
- AX_VDEC_SetGrpParam：设置解码通道参数。
- AX_VDEC_GetGrpParam：获取解码通道参数。
- AX_VDEC_SelectGrp：查询所有创建通道中是否有通道有帧输出。
- AX_VDEC_SendStream：发送码流数据到解码通道。
- AX_VDEC_GetFrame：获取解码通道解码后的图像。
- AX_VDEC_ReleaseFrame：释放解码图像缓存。
- AX_VDEC_GetUserData：获取视频解码通道的用户数据。
- AX_VDEC_ReleaseUserData：释放视频解码通道的用户数据。
- AX_VDEC_SetUserPic：设置输出通道的用户图片属性。

- AX_VDEC_EnableUserPic：使能输出通道插入用户图片。
- AX_VDEC_DisableUserPic：禁止使能输出通道插入用户图片。
- AX_VDEC_SetDisplayMode：设置显示模式参数。
- AX_VDEC_GetDisplayMode：获取显示模式参数。
- AX_VDEC_AttachPool：将解码通道绑定到某个缓存 VB 池中。
- AX_VDEC_DetachPool：将解码通道从某个缓存 VB 池中解绑定。
- AX_VDEC_JpegDecodeOneFrame：获取解码输出通道属性参数。
- AX_VDEC_ExtractStreamHeaderInfo：获取码流 I 帧头中的某些信息。
- AX_VDEC_GetStreamBufInfo：获取解码通道输入码流缓存信息。
- AX_VDEC_GetVuiParam：获取解码通道码流 VUI 信息。
- AX_VDEC_GetPicBufferSize：获取 VDEC 图像 buffer 的大小。
- AX_JDEC_GetYuvBufferSize：获取 JDEC 输出 YUV buffer 的大小。

3 API 定义

AX_VDEC_Init

【描述】

初始化解码硬件资源及模块公共资源。

本芯片 VDEC 和 JDEC 共享硬件模块。

【语法】

AX_S32 AX_VDEC_Init(const AX_VDEC_MOD_ATTR_T *pstModAttr);

【参数】

参数名称	描述	输入/输出
pstModAttr	模块属性参数的结构体指针(预留参数)。	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 启动 VDEC 必须首先执行 AX_VDEC_Init()，否则 VDEC 其他 API(AX_VDEC_Deinit 除外)均返回错误码

AX_ERR_VDEC_NOT_INIT。

- 传入参数 pstModAttr 为预留参数，AX_VDEC_Init 内部使用默认值。
- 如果内核驱动未加载或硬件初始化失败，返回 AX_ERR_VDEC_SYS_NOTREADY。
- 如果模块公共资源初始化失败，返回 AX_ERR_VDEC_RUN_ERROR。
- 对于多线程场景，允许重复调用该接口。只有第一次调用时会执行实际初始化操作，后续调用不返回错误但也不做任何实际操作。
- 对于多进程场景，每个进程都需要分别调用该接口，从而对进程相关资源做初始化。

AX_VDEC_Deinit

【描述】

解码结束释放解码硬件及模块公共资源。

【语法】

AX_S32 AX_VDEC_Deinit(AX_VOID);

【参数】

无

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 如果还有通道没有销毁，返回错误码 AX_ERR_VDEC_NOT_PERM。
- 如果释放资源失败，该返回值为错误码 AX_ERR_VDEC_UNKNOWN。
- 对于多线程场景，允许重复调用该接口。如果用户多次调用 AX_VDEC_Init，则只有在最后一次调用 AX_VDEC_Deinit 时才会执行实际的反初始化操作。
- 如果用户没有调用 AX_VDEC_Init 初始化，调用 AX_VDEC_Deinit 时会直接返回 AX_SUCCESS
- 对于多进程场景，每个进程都需要分别调用该接口，从而对进程相关资源做反初始化。

- 反初始化真正执行后，VDEC 其他 API(除了 AX_VDEC_Init())无法再执行成功，需要再次执行 AX_VDEC_Init 打开 VDEC。

AX_VDEC_CreateGrp

【描述】

创建解码通道。

【语法】

AX_S32 AX_VDEC_CreateGrp(AX_VDEC_GRP VdGrp, const AX_VDEC_GRP_ATTR_T *pstGrpAttr);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号。 取值范围：[0, AX_VDEC_MAX_GRP_NUM)。	输入
pstGrpAttr	解码通道属性参数的结构体指针。	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 如果参数 pstGrpAttr 为空，返回错误码 AX_ERR_VDEC_NULL_PTR。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 如果初始化通道实例失败，该返回值为错误码 AX_ERR_VDEC_RUN_ERROR。

- 创建解码通道前必须保证该通道未被创建，否则返回错误码 AX_ERR_VDEC_EXIST。
- 通道的 enCodecType 必须为 H264/JPEG/MJPEG，否则返回错误码 AX_ERR_VDEC_NOT_SUPPORT。
- 如果分配内存失败，该返回值为错误码 AX_ERR_VDEC_NOMEM。
- 创建通道的最大宽高参数若为 0，则默认使用码流的实际宽高创建解码器；若不为 0，数值范围需在硬件支持的范围内，否则返回错误码 AX_ERR_VDEC_NOT_SUPPORT。

AX_VDEC_CreateGrpEx

【描述】

查询可用 GrpId 并创建解码通道。

【语法】

AX_S32 AX_VDEC_CreateGrp(AX_VDEC_GRP *VdGrp, const AX_VDEC_GRP_ATTR_T *pstGrpAttr);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号指针。	输出
pstGrpAttr	解码通道属性参数的结构体指针。	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 当没用可用通道，返回错误码 AX_ERR_VDEC_NO_AVAILABLE_GRP。
- 如果参数 VdGrp 为空，返回错误码 AX_ERR_VDEC_NULL_PTR。
- 如果参数 pstGrpAttr 为空，返回错误码 AX_ERR_VDEC_NULL_PTR。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 如果初始化通道实例失败，该返回值为错误码 AX_ERR_VDEC_RUN_ERROR。

- 创建解码通道前必须保证该通道未被创建，否则返回错误码 AX_ERR_VDEC_EXIST。
- 通道的 enCodecType 必须为 H264/JPEG/MJPEG，否则返回错误码 AX_ERR_VDEC_NOT_SUPPORT。
- 如果分配内存失败，该返回值为错误码 AX_ERR_VDEC_NOMEM。
- 创建通道的最大宽高参数若为 0，则默认使用码流的实际宽高创建解码器；若不为 0，数值范围需在硬件支持的范围内，否则返回错误码 AX_ERR_VDEC_NOT_SUPPORT。

AX_VDEC_DestroyGrp

【描述】

销毁解码通道。

【语法】

AX_S32 AX_VDEC_DestroyGrp(AX_VDEC_GRP VdGrp);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号。 取值范围：[0, AX_VDEC_MAX_GRP_NUM)。	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。
- 每个输出通道获取帧数必须都归还，即 AX_VDEC_GetFrame 数与 AX_VDEC_ReleaseFrame 数相等，否则返回 AX_ERR_VDEC_BUSY。

AX_VDEC_GetGrpAttr

【描述】

获取通道属性。

【语法】

AX_S32 AX_VDEC_GetGrpAttr(AX_VDEC_GRP VdGrp, AX_VDEC_GRP_ATTR_T *pstGrpAttr);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号。 取值范围：[0, AX_VDEC_MAX_GRP_NUM)。	输入
pstGrpAttr	解码通道属性参数的结构体指针。	输出

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 通道参数指针 pstGrpAttr 不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。

- 保证通道已经被创建，否则返回错误码 AX_ERR_VDEC_UNEXIST。

AX_VDEC_SetGrpAttr

【描述】

设置通道属性。

【语法】

AX_S32 AX_VDEC_SetGrpAttr(AX_VDEC_GRP VdGrp, const AX_VDEC_GRP_ATTR_T *pstGrpAttr);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号。 取值范围：[0, AX_VDEC_MAX_GRP_NUM)。	输入
pstGrpAttr	解码通道属性参数的结构体指针。	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 通道参数指针 pstGrpAttr 不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。

- 保证通道已经被创建，否则返回错误码 AX_ERR_VDEC_UNEXIST。
- 静态属性参数不允许设置，否则返回错误码 AX_ERR_VDEC_NOT_PERM。
- 通道的最大宽高参数若为 0，则默认使用硬件最大支持的宽高；若不为 0，数值范围需在硬件支持的范围内，否则返回错误码 AX_ERR_VDEC_NOT_SUPPORT。

AX_VDEC_StartRecvStream

【描述】

触发解码通道开始接收码流数据，并开始解码。

【语法】

AX_S32 AX_VDEC_StartRecvStream(AX_VDEC_GRP VdGrp, const AX_VDEC_RECV_PIC_PARAM_T *pstRecvParam);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号。	输入
pstRecvParam	解码接收参数的结构体指针	输入(当前无效，可传入空指针)

