H.265 PC 解码库软件

API 参考

文档版本　02

发布日期　2015-02-10

版权所有 © 深圳市海思半导体有限公司 2014-2015。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

目 录

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

插图目录

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

表格目录

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

前 言

概述

本节介绍本文档的内容、对应的产品版本、适用的读者对象、行文表达约定、历史修订记录等。

说明
本文以 Hi3516A 描述为例，未有特殊说明，Hi3516D 与 Hi3516A 一致。

产品版本

与本文档相对应的产品版本如下所示。

产品名称	产品版本
Hi3516A	V100
Hi3516D	V100

读者对象

本参考适用于程序员阅读，描述了基于海思 H.265 PC 解码库开发的各种参考信息。使用本参考的程序员应该：

- 熟练使用 C/C++语言
- 掌握基本的 Windows32 调用

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。

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

符号	说明
 危险	以本标志开始的文本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。
 警告	以本标志开始的文本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。
 注意	以本标志开始的文本表示有潜在风险，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	以本标志开始的文本能帮助您解决某个问题或节省您的时间。
说明	以本标志开始的文本是正文的附加信息，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、三级标题采用黑体。
楷体	警告、提示等内容一律用**楷体**，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。

表格内容约定

内容	说明
-	表格中的无内容单元。
*	表格中的内容用户可根据需要进行配置。

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

修改记录

修改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。

修改日期	版本	修改说明
2015-02-10	02	2.1 节，pstInitParam 参数的取值范围有修改
2014-12-20	01	添加 Hi3516D 的相关内容
2014-08-06	00B01	第 1 次临时版本发布

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

1 概述

1.1 描述范围

海思提供的 H.265 PC 解码库软件是一套高性能、高可靠性、兼容性良好的解码软件。解码库内部完成了 H.265 解码的主要流程，并对外提供了灵活简单的 API，用户可快速地开发应用程序。

解码库软件为用户提供 Windows 环境下的动态库和静态库两种调用形式，可更方便地开发应用程序。解码库的主要组件及相关说明如表 1-1 所示。

表1-1 解码库开发包组件

组件	名称	说明
API 接口	IHWVideo_Typedef.h IHW265Dec_Api.h	用户工程中，应该保证先包含 IHWVideo_Typedef.h，再包含 IHW265Dec_Api.h。
32 位静态库	hi_h265_dec_w32.lib	-
32 位动态库	hi_h265_dec_w32.lib hi_h265_dec_w32.dll	-
64 位静态库	hi_h265_dec_w64.lib	-
64 位动态库	hi_h265_dec_w64.lib hi_h265_dec_w64.dll	-
示范代码	hi_h265sample.c	以读文件解码为例，示范解码库 API 的调用方式。

用户可在多种编译环境上进行基于解码库的应用程序开发，解码库兼容微软公司的 Windows XP、Windows 7、Windows 8 等主流视窗操作系统，兼容 Intel 公司和 AMD 公司的绝大部分面向 PC 机的 CPU 芯片组。其主要开发以及运行环境说明如表 1-2 所示。

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

表1-2 解码库运行环境

分类	兼容配置	推荐配置	说明
编译器	Visual Studio 2008 Visual Studio 2010 Visual Studio 2012	Visual Studio 2010	无。
操作系统	Windows XP Windows 7 (32bit) Windows 7 (64bit) Windows 8 (32bit) Windows 8 (64bit)	Windows 7	无
硬件	Intel 酷睿系列 CPU Intel 奔腾系列 CPU AMD Ax 系列 CPU	Intel 酷睿 i5 系列 CPU	解码库有基于多媒体指令集 SSE、SSE2、SSE3、SSE4、AVX 的优化，在不支持这些指令集的机器上性能较低。

1.2 接口格式

无。

1.3 函数列表

函数	功能	页码
IHW265D_Create	创建、初始化解码器句柄。	4
IHW265D_Delete	销毁解码器句柄。	5
IHW265D_GetVersion	查询解码库版本信息。	6
IHW265D_DecodeFrame	对输入的一段码流进行解码并输出图像。	7
IHW265D_DecodeAU	对输入的一帧码流进行解码并立即输出此帧图像。	9

1.4 函数描述方式

本章用 6 个域对 API 参考信息进行描述。

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

参数域	作用
目的	简要描述 API 的主要功能。
语法	列出 API 的语法样式。
描述	简要描述 API 的工作过程。
参数	列出 API 的参数、参数说明及参数属性。
返回值	列出 API 的返回值及返回值说明。
注意	使用 API 时应注意的事项。