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。

AX_VDEC_StopRecvStream

【描述】

解码通道停止接收码流数据。

【语法】

AX_S32 AX_VDEC_StopRecvStream(AX_VDEC_GRP VdGrp);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。
- 解码通道停止接收码流数据，但不停止解码，若需要立即停止解码，可调用 AX_VDEC_ResetGrp 或 AX_VDEC_DestroyGrp。

AX_VDEC_QueryStatus

【描述】

获取通道状态。

【语法】

AX_S32 AX_VDEC_QueryStatus(AX_VDEC_GRP VdGrp, AX_VDEC_GRP_STATUS_T *pstGrpStatus);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入
pstGrpStatus	解码通道状态的结构体指针。	输出

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 通道状态参数指针 pstGrpStatus 不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。

AX_VDEC_ResetGrp

【描述】

复位解码通道。

【语法】

AX_S32 AX_VDEC_ResetGrp(AX_VDEC_GRP VdGrp);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。
- 复位前必须停止接收码流，否则返回错误码 AX_ERR_VDEC_NOT_PERM。
- 每个输出通道获取帧数必须都归还，即 AX_VDEC_GetFrame 数与 AX_VDEC_ReleaseFrame 数相等，否则返回 AX_ERR_VDEC_BUSY。

AX_VDEC_SetGrpParam

【描述】

设置通道参数。

【语法】

AX_S32 AX_VDEC_SetGrpParam(AX_VDEC_GRP VdGrp, const AX_VDEC_GRP_PARAM_T *pstGrpParam);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入
pstGrpParam	输入通道参数的结构体指针。	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。
- 该接口可设置通道的一些高级属性，这些属性的默认值见数据类型 AX_VDEC_GRP_PARAM_T 的说明。

- 通道参数指针 pstGrpParam 不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。
- 各项参数如超过合法范围，会返回 AX_ERR_VDEC_ILLEGAL_PARAM 的错误码，各参数范围见数据类型 AX_VDEC_GRP_PARAM_T 的说明。

AX_VDEC_GetGrpParam

【描述】

获取通道参数。

【语法】

AX_S32 AX_VDEC_GetGrpParam(AX_VDEC_GRP VdGrp, AX_VDEC_GRP_PARAM_T *pstGrpParam);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入
pstGrpParam	输入通道参数的结构体指针。	输出

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。
- 通道参数指针 pstGrpParam 不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。
- 各项参数如超过合法范围，会返回 AX_ERR_VDEC_ILLEGAL_PARAM 的错误码，各参

数范围见数据类型 AX_VDEC_GRP_PARAM_T 的说明。

AX_VDEC_SelectGrp

【描述】

查询所有已创建的通道中是否有通道有帧输出。

【语法】

AX_S32 AX_VDEC_SelectGrp(AX_VDEC_GRP_SET_INFO_T *pstGrpSet, AX_S32 s32MilliSec);

【参数】

参数名称	描述	输入/输出
pstGrpSet	有帧的通道集合信息结构体指针。	输出
s32MilliSec	获取用户数据超时参数。 取值范围： -1：阻塞。 0：非阻塞。 正值：超时时间，没上限值，以 ms 为单位动态属性。	输入

【返回值】

返回值	描述
0	成功
非 0	失败，返回错误码

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。

- 输出参数 pstGrpSet 指向的地址不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。
- 通常在一个线程里调用 AX_VDEC_SelectGrp，再调用 AX_VDEC_GetFrame 获取查到的所有通道的输出。
- 若 AX_VDEC_SelectGrp 与 AX_VDEC_GetFrame 在不同线程中调用，会有时序同步问题，即 AX_VDEC_SelectGrp 返回的信息可能已经被另一线程调用 AX_VDEC_GetFrame 改变了

AX_VDEC_SendStream

【描述】

发送码流数据到解码通道。

【语法】

AX_S32 AX_VDEC_SendStream(AX_VDEC_GRP VdGrp, const AX_VDEC_STREAM_T *pstStream, AX_S32 s32MilliSec);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入
pstStream	码流结构体指针	输入
s32MilliSec	超时选项，取值范围： -1：阻塞方式 0：非阻塞方式 正值：超时时间，没有上限值，以 ms 为单位	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。

- 码流结构体指针不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 发送数据前应已经调用 AX_VDEC_StartRecvStream() 接口启动接收码流，否则返回错误码 AX_ERR_VDEC_NOT_PERM。
- 发送数据前应保证通道已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。
- 如果发送码流长度为非法值，返回错误码 AX_ERR_VDEC_NOT_PERM。
- 对于 H264 协议可能返回以下错误码：
  - 如果是不支持的码流会返回 AX_ERR_VDEC_NOT_SUPPORT；
  - 如果发生解码超时则会返回错误码 AX_ERR_VDEC_TIMED_OUT；
  - 如果送入的码流有错误，则会返回错误码 AX_ERR_VDEC_STRM_ERROR。
- 在发送完所有码流后，用户可以发送一个 bEndStream 为 1 的空码流包，表示当前码流已结束，解码器会把所有码流全部解完并输出全部图像。除此之外，其他情况应该把 bEndStream 置为 0。
- 发送数据与获取帧数据是异步方式处理，发送数据成功不代表解码完成。
- AX_VDEC_SendStream 与 AX_VDEC_GetFrame 可以在一个线程中先后运行，也可以分别在两个线程中运行。

AX_VDEC_GetFrame

【描述】

获取解码后输出通道的图像。

【语法】

AX_S32 AX_VDEC_GetFrame(AX_VDEC_GRP VdGrp, AX_VIDEO_FRAME_INFO_T *pstFrameInfo, AX_S32 s32MilliSec);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入
pstFrameInfo	获取解码图像信息结构体指针。	输出
s32MilliSec	超时选项，取值范围： -1：阻塞 0：非阻塞 正值：超时时间，没有上限值，以 ms 为单位	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。

- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 结构体指针 pstFrameInfo 不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。如果在获取图像的过程中销毁通道，会立刻返回错误码 AX_ERR_VDEC_UNEXIST。
- 在以非阻塞方式获取解码图像时，如果缓冲区内没有图像，会立刻返回错误码 AX_ERR_VDEC_BUF_EMPTY。
- 若 AX_VDEC_GetFrame 返回值为 AX_SUCCESS，则获取到的帧必须使用 AX_VDEC_ReleaseFrame 进行释放，否则可能造成内存泄露或销毁通道失败。
- 若 AX_VDEC_GetFrame 返回值为 AX_ERR_VDEC_FLOW_END，此时 pstFrameInfo 返回 NULL，无须使用 AX_VDEC_ReleaseFrame 进行释放。

【备注】

AX_VIDEO_FRAME_INFO_T 见文档《50 - AX 公共数据结构文档.docx》

AX_VDEC_ReleaseFrame

【描述】

释放解码输出通道图像缓存。

【语法】

AX_S32 AX_VDEC_ReleaseFrame(AX_VDEC_GRP VdGrp, const AX_VIDEO_FRAME_INFO_T *pstFrameInfo);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入
pstFrameInfo	解码图像信息	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 结构体指针 pstFrameInfo 不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。

- 结构体指针 pstFrameInfo 指向的数据必须是由 AX_VDEC_GetFrame 获取到，且无任何修改，否则返回错误码 AX_ERR_VDEC_ILLEGAL_PARAM。
- 支持 AX_VDEC_GetFrame 连续获取多帧，AX_VDEC_ReleaseFrame 乱序释放。

【备注】

AX_VIDEO_FRAME_INFO_T 见文档《50 - AX 公共数据结构文档.docx》

AX_VDEC_GetUserData

【描述】

获取码流 userData 信息， 对于 H264，支持获取 SEI 信息，最大可获取长度为 2048 字节；对于 JPEG/MJPG，支持获取 COM(FFFE 开头)信息，最大获取长度为 65533 字节(0xFFFF – 2)。

【语法】

AX_S32 AX_VDEC_GetUserData(AX_VDEC_GRP VdGrp, AX_VDEC_USERDATA_T *pstUserData);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入
pstUserData	userData 信息结构体指针。	输出

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 结构体指针 pstUserData 不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。如果在获取 userData 的过程中销毁通道，会立刻返回错误码 AX_ERR_VDEC_UNEXIST。

AX_VDEC_ReleaseUserData

【描述】

此接口内部无释放逻辑，仅作接口保留，无需调用。

【语法】

AX_S32 AX_VDEC_ReleaseUserData (AX_VDEC_GRP VdGrp, const AX_VDEC_USERDATA_T *pstUserData);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入
pstUserData	userData 信息结构体指针。	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 结构体指针 pstUserData 不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。如果在获取 userData 的过程中销毁通道，会立刻返回错误码 AX_ERR_VDEC_UNEXIST。

AX_VDEC_SetUserPic

【描述】

设置用户图片插入属性。

【语法】

AX_S32 AX_VDEC_SetUserPic (AX_VDEC_GRP VdGrp, const AX_VDEC_USRPIC_T *pstUsrPic);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入
pstUserData	用户图片属性结构指针，结构体实参使用前需先清 0。	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 如果 pstUsrPic 指针为空，返回错误码 AX_ERR_VDEC_NULL_PTR。
- pstUsrPic 指向结构体 stFrmInfo 应具备有效的地址、宽度、高度、跨度、格式等信息。
- 目前只支持设置 8bit YVU420 格式不压缩的用户图片。

- 插入图片后禁止使能用户图片，再送入新码流时，不能希望马上开始正确解码，必须等到下一个 I 帧到来才能开始正确解码。

AX_VDEC_EnableUserPic

【描述】

使能用户图片插入。

【语法】

AX_S32 AX_VDEC_EnableUserPic (AX_VDEC_GRP VdGrp);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 如果通道未停止接收码流，返回错误码 AX_ERR_VDEC_NOT_PERM
- 如果用户图片属性未设置，返回错误码 AX_ERR_VDEC_ILLEGAL_PARAM。
- 调用 AX_VDEC_EnableUserPic 之前，必须调用 AX_VDEC_DisableUserPic 禁止使能插入用户图片。

- 重复使能插入用户图片返回成功，但不会再次插入用户图片。

AX_VDEC_DisableUserPic

【描述】

禁止使能插入用户图片。

【语法】

AX_S32 AX_VDEC_DisableUserPic (AX_VDEC_GRP VdGrp);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。

AX_VDEC_SetDisplayMode

【描述】

设置显示模式。

【语法】

AX_S32 AX_VDEC_SetDisplayMode(AX_VDEC_GRP VdGrp, AX_VDEC_DISPLAY_MODE_E enDisplayMode);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入
enDisplayMode	显示模式枚举。	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。
- 预览模式(AX_VDEC_DISPLAY_MODE_PREVIEW)：预览模式下 VDEC 链接的直接后级模块(比如 IVPS)以非阻塞方式接收解码图像，即当 IVPS 的图像 Buffer 满时（解码帧

存个数比 IVPS 缓存队列个数多），VDEC 会主动丢弃解码帧，以达到不反压 VDEC 解码的目的，实现实时预览。需要注意的是，当解码帧存个数比 IVPS 缓存队列个数少时，即使开启预览模式，IVPS 还是会反压解码。

- 回放模式(AX_VDEC_DISPLAY_MODE_PLAYBACK)：回放模式下 VDEC 链接的直接后级模块（比如 IVPS）以阻塞方式接收解码图像，即当 IVPS 的图像 Buffer 满时，拒绝接收 VDEC 发送过来的图像，VDEC 发现当前图像发送失败后启动图像重新发送机制，直到图像发送成功为止。回放模式下 VDEC 链接的直接后级模块能够反压 VDEC 解码，以达到不丢弃任何一帧解码图像的回放效果。

AX_VDEC_GetDisplayMode

【描述】

获取显示模式参数。

【语法】

AX_S32 AX_VDEC_GetDisplayMode(AX_VDEC_GRP VdGrp, AX_VDEC_DISPLAY_MODE_E *penDisplayMode);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入
penDisplayMode	显示模式枚举指针。	输出

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。

AX_VDEC_AttachPool

【描述】

将解码通道绑定到某个缓存 VB 池中。

【语法】

AX_S32 AX_VDEC_AttachPool(AX_VDEC_GRP VdGrp, AX_POOL PoolId);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号。 取值范围：[0, AX_VDEC_MAX_GRP_NUM)。	输入
PoolId	缓存池 Pool ID	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 必须保证已创建 VB 池的 Pool Id 有效，如果 VB 池 ID 为 AX_INVALID_POOLID，则返回错误码 AX_ERR_VDEC_ILLEGAL_PARAM。
- 用户必须要调用 AX_POOL_CreatePool 创建一个缓存 VB 池，再通过调用接口 AX_VDEC_AttachPool 把当前解码通道绑定到固定 Pool Id 上。不允许一个解码通道绑定

多个 Pool Id。

- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。
- 如果通道正在销毁，返回错误码 AX_ERR_VDEC_NOT_PERM。

AX_VDEC_DetachPool

【描述】

将解码通道从某个缓存 VB 池中解绑定。

【语法】

AX_S32 AX_VDEC_DetachPool(AX_VDEC_GRP VdGrp);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号。 取值范围：[0, AX_VDEC_MAX_GRP_NUM)。	输入

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 如果通道正在销毁，返回错误码 AX_ERR_VDEC_NOT_PERM。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。

AX_VDEC_JpegDecodeOneFrame

【描述】

JPEG 单帧解码。

【语法】

AX_S32 AX_VDEC_JpegDecodeOneFrame(AX_VDEC_DEC_ONE_FRM_T *pstParam);

【参数】

参数名称	描述	输入/输出
pstParam	单帧编码信息结构体指针。	既有输入也有输出

【返回值】

返回值	描述
0	成功
非 0	失败，返回错误码

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 输出参数 pstGrpSet 指向的地址不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。
- 输入参数校验，如果输入参数非法，返回 AX_ERR_VDEC_ILLEGAL_PARAM。
- 解码帧校验，如果输入流不支持，返回 AX_ERR_VDEC_NOT_SUPPORT。
- 源数据使用的结构体和创建通道解码送流结构体相同，均使用“AX_VDEC_STREAM_T”，用户需要使用物理连续内存，并且物理地址必须赋值。
- 输入内存和输出内存使用 AX_SYS_MemAlloc 申请内存或者申请 pool 再 map。

- 如果输入内存虚拟地址是 cached，在读入 jpg 文件后，需要 flushcache 再调用单次解码接口，否则会出现解码失败情况。

AX_VDEC_GetStreamBufInfo

【描述】

获取解码通道输入码流缓存的信息。

【语法】

AX_S32 AX_VDEC_GetStreamBufInfo(AX_VDEC_GRP VdGrp, AX_VDEC_STREAM_BUF_INFO_T *pstStreamBufInfo);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号。 取值范围：[0, AX_VDEC_MAX_GRP_NUM)。	输入
pstStreamBufInfo	解码通道输入码流缓存信息的结构体指针。	输出

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。

- 保证通道已经被创建，否则返回错误码 AX_ERR_VDEC_UNEXIST。
- 输入码流缓存的信息结构体指针不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。

AX_VDEC_ExtractStreamHeaderInfo

【描述】

提取 H264/H265 码流 I 帧头（SPS）中的参考帧数，宽高分辨率等信息。

【语法】

```
AX_S32 AX_VDEC_ExtractStreamHeaderInfo(const AX_VDEC_STREAM_T *pstStreamBuf,
AX_PAYLOAD_TYPE_E enVideoType, AX_VDEC_BITSTREAM_INFO_T *pstBitStreamInfo);
```

【参数】

参数名称	描述	输入/输出
pstStreamBuf	输入码流信息结构体指针。	输入
enVideoType	输入码流类型，仅支持 PT_H264。	输入
pstBitStreamInfo	获取到码流 I 帧头中的信息结构体指针。	输出

【返回值】

返回值	描述
0	成功
非 0	失败，返回错误码

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 结构体指针 pstStreamBuf 和 pstBitStreamInfo 不能指向空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。
- 结构体指针 pstStreamBuf 送入的地址和长度信息不能为 0，否则返回错误码 AX_ERR_VDEC_ILLEGAL_PARAM。

- enVideoType 仅支持 PT_H264，否则返回错误码 AX_ERR_VDEC_NOT_SUPPORT。
- 若输入码流中找到 I 帧头（SPS），函数返回 AX_SUCCESS，否则返回 AX_ERR_VDEC_STRM_ERROR。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。

AX_VDEC_GetVuiParam

【描述】

获取解码通道码流 VUI 信息。

【语法】

AX_S32 AX_VDEC_GetVuiParam(AX_VDEC_GRP VdGrp, AX_VDEC_VUI_PARAM_T *pstVuiParam);

【参数】

参数名称	描述	输入/输出
VdGrp	解码通道号	输入
pstVuiParam	VUI 信息结构体指针。	输出

【返回值】

返回值	描述
非 0	失败，该返回值为错误码
0	成功

【需求】

- 头文件：ax_vdec_api.h
- 库文件：libax_vdec.so

【注意】

- 通道号应在有效通道号范围内，否则返回错误码 AX_ERR_VDEC_INVALID_GRPID。
- 硬件资源已经初始化，否则返回错误码 AX_ERR_VDEC_NOT_INIT。
- 结构体指针 pstVuiParam 不能为空，否则返回错误码 AX_ERR_VDEC_NULL_PTR。
- 通道应已经创建成功，否则返回错误码 AX_ERR_VDEC_UNEXIST。

AX_VDEC_GetPicBufferSize

【描述】

获取 VDEC 图像 buffer 的大小。

【语法】

AX_U32 AX_VDEC_GetPicBufferSize(AX_U32 uWidth, AX_U32 uHeight, AX_PAYLOAD_TYPE_E enType);

【参数】

参数名称	描述	输入/输出
uHeight	图像高度	输入
uWidth	图像宽度	输入
enType	码流编码类型枚举	输入

【返回值】

返回值	描述
refBuffSize	解码图像每帧 buffer 大小

【需求】

- 头文件：ax_buffer_tool.h

【注意】

- 对于 JDEC 解码，该接口输出 buffer 的大小始终按照 NV12 格式计算。如果 JDEC 输出 YUV 格式为 YUV444 等其他格式，请使用 AX_JDEC_GetYuvBufferSize 接口。

【举例】

无

AX_JDEC_GetYuvBufferSize

【描述】

获取 JDEC 输出 YUV buffer 的大小。

【语法】

AX_U32 AX_JDEC_GetYuvBufferSize(AX_U32 uWidth, AX_U32 uHeight, AX_IMG_FORMAT_E eOutputFormat);

【参数】

参数名称	描述	输入/输出
uHeight	图像高度	输入
uWidth	图像宽度	输入
eOutputFormat	解码出的 YUV 格式	输入

【返回值】

返回值	描述
refBuffSize	存储解码输出每帧 YUV 的 buffer 大小

【需求】

- 头文件：ax_buffer_tool.h

【注意】

- 此接口只用于 JDEC 解码计算输出 YUV buffer 大小。
- 根据传入的 eOutputFormat，计算 YUV buffer 大小，计算规则：
  - YUV400：按照 width*height 申请，width、height 按照 16 对齐。
  - YUV422：按照 width*height*2 申请，width、height 按照 16 对齐。
  - YUV420：按照 width*height*3/2 申请，width、height 按照 16 对齐。
  - YUV444：按照 width*height*3 申请，width、height 按照 16 对齐。

- 其他：均按照 width*height*3 申请，width、height 按照 16 对齐。

【举例】

无

4 数据结构

AX_VDEC_MAX_GRP_NUM

【说明】

定义解码输入通道最大个数。

【定义】

#define AX_VDEC_MAX_GRP_NUM 16

【注意】

本芯片 VDEC 硬件支持输出通道最大个数，常量。

【相关数据类型及接口】

无。

AX_VDEC_MAX_WIDTH

【说明】

定义能够解码的 H264 码流的最大宽度。

【定义】

#define AX_VDEC_MAX_WIDTH 1920

【注意】

VDEC 硬件支持解码 H264 图像最大宽度，取决于芯片型号。

【相关数据类型及接口】

无。

AX_VDEC_MAX_HEIGHT

【说明】

定义能够解码的 H264 码流的最大高度。

【定义】

#define AX_VDEC_MAX_HEIGHT 1920

【注意】

VDEC 硬件支持解码 H264 图像最大高度，取决于芯片型号。

【相关数据类型及接口】

无。

AX_VDEC_MIN_WIDTH

【说明】

定义能够解码的 H264 码流的最小宽度。

【定义】

```
#define AX_VDEC_MIN_WIDTH                48
```

【注意】

VDEC 硬件支持解码 H264 图像最小宽度，取决于芯片型号。

【相关数据类型及接口】

无。

AX_VDEC_MIN_HEIGHT

【说明】

定义能够解码的 H264 码流的最小高度。

【定义】

#define AX_VDEC_MIN_HEIGHT 48

【注意】

VDEC 硬件支持解码 H264 图像最小高度，取决于芯片型号。

【相关数据类型及接口】

无。

AX_JDEC_MAX_WIDTH

【说明】

定义能够解码的 JPEG/MJPEG 码流的最大宽度。

【定义】

```
#define AX_JDEC_MAX_WIDTH                16384
```

【注意】

JDEC 硬件支持解码 JPEG/MJPEG 图像最大宽度，取决于芯片型号。

【相关数据类型及接口】

无。

AX_JDEC_MAX_HEIGHT

【说明】

定义能够解码的 JPEG/MJPEG 码流的最大高度。

【定义】

#define AX_JDEC_MAX_HEIGHT 16384

【注意】

JDEC 硬件支持解码 JPEG/MJPEG 图像最大高度，取决于芯片型号。

【相关数据类型及接口】

无。

AX_JDEC_MIN_WIDTH

【说明】

定义能够解码的 JPEG/MJPEG 码流的最小宽度。

【定义】

```
#define AX_JDEC_MIN_WIDTH                48
```

【注意】

JDEC 硬件支持解码 JPEG/MJPEG 图像最小宽度，取决于芯片型号。

【相关数据类型及接口】

无。

AX_JDEC_MIN_HEIGHT

【说明】

定义能够解码的 JPEG/MJPEG 码流的最小高度。

【定义】

#define AX_JDEC_MIN_HEIGHT 48

【注意】

JDEC 硬件支持解码 JPEG/MJPEG 图像最小高度，取决于芯片型号。

【相关数据类型及接口】

无。

AX_MAX_VDEC_USER_DATA_SIZE

【说明】

定义 VDEC 可获取的最大数据长度(H264)。

【定义】

#define AX_MAX_VDEC_USER_DATA_SIZE (2048)

【注意】

对应 H264 码流中可获取的最大 SEI 信息长度。

【相关数据类型及接口】

AX_VDEC_GetUserData ()

AX_VDEC_USERDATA_T

AX_MAX_JDEC_USER_DATA_SIZE

【说明】

定义 JDEC 可获取的最大数据长度(JPEG，MJPG)。

【定义】

#define AX_MAX_JDEC_USER_DATA_SIZE (0xFFFF - 2)

【注意】

对应 JPEG、MJPG 码流中可获取的最大 COM 信息长度，COM TAG 类型对应起始头为“FFFE”。

【相关数据类型及接口】

AX_VDEC_GetUserData ()

AX_VDEC_USERDATA_T

AX_MAX_VDEC_USER_DATA_CNT

【说明】

定义 VDEC 可获取的最大 userdata 数据个数(H264)。

【定义】

#define AX_MAX_VDEC_USER_DATA_CNT (20)

【注意】

JDEC 可获取 userdata 数据个数为 1

【相关数据类型及接口】

AX_VDEC_GetUserData ()

AX_VDEC_USERDATA_T

AX_VDEC_GRP

【说明】

定义解码输入通道号类型。

【定义】

typedef AX_S32 AX_VDEC_GRP;

【注意】

取值范围：[0, AX_VDEC_MAX_GRP_NUM)。

【相关数据类型及接口】

无。

AX_VDEC_MOD_ATTR_T

【说明】

VDEC 模块属性，所有通道功能均受其影响。

【定义】

```
typedef struct axVDEC_MOD_ATTR_T {
    AX_U32 u32MaxGroupCount;
} AX_VDEC_MOD_ATTR_T;
```

【成员】

参数名称	描述
u32MaxGroupCount	最大解码通道数；(当前不支持) 取值范围[0, AX_VDEC_MAX_GRP_NUM] 若为 0，则等同于 AX_VDEC_MAX_GRP_NUM

【注意】

无。

【相关数据类型及接口】

无。

AX_VDEC_INPUT_MODE_E

【说明】

发送解码数据流方式。

【定义】

```
typedef enum axVDEC_INPUT_MODE_E {
    AX_VDEC_INPUT_MODE_NAL = 0,
    AX_VDEC_INPUT_MODE_FRAME,
    AX_VDEC_INPUT_MODE_SLICE,
    AX_VDEC_INPUT_MODE_STREAM,
    AX_VDEC_INPUT_MODE_COMPAT,
    AX_VDEC_INPUT_MODE_BUTT
} AX_VDEC_INPUT_MODE_E;
```

【成员】

参数名称	描述
AX_VDEC_INPUT_MODE_NAL	按 NAL 方式发送码流，每次发送以一个 NAL 为单位(未支持)
AX_VDEC_INPUT_MODE_FRAME	按帧方式发送码流，以帧为单位(推荐使用)
AX_VDEC_INPUT_MODE_SLICE	按 SLICE 方式发送码流，以 SLICE 为单位(未支持)
AX_VDEC_INPUT_MODE_STREAM	按流方式发送码流。 JPEG/MJPEG 解码不支持此模式
AX_VDEC_INPUT_MODE_COMPAT	兼容模式发送码流(未支持)

【注意】

- 当一帧码流的最后一包没把 bEndOfFrame 为 AX_TRUE 时，还可以再发送一个 bEndOfFrame 为 AX_TRUE 的空包(注意带上当前帧的 u64PTS)给解码器内部标识当前帧

码流已发送完毕。

【相关数据类型及接口】

AX_VDEC_GRP_ATTR_T

AX_VDEC_CreateGrp()

AX_VDEC_GRP_ATTR_T

【说明】

视频解码通道属性结构体。

【定义】

```
typedef struct axVDEC_GRP_ATTR_T {
    AX_PAYLOAD_TYPE_E           enCodecType;
    AX_VDEC_INPUT_MODE_E        enInputMode;
    AX_LINK_MODE_E              enLinkMode;
    AX_VDEC_OUTPUT_ORDER_E      enOutOrder;
    AX_U32                      u32PicWidth;
    AX_U32                      u32PicHeight;
    AX_U32                      u32FrameHeight;
    AX_U32                      u32StreamBufSize;
    AX_POOL_SOURCE_E            enVdecVbSource;
    AX_U32                      u32FrameBufCnt;
    AX_S32                      s32DestroyTimeout;
} AX_VDEC_GRP_ATTR_T;
```

【成员】

成员名称	描述
enCodecType	解码协议类型
enInputMode	送解码数据流方式
enLinkMode	是否 link 模式
enOutOrder	解码输出顺序，推荐值 VIDEO_OUTPUT_ORDER_DISP
u32PicWidth	通道支持的解码图像最大宽度(以像素为单位)
u32PicHeight	通道支持的解码图像最大高度(以像素为单位)
u32FrameHeight	通道输出图像的总高度，当此值大于图像高度时，图像尾部区域可视为填充数据；该值不能超过 SPS 信息中描述的高度
u32StreamBufSize	SDK 内部输入码流最大缓存容量，以 Byte 为单位，buf 大小建议 16

成员名称	描述
	对齐
enVdecVbSource	解码使的 VB 模式(com、user、private)，默认使用 com vb
u32FrameBufCnt	用户需要设置输出 buffer 个数，每路开的 output buffer 个数最大值为 AX_VDEC_MAX_FRAME_BUF_CNT，输出 buffer 个数由客户控制
s32DestroyTimeout	当前只支持输入 0 的模式

【注意】

- u32StreamBufSize 最小必须能容纳码流中最占内存的一帧码流，否则解码会失败。建议最小设置为待解码码流一帧 YUV 大小，以 YUV NV12 格式为例，此大小为 Width * Height * 3 / 2。
- enVdecVbSource 仅在设置为 AX_POOL_SOURCE_USER 时支持 AX_VDEC_AttachPool 和 AX_VDEC_DetachPool。

【相关数据类型及接口】

AX_VDEC_CreateGrp()

【备注】

AX_LINK_MODE_E 和 AX_PAYLOAD_TYPE_E 见文档《50 - AX 公共数据结构文档.docx》

AX_VDEC_OUTPUT_ORDER_E

【说明】

解码输出图像的顺序定义。

【定义】

```
typedef enum axVIDEO_OUTPUT_ORDER_E {
    AX_VDEC_OUTPUT_ORDER_DISP = 0,
    AX_VDEC_OUTPUT_ORDER_DEC,
    AX_VDEC_OUTPUT_ORDER_BUTT
} AX_VDEC_OUTPUT_ORDER_E;
```

【成员】

成员名称	描述
AX_VDEC_OUTPUT_ORDER_DISP	按照显示顺序输出
AX_VDEC_OUTPUT_ORDER_DEC	按照解码顺序输出

【注意】

解码含 B 帧的码流应设为显示顺序输出。

【相关数据类型及接口】

AX_VDEC_PARAM_VIDEO_T

AX_VDEC_GRP_PARAM_S

AX_VDEC_SetGrpParam()

AX_VDEC_GetGrpParam()

AX_VDEC_MODE_E

【说明】

解码模式定义。

【定义】

```
typedef enum axVIDEO_MODE_E{
    VIDEO_DEC_MODE_IPB = 0,
    VIDEO_DEC_MODE_IP,
    VIDEO_DEC_MODE_I,
    VIDEO_DEC_MODE_BUTT
} AX_VDEC_MODE_E;
```

【成员】

成员名称	描述
VIDEO_DEC_MODE_IPB	解码 IPB 帧
VIDEO_DEC_MODE_IP	解码 IP 帧(跳过非参考帧解码，B 帧被参考的情况下不会跳过)
VIDEO_DEC_MODE_I	解码 I 帧
VIDEO_DEC_MODE_BUTT	无效参数

AX_VDEC_GRP_PARAM_T

【说明】

视频解码的属性定义。

【定义】

```
typedef struct axVDEC_GRP_PARAM_T {
    AX_VDEC_MODE_E enVdecMode;
} AX_VDEC_GRP_PARAM_T;
```

【成员】

成员名称	描述
enVdecMode	解码模式，具体定义请查看结构体描述。

【相关数据类型及接口】

AX_VDEC_SetGrpParam()

AX_VDEC_GetGrpParam()

AX_VDEC_RECV_PIC_PARAM_T

【说明】

视频码流结构体定义。

【定义】

```
typedef struct axVDEC_RECV_PIC_PARAM_T {
    AX_S32 s32RecvPicNum;
} AX_VDEC_RECV_PIC_PARAM_T;
```

【成员】

参数名称	描述
s32RecvPicNum	解码最大帧数。 取值范围[-1, 2147483647] -1 和 0 都代表最大帧数无限制。

【注意】

无。

【相关数据类型及接口】

AX_VDEC_StartRecvStream()

AX_VDEC_STREAM_T

【说明】

视频解码的码流结构体定义。

【定义】

```
typedef struct axVDEC_STREAM_T {
    AX_U64    u64PTS;
    AX_U64    u64PrivateData;
    AX_BOOL bEndOfFrame;
    AX_BOOL bEndOfStream;
    AX_BOOL bSkipDisplay;
    AX_U32    u32StreamPackLen;
    AX_U8* ATTRIBUTE pu8Addr;
    AX_U64    u64PhyAddr;
} AX_VDEC_STREAM_T;
```

【成员】

成员名称	描述
u64PTS	时间戳，以 us 为单位； 若不为-1，VDEC 将透传到输出帧结构体 AX_VIDEO_FRAME_T 的 u64PTS； 若为-1，VDEC 在 unlInk 模式下 GetFrame 获取不到当前帧，link 模式下该帧不会往下级发送。 JDEC 不判断 PTS，直接透传到输出帧结构体 AX_VIDEO_FRAME_T 的 u64PTS。
u64PrivateData	私有数据，透传到输出帧结构体 AX_VIDEO_FRAME_T 的 u64PrivateData。(暂未实现)
bEndOfFrame	当前帧是否结束。
bEndOfStream	是否发完所有码流。

成员名称	描述
bSkipDisplay	当前帧解码后是否跳过显示输出；(暂未使用)
u32StreamPackLen	码流包的长度，以 byte 为单位，取值范围[0,u32StreamBufSize]。
pu8Addr	送入视频码流的地址。
u64PhyAddr	送入码流包的物理地址。(暂未使用) 注：用户指定物理地址的内存空间必须物理地址连续。

【注意】

- u64PTS 是每帧的时间戳，只有在按照帧的方式送流解码的情况下有效。若按流方式(INPUT_MODE_STREAM)送流解码的情况下无效。还有 H264 协议中如果开始送帧的 header 部分单独送入解码，不会有解码图像输出该时间戳。
- 当发完所有码流后，把 bEndOfStream 置为 1，表示码流结束，这时解码器会解完发送下来的所有码流并输出所有图像。如果发完所有码流后把 bEndOfStream 置为 0，解码器内部可能残余大于等于一帧的图像未解码输出，因为解码器必须等到下一帧码流到来才能知道当前帧已经结束，送入解码。
- VDEC 支持发送一包 bEndOfStream 为 1 的空码流包(地址为空或长度为 0)。
- JPEG/MJPEG 解码或者流模式发送码流时 bSkipDisplay 标志无效。

【相关数据类型及接口】

AX_VDEC_SendStream()

AX_VDEC_DISPLAY_MODE_E

【说明】

定义显示模式枚举。

【定义】

```
typedef enum axVIDEO_DISPLAY_MODE_E {
    AX_VDEC_DISPLAY_MODE_PREVIEW = 0x0,
    AX_VDEC_DISPLAY_MODE_PLAYBACK = 0x1,
    AX_VDEC_DISPLAY_MODE_BUTT
} AX_VDEC_DISPLAY_MODE_E;
```

【成员】

成员名称	描述
AX_VDEC_DISPLAY_MODE_PREVIEW	预览模式
AX_VDEC_DISPLAY_MODE_PLAYBACK	回放模式

【注意】

- 若设为预览模式，不论是 link 模式或 Unlink 模式，输出帧以不阻塞模式输出，若输出队列满，则丢弃新帧。
- 若设为回放模式，不论是 link 模式或 Unlink 模式，输出帧以阻塞模式输出，若输出队列满，则阻塞等待，以此反压到输入队列满，阻塞输入。

【相关数据类型及接口】

AX_VDEC_SetDisplayMode()

AX_VDEC_GetDisplayMode()

AX_VDEC_GRP_STATUS_T

【说明】

定义通道状态结构体。

【定义】

```
typedef struct axVDEC_GRP_STATUS_T {
    AX_PAYLOAD_TYPE_E enCodecType;
    AX_U32 u32LeftStreamBytes;
    AX_U32 u32LeftStreamFrames;
    AX_U32 u32LeftPics;
    AX_BOOL bStartRecvStream;
    AX_U32 u32RecvStreamFrames;
    AX_U32 u32DecodeStreamFrames;
    AX_U32 u32PicWidth;
    AX_U32 u32PicHeight;
    AX_BOOL bInputFifoIsFull;
    AX_VDEC_DECODE_ERROR_T stVdecDecErr;
} AX_VDEC_GRP_STATUS_T;
```

【成员】

成员名称	描述
enCodecType	解码协议类型。
u32LeftStreamBytes	码流 buffer 中待解码的 byte 数，包括正在解码的当前帧中未解码的 byte 数。
u32LeftStreamFrames	1、码流 buffer 中待解码的帧数，不包括正在解码的当前帧。–1 表示无效(流模式发送时无效)。 3、码流出错丢帧时可能计数不准。

成员名称	描述
u32LeftPics	图像 buffer 中剩余的 pic 数目。
bStartRecvStream	解码器是否已经启动接收码流。
u32RecvStreamFrames	码流 buffer 中已接收码流帧数。 -1 表示无效(流模式发送时无效)。
u32DecodeStreamFrames	码流 buffer 中已解码帧数。
u32PicWidth	当前通道图片宽度。
u32PicHeight	当前通道图片高度。
bInputFifoIsFull	输入队列(缓存)是否满。
stVdecDecErr	解码错误信息。

【注意】

无。

【相关数据类型及接口】

AX_VDEC_QueryStatus()

【备注】

AX_PAYLOAD_TYPE_E 见文档《50 - AX 公共数据结构文档.docx》

AX_VDEC_DECODE_ERROR_T

【说明】

定义通道解码错误信息结构体。

【定义】

```
typedef struct ax_VDEC_DECODE_ERROR_T {
    AX_S32 s32FormatErr;
    AX_S32 s32PicSizeErrSet;
    AX_S32 s32StreamUnsprt;
    AX_S32 s32PackErr;
    AX_S32 s32RefErrSet;
    AX_S32 s32PicBufSizeErrSet;
    AX_S32 s32StreamSizeOver;
    AX_S32 s32VdecStreamNotRelease;
} AX_VDEC_DECODE_ERROR_T;
```

【成员】

成员名称	描述
s32FormatErr	暂不支持。
s32PicSizeErrSet	图像的宽(或高)不在芯片解码支持的范围。
s32StreamUnsprt	不支持解码的码流。
s32PackErr	码流有错误。
s32RefErrSet	暂不支持。
s32PicBufSizeErrSet	图像 buffer 内存大小不够。

成员名称	描述
s32StreamSizeOver	暂不支持。
s32VdecStreamNotRelease	暂不支持。

【注意】

- 所有错误信息均以累加计数的形式表现。比如解码每发现一次码流有错，s32PackErr 数值就加 1。
- 复位通道后所有错误信息计数清零。

【相关数据类型及接口】

AX_VDEC_QueryStatus()

AX_VDEC_GRP_SET_INFO_T

【描述】

查询所有创建通道中是否有通道有帧输出。

【定义】

```
typedef struct axVDEC_GRP_SET_INFO_T {
    AX_U32 u32GrpCount;
    AX_VDEC_GRP VdGrp[AX_VDEC_MAX_GRP_NUM];
} AX_VDEC_GRP_SET_INFO_T;
```

【成员】

成员名称	描述
u32GrpCount	有输出的通道数量。
VdGrp	有输出的通道号数组。

【相关数据类型及接口】

AX_VDEC_SelectGrp()

AX_VDEC_DEC_ONE_FRM_T

【说明】

视频解码的码流结构体定义。

【定义】

```
typedef struct axVDEC_DEC_ONE_FRM_T {
    AX_VDEC_STREAM_T stStream;
    AX_VIDEO_FRAME_T stFrame;
} AX_VDEC_DEC_ONE_FRM_T;
```

【成员】

成员名称	描述
stStream	码流结构体
stFrame	帧结构体

AX_VDEC_USERDATA_T

【说明】

userData 的码流结构体定义。

【定义】

```
typedef struct axVDEC_USERDATA_T {
    AX_U64 u64PhyAddr;
    AX_U32 u32UserDataCnt;
    AX_U32 u32Len;
    AX_U32 u32BufSize;
    AX_U32 u32DataLen[AX_MAX_VDEC_USER_DATA_CNT];
    AX_BOOL bValid;
    AX_U8 ATTRIBUTE *pu8Addr;
} AX_VDEC_USERDATA_T;
```

【成员】

成员名称	描述
u64PhyAddr	userData 信息存储内存物理地址
u32UserDataCnt	userData 信息个数
u32Len	userData 信息总有效长度
u32BufSize	userData 信息存储 buf 大小
u32DataLen	每个 userData 信息的有效长度
bValid	userData 信息是否有效，仅当为 1 时，代表存储内存中的数据是有效的。
pu8Addr	userData 信息存储内存虚拟地址。

AX_VDEC_STREAM_BUF_INFO_T

【说明】

解码通道输入缓存信息的结构体定义。

【定义】

```
typedef struct axVDEC_STREAM_BUF_INFO_T{
    AX_U64 phyStart;
    AX_U8 *virStart;
    AX_U32 totalSize;
    AX_U32 readAbleSize;
    AX_U32 writeAbleSize;
    AX_U32 readOffset;
    AX_U32 writeOffset;
} AX_VDEC_STREAM_BUF_INFO_T;
```

【成员】

成员名称	描述
phyStart	输入缓存的物理地址。
virStart	输入缓存的虚拟地址
totalSize	输入缓存的总大小
readAbleSize	输入缓存的可读空间大小。
writeAbleSize	输入缓存的可写空间大小
readOffset	输入缓存的读指针偏移量
writeOffset	输入缓存的写指针偏移量

AX_VDEC_VUI_PARAM_T

【说明】

解码通道 VUI 信息的结构体定义。

【定义】

typedef struct axVdec_VUI_PARAM_T {

AX_VDEC_VUI_ASPECT_RATIO_T stVuiAspectRatio;

AX_VDEC_VUI_TIME_INFO_T stVuiTimeInfo;

AX_VDEC_VUI_VIDEO_SIGNAL_T stVuiVideoSignal;

AX_VDEC_VUI_BITSTREAM_RESTRIC_T stVuiBitstreamRestric;

} AX_VDEC_VUI_PARAM_T;

【成员】

成员名称	描述
stVuiAspectRatio	定义 H.264 协议解码通道 Vui 中 AspectRatio 信息的结构体。
stVuiTimeInfo	定义 H.264 协议解码通道 Vui 中 Time_Info 信息的结构体
stVuiVideoSignal	定义 H.264 协议解码通道 Vui 中 VideoSignal 信息的结构体
stVuiBitstreamRestric	定义 H.264 协议解码通道 Vui 中 BitstreamRestric 信息的结构体

AX_VDEC_VUI_ASPECT_RATIO_T

【说明】

H.264 协议解码通道 Vui 中 AspectRatio 信息的结构体定义。

【定义】

```
typedef struct axVdec_VUI_ASPECT_RATIO_T {
    AX_U8 aspect_ratio_info_present_flag;
    AX_U8 aspect_ratio_idc;
    AX_U8 overscan_info_present_flag;
    AX_U8 overscan_appropriate_flag;
    AX_U16 sar_width;
    AX_U16 sar_height;
} AX_VDEC_VUI_ASPECT_RATIO_T;
```

【成员】

成员名称	描述
aspect_ratio_info_present_flag	具体含义请参见 H.264 协议。 取值范围：0 或 1。
aspect_ratio_idc	具体含义请参见 H.264 协议。 取值范围：[0, 255],17 ~ 254 保留。
overscan_info_present_flag	具体含义请参见 H.264 协议。 取值范围：0 或 1。
overscan_appropriate_flag	具体含义请参见 H.264 协议。 取值范围：0 或 1。
sar_width	具体含义请参见 H.264 协议。

成员名称	描述
	取值范围：(0, 65535]，并且与 sar_height 互质。
sar_height	具体含义请参见 H.264 协议，系统默认为 1。 取值范围：(0, 65535]，并且与 sar_width 互质。

AX_VDEC_VUI_TIME_INFO_T

【说明】

H.264 协议解码通道 Vui 中 Time_Info 信息的结构体定义。

【定义】

```
typedef struct axVdec_VUI_TIME_INFO_T {
    AX_U8 timing_info_present_flag;
    AX_U32 num_units_in_tick;
    AX_U32 time_scale;
    AX_U8 fixed_frame_rate_flag;
    AX_U32 num_ticks_poc_diff_one_minus1;
} AX_VDEC_VUI_TIME_INFO_T;
```

【成员】

成员名称	描述
timing_info_present_flag	具体含义请参见 H.264 协议。 取值范围：0 或 1。
num_units_in_tick	具体含义请参见 H.264 协议。 取值范围：大于 0。
time_scale	具体含义请参见 H.264 协议。 取值范围：大于 0。
fixed_frame_rate_flag	具体含义请参见 H.264 协议。 取值范围：0 或 1。
num_ticks_poc_diff_one_minus1	暂未使用。

AX_VDEC_VUI_VIDEO_SIGNAL_T

【说明】

H.264 协议解码通道 Vui 中 VideoSignal 信息的结构体定义。

【定义】

```
typedef struct axVdec_VIDEO_SIGNAL_T {
    AX_U8 video_signal_type_present_flag;
    AX_U8 video_format;
    AX_U8 video_full_range_flag;
    AX_U8 colour_description_present_flag;
    AX_U8 colour_primaries;
    AX_U8 transfer_characteristics;
    AX_U8 matrix_coefficients;
} AX_VDEC_VUI_VIDEO_SIGNAL_T;
```

【成员】

成员名称	描述
video_signal_type_present_flag	具体含义请参见 H.264 协议。 取值范围：0 或 1。
video_format	具体含义请参见 H.264 协议。 取值范围：H264:(0, 7]。
video_full_range_flag	具体含义请参见 H.264 协议。 取值范围：0 或 1。
colour_description_present_flag	具体含义请参见 H.264 协议。 取值范围：0 或 1。

成员名称	描述
colour_primaries	具体含义请参见 H.264 协议。 取值范围：[0,255]。
transfer_characteristics	具体含义请参见 H.264 协议。 取值范围：[0,255]。
matrix_coefficients	具体含义请参见 H.264 协议。 取值范围：[0,255]。

AX_VDEC_VUI_BITSTREAM_RESTRIC_T

【说明】

H.264 协议解码通道 Vui 中 BitstreamRestric 信息的结构体定义。

【定义】

```
typedef struct axVdec_VUI_BITSTREAM_RESTRIC_T {
    AX_U8 bitstream_restriction_flag;
} AX_VDEC_VUI_BITSTREAM_RESTRIC_T;
```

【成员】

成员名称	描述
bitstream_restriction_flag	具体含义请参见 H.264 协议。 取值范围：0 或 1。

AX_VDEC_USRPIC_T

【说明】

用户图片属性结构指针。

【定义】

```
typedef struct {
    AX_VIDEO_FRAME_INFO_T stFrmInfo;
    AX_BOOL bInstant;
    AX_BOOL bEnable;
} AX_VDEC_USRPIC_T;
```

【成员】

成员名称	描述
stFrmInfo	用户图片帧属性结构体。
bInstant	暂未使用
bEnable	用户图片插入使能控制。

AX_VDEC_BITSTREAM_INFO_T

【说明】

获取 I 帧头中的某些信息的结构体定义。

【定义】

```
typedef struct axVDEC_BITSTREAM_INFO_T {
    AX_U32 u32Width;
    AX_U32 u32Height;
    AX_U32 u32RefFramesNum;
    AX_U32 u32BufferCnt;
    AX_U32 u32BitDepthY;
    AX_U32 u32BitDepthC;
} AX_VDEC_BITSTREAM_INFO_T;
```

【成员】

成员名称	描述
u32Width	SPS 中的宽度，某些码流可能无此信息。
u32Height	SPS 中的高度，某些码流可能无此信息。
u32RefFramesNum	SPS 中的参考帧个数，某些码流可能无此信息，按标准最大为 16。
u32BufferCnt	根据 SPS 中的信息计算出的解码需要的 buf 个数。
u32BitDepthY	SPS 中的 Y 位宽，某些码流可能无此信息，默认为 8 bit。
u32BitDepthC	SPS 中的 C（UV）位宽，某些码流可能无此信息，默认为 8 bit。

5 错误码

错误码详见《55 - AX 软件错误码文档》文档。

6 调试信息

【应用层 so 库版本号】

终端输入命令 # strings /opt/lib/libax_vdec.so | grep "version"

```
/opt/data # strings /opt/lib/libax_vdec.so | grep "version"
[Axera version]: libax_vdec.so V1.8.0_20240130021715 Jan 30 2024 02:21:30 JK
```

【应用层 LOG 等级】

命令方式：echo ulog [id] [level] > /proc/ax_proc/logctl

- id: [0-33]表示单模块 id，如果设置所有模块则用 all 表示；此 ID 遵从 AX_MOD_ID_E 枚举类型中的定义，比如 AX_ID_VDEC = 0x08 。
- level: Log 等级，0-7 数字越小代表 Log 等级越高；默认是 4，只打印出 WARN 及以上等级的 Log；

若使用此命令： echo ulog 8 7 > /proc/ax_proc/logctl

可以打印出 VDEC DEBUG 及以上等级的 Log，也就是 VDEC 应用层全部 Log。

注意：应用层 VDEC/JDEC 功能信息统一通过 AX_ID_VDEC 控制

【应用层 LOG 输出】

默认打开输出 WARN 及以上等级的 Log 到文件(Log 文件存储路径位于 /opt/data/AXSyslog/syslog/目录)，可以使用如下命令来控制关闭 Log 或将 Log 输出到终端：

- 关闭 Log：echo ulog off > /proc/ax_proc/logctl
- Log 输出到控制台：echo ulog console > /proc/ax_proc/logctl
- 丢弃 Log：echo ulog null > /proc/ax_proc/logctl，作用与关闭 Log 一致。

可以再使用如下命令动态控制 Log 输出：

- 打开 Log：echo ulog on > /proc/ax_proc/logctl

- Log 输出到文件：echo ulog file > /proc/ax_proc/logctl

注意：以上修改/proc/ax_proc/logctl 只在当前开机时有效，重启后失效。

假设想修改每次开机默认 Log 配置，需改配置文件/etc/ax_syslog.conf，修改方式雷同上述命令，修改完保存并重启，配置文件生效。

若改了配置文件/etc/ax_syslog.conf 之后重启，再使用命令修改/proc/ax_proc/logctl，则当前开机状态下以/proc/ax_proc/logctl 为优先，再重启后恢复为/etc/ax_syslog.conf 配置。

【内核层 LOG 等级】

默认输出 WARN 及以上等级的 Log 到文件，存储路径同样位于/opt/data/AXSyslog/ syslog 目录。

命令方式：echo klog [id] [level] > /proc/ax_proc/logctl

- id: [0-33]表示单模块 id，如果设置所有模块则用 all 表示；此 ID 遵从 AX_MOD_ID_E 枚举类型中的定义，比如 AX_ID_VDEC = 0x08。
- level: Log 等级，0-7 数字越小代表 Log 等级越高；默认是 4，只打印出 WARN 及以上等级的 Log；

若使用此命令：echo klog 8 7 > /proc/ax_proc/logctl

可以打印出 VDEC DEBUG 及以上等级的 Log，也就是 VDEC 内核层全部 Log。

注意：内核态 VDEC/JDEC 功能信息统一通过 AX_ID_VDEC 控制

【内核层 LOG 输出】

默认打开输出 WARN 及以上等级的 Log 到文件(Log 文件存储路径位于/opt/data/AXSyslog/syslog/目录)，可以使用如下命令来控制关闭 Log 或开启 LOG：

- 关闭 Log：echo klog off > /proc/ax_proc/logctl

可以再使用如下命令动态控制 LOG 输出：

- 打开 Log：echo klog on > /proc/ax_proc/logctl

注意：以上修改 /proc/ax_proc/logctl 只在当前开机时有效，重启后失效。

假设想修改每次开机默认 Log 配置，需改配置文件 /etc/ax_syslog.conf，修改方式雷同上

述命令，修改完保存并重启，配置文件生效。

若改了配置文件/etc/ax_syslog.conf 之后重启，再使用命令修改/proc/ax_proc/logctl，则当前开机状态下以/proc/ax_proc/logctl 为优先，再重启后恢复为/etc/ax_syslog.conf 配置。

【内核层 ko 库版本号】

终端输入命令 # cat /proc/ax_proc/vdec ，如下调试信息第二行 `[version]:`

【调试信息】

```
/root # cat /proc/ax_proc/vdec
/opt/bin/FRTTest # cat /proc/ax_proc/vdec
----------------------- AX VDEC VERSION -----------------------
[Axera version]: ax_vdec V1.8.0 Feb  2 2024 11:40:53

[Axera version]: libax_vdec.so V1.8.1 Feb  4 2024 10:24:52

================ AX_VDEC_PID: 1529 WorkingGrpNum: 1 ================

----------------------- JDEC MODULE PARAM -----------------------
AX_JDEC_MAX_WIDTH     AX_JDEC_MAX_HEIGHT    AX_JDEC_MIN_WIDTH     AX_JDEC_MIN_HEIGHT
16384                 16384                 48                    48

----------------------- VDEC MODULE PARAM -----------------------
AX_VDEC_MAX_WIDTH     AX_VDEC_MAX_HEIGHT    AX_VDEC_MIN_WIDTH     AX_VDEC_MIN_HEIGHT
1920                  1920                  48                    48

------------------------- VDEC GRP ATTR -------------------------
Grp  DecType  InputMode  IsLink   PicWidth    PicHeight   FrmHeight   StrmBufSize  InFifoFilled  FrameBufCnt   PoolType    DisplayMode  DecMode  OutputOrder
0    H264     FRAME      True     1920        1080        0           3110400      1             10            USER        PLAYBACK     IPB      DISP

------------------------- VDEC GRP STATUS -------------------------
Grp  Fps    AvgFps  AvgTime(ms)   Width   Height  Crop_w  Crop_h  SendFrmNum  DecFrmNum   SkipFrmNum  ErrFrmNum   BlkFailNum  FreeBufNum  FrmInFifo   ProcStat
0    23.23  25.45   8.20          1920    1088    1920    1080    677         677         0           0           0           0           6           0xb0005

------------------------- VDEC PROC STATUS -------------------------
Grp  GetFrmNum   ReleaseFrmNum   OutFrmNum   DropFrmNum  FrmWaitRelease
0    0           0               669         0           2
```

【调试信息分析】

```
----------------------- AX VDEC VERSION -----------------------
[Axera version]: ax_vdec V1.8.0 Feb  2 2024 11:40:53

[Axera version]: libax_vdec.so V1.8.1 Feb  4 2024 10:24:52

================ AX_VDEC_PID: 1529 WorkingGrpNum: 1 ================

----------------------- JDEC MODULE PARAM -----------------------
AX_JDEC_MAX_WIDTH     AX_JDEC_MAX_HEIGHT    AX_JDEC_MIN_WIDTH     AX_JDEC_MIN_HEIGHT
16384                 16384                 48                    48

----------------------- VDEC MODULE PARAM -----------------------
AX_VDEC_MAX_WIDTH     AX_VDEC_MAX_HEIGHT    AX_VDEC_MIN_WIDTH     AX_VDEC_MIN_HEIGHT
1920                  1920                  48                    48
```

记录当前视频解码通道属性配置及状态信息

【参数说明】

分类	参数	描述
VDEC VERSION	Version	模块版本信息
JDEC MODULE PARAM	AX_JDEC_MAX_WIDTH	JDEC 最大解码图像宽度
	AX_JDEC_MAX_HEIGHT	JDEC 最大解码图像高度
	AX_JDEC_MIN_WIDTH	JDEC 最小解码图像宽度
	AX_JDEC_MIN_HEIGHT	JDEC 最小解码图像高度
VDEC MODULE PARAM	AX_VDEC_MAX_WIDTH	VDEC 最大解码图像宽度
	AX_VDEC_MAX_HEIGHT	VDEC 最大解码图像高度
	AX_VDEC_MIN_WIDTH	VDEC 最小解码图像宽度
	AX_VDEC_MIN_HEIGHT	VDEC 最小解码图像高度
MODULE PARAM	AX_VDEC_PID	进程 ID
	WorkingGrpNum	正在工作的通道数

```
------------------------ VDEC GRP ATTR -------------------------
Grp  DecType  InputMode  IsLink   PicWidth    PicHeight   FrmHeight   StrmBufSize  InFifoFilled  FrameBufCnt   PoolType    DisplayMode  DecMode  OutputOrder
0    H264     FRAME      False    1920        1920        0           2100811      0             7             USER        PLAYBACK     IPB      DISP
```

分类	参数	描述
VDEC GRP ATTR	Grp	通道号
	DecType	码流协议
	InputMode	输入模式。如下几种： NAL：不支持 FRAME：帧模式 SLICE：不支持 STREAM：流模式 COMPAT：不支持
	IsLink	是否 link 模式
	PicWidth	通道属性中用户配置的图像宽度
	PicHeight	通道属性中用户配置的图像高度

分类	参数	描述
	FrmHeight	通道属性中用户配置的图像输出高度，不能超过输入码流协议中描述的图像高度
	StrmBufSize	解码 buf 大小
	InFifoFilled	解码队列中待解码帧数
	FrameBufCnt	VDEC：解码输出 buf 个数，不能小于参考帧个数+1 JDEC：N/A 无意义
	PoolType	解码帧存分配方式。支持三种： COMMON：使用公共缓存池 PRIVATE：使用解码私有缓存池 USER：使用 USER 缓存池
	DisplayMode	VDEC：显示模式。PLAYBACK（回放模式），PREVIEW（预览模式） JDEC：N/A 无意义
	DecMode	VDEC：解码模式：IPB/IP/I JDEC：N/A 无意义
	OutputOrder	VDEC：解码输出顺序：显示顺序/解码顺序 JDEC：N/A 无意义

```
------------------------- VDEC GRP STATUS -------------------------
Grp  Fps    AvgFps  AvgTime(ms)   Width    Height   Crop_w   Crop_h  SendFrmNum  DecFrmNum   SkipFrmNum  ErrFrmNum   BlkFailNum  FreeBufNum  FrmInFifo   ProcStat
0    23.23  25.45   8.20          1920     1088     1920     1080    677         677         0           0           0           0           6           0xb0005
```

分类	参数	描述
VDEC GRP STATUS	Grp	通道号
	Fps	解码帧率
	AvgFps	解码平均帧率
	AvgTime	硬件解码单帧的平均时间，单位 ms

分类	参数	描述
	Width	协议解析的图像宽度
	Height	协议解析的图像高度
	Crop_w	协议解析的图像 crop 宽度
	Crop_h	协议解析的图像 crop 高度
	SendFrmNum	送解码帧帧数(流模式无效)
	DecFrmNum	已解码的帧数
	SkipFrmNum	VDEC：解码不输出的帧数统计，对应 PTS=-1 的场景 JDEC：未使用
	ErrFrmNum	解码出错的帧数
	BlkFailNum	获取输出帧缓存块失败次数
	FreeBufNum	VDEC：空闲的帧 buf 个数 JDEC：N/A 无意义
	FrmInFifo	VDEC：解码 buf 中等待输出的帧个数 JDEC：N/A 无意义
	ProcStat	VDEC：解码状态，SDK 内部统计信息 JDEC：N/A 无意义

```
-------------------------- VDEC PROC STATUS --------------------------
Grp  GetFrmNum   ReleaseFrmNum   OutFrmNum   DropFrmNum  FrmWaitRelease
0    0           0               3889        0           2
```

分类	参数	描述
VDEC PROC STATUS	Grp	通道号
	GetFrmNum	Unlink 模式下用户 GetFrame 的帧数
	ReleaseFrmNum	Unlink 模式下用户 ReleaseFrame 的帧数
	OutFrmNum	VDEC：Link 模式下送给下级的帧数，

分类	参数	描述
		Unlink 场景该值不做统计 JDEC：Link/Unlink 模式下，往下级送的帧数
	DropFrmNum	VDEC Link 非反压模式下，后级 fifo 满了的时候，解码帧丢弃的帧数
	FrmWaitRelease	VDEC：Link 模式下等待下级模块释放的帧数 JDEC：未使用

7 Dump Tool

SDK 提供了抓取解码通道输出端数据的 dump 工具，方便用户在业务运行过程中，抓取指定通道数据，判断输出端数据是否正常。

【使用方法】

工具所在路径：/opt/bin/dump_tools/ax_decoder_dump，通过执行如下命令可以查看工具具体使用方法。

```
# ./ax_decoder_dump -h
```

```
/opt/bin/dump_tools # ./ax_decoder_dump -h
DECODER DUMP TOOL. Build at Feb  1 2024 12:36:44
To see more usage, please enter: ./ax_decoder_dump -h


************************************************
Usage: ./ax_decoder_dump [GrpId] [FifoDepth] [DumpNum] [EnableLog] [FilePath]
1)GrpId: VDEC/JDEC Grp id
2)FifoDepth: input debug fifo depth,range [0,256]
3)DumpNum: how many frames user want to dump
4)EnableLog: whether enable detail dump log,default 0(0:disable 1:enable)
5)FilePath: The file path to save the dump file,default root directory
*)Example:
e.g : ./ax_decoder_dump 0 4 9
e.g : ./ax_decoder_dump 0 4 9 0 /opt/data
************************************************
```

【参数说明】

参数	描述
GrpId	解码通道号[0, 15]
FifoDepth	Debug FIFO 深度。 备注：FIFO 深度越大越容易抓取到连续的帧，但同时对业务性能的影响可能也会越大，当 FIFO 满时将会丢掉最老保存的 YUV 图像。
DumpNum	期望抓取的帧数
EnableLog	是否使能详细 dump log（默认 0）。

参数	描述
	0：不显示详细 dump log 1：显示详细 dump log
FilePath	输出文件存放的路径

【举例】

```
#./ax_decoder_dump 0 8 8 0 /opt/data/
```

```
/opt/bin/dump_tools # ./ax_decoder_dump 0 8 8 0 /opt/data/
DECODER DUMP TOOL. Build at Jan 30 2024 02:22:34
To see more usage, please enter: ./ax_decoder_dump -h

Write YUV to: /opt/data/group0_1920_1088.yuv
ax_decoder_dump run success.
/opt/bin/dump_tools #
```

抓取 VDEC 解码通道 0 的数据，Debug FIFO 深度设置为 8，抓取 8 帧，默认不显示详细 dump log，文件输出路径设置为/opt/data/。

到/opt/data 下面可以看到抓取的 group0_1920_1088.yuv 数据

```
/opt/bin/dump_tools # cd /opt/data/
/opt/data # ls -lh
total 25M
drwxr-xr-x    4 root     root        4.0K Jan 30 12:55 AXSyslog
drwxrwxr-x    2 root     root        4.0K Jan 30 02:28 audio
drwxrwxr-x    3 root     root        4.0K Jan 30 02:28 avs
-rw-r--r--    1 root     root       23.9M Jan 30 12:58 group0_1920_1088.yuv
```