1.5 结构体描述方式

参数域	作用
说明	简要描述结构体所实现的功能。
定义	列出结构体的定义。
注意事项	列出结构体的注意事项。

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

2 API 函数说明

2.1 IHW265D_Create

【目的】

创建、初始化解码器句柄。

【语法】

```
INT32 IHW265D_Create(IH265DEC_HANDLE *phDecoder, IHW265D_INIT_PARAM
*pstInitParam);
```

【描述】

创建解码器句柄。在解码开始时，分配解码空间和初始化解码器相关的变量及状态，设置解码器支持的最大图像宽高、解码器支持的最大参考帧数目、解码器线程类型、输出图像顺序等解码器属性以及分配内存回调函数、释放内存回调函数、日志回调函数。

上层应用可以创建多个解码器，实现多路解码。

【参数】

参数	成员	取值范围	输入/输出	描述
phDecoder	-	-	输出	解码器句柄。
pstInitParam	uiChannelID	-	输入	通道号。
	iMaxWidth	[8,8192]	输入	图像宽度，以像素为单位。
	iMaxHeight	[8,8192]	输入	图像高度，以像素为单位。
	iMaxRefNum	[0,15]	输入	最大参考帧数。
	eThreadType	[0,1]	输入	线程类型，支持单线程和多线程。
	eOutputOrder	[0,1]	输入	输出顺序，支持解码序和显示序。

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

参数	成员	取值范围	输入/输出	描述
	pstUserData	-	输入	指向输入的用户数据，数据类型请参见 HW265D_USERDATA 定义。
	MallocFxn	-	输入	分配内存回调函数。
	FreeFxn	-	输入	释放内存回调函数。
	LogFxn	-	输入	日志输出回调函数。

【返回值】

返回值	宏定义	描述
0	IHW265D_OK	函数调用成功。
0xF0401000	IHW265D_INVALID_ARGUMENT	输入的形参不正确。
0xF0401002	IHW265D_INVALID_MAX_REF_PIC	最大参考帧数超过范围。
0xF0401003	IHW265D_INVALID_MALLOC_FXN	Malloc 回调函数指针无效。
0xF0401004	IHW265D_INVALID_FREE_FXN	Free 回调函数指针无效。
0xF0401006	IHW265D_INVALID_LOG_FXN	Log 回调函数指针无效。
0xE0404008	IHW265D_THREAD_ERROR	多线程错误。

【注意】

- 只有解码顺序和图像输出顺序一致时，才能使用解码序输出模式，一般而言，如果图像不包含 B 帧，则可以使用解码序输出模式以降低输出时延。
- 多线程解码适用于对单通道解码性能要求很高的场合，例如大分辨率高帧率场景，单个 CPU 核心性能不足的应用场景。多线程调度会有内部开销，对于多通道应用场景不推荐使用。
- 对超大图像的解码，创建解码器时应留意系统内存容量，如果内存不够，创建解码器可能会失败。

2.2 IHW265D_Delete

【目的】

销毁解码器句柄。

【语法】

```
INT32 IHW265D_Delete( IH265DEC_HANDLE hDecoder);
```


海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

【描述】

解码结束后，销毁解码器工作时分配的内存空间，以防止内存泄漏。

【参数】

参数	成员	取值范围	输入/输出	描述
hDecoder	-	-	输入	待销毁的解码器句柄。

【返回值】

返回值	宏定义	描述
0	IHW265D_OK	函数调用成功。
0xF0401000	IHW265D_INVALID_ARGUMENT	无效参数。
0xF0401001	IHW265D_DECODER_NOT_CREATE	解码器未创建。

【注意】

销毁后的句柄需要手动置空。

2.3 IHW265D_GetVersion

【目的】

查询解码库版本信息。

【语法】

```
INT32 IHW265D_GetVersion(IHWVIDEO_ALG_VERSION_STRU *pstVersion);
```

【描述】

察看解码库版本信息。

【参数】

参数	成员	取值范围	输入/输出	描述
pstVersion	cVersionChar	-	输出	解码库版本号。
	cReleaseTime	-	输出	编译时间。
	uiCompileVersion	-	输出	编译器版本号。

【返回值】

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

返回值	宏定义	描述
0	IHW265D_OK	函数调用成功。
0xF0401000	IHW265D_INVALID_ARGUMENT	无效参数。

【注意】

无。

2.4 IHW265D_DecodeFrame

【目的】

对输入的一段码流进行解码并输出图像。

【语法】

INT32 IHW265D_DecodeFrame(IH265DEC_HANDLE hDecoder, IH265DEC_INARGS *pstInArgs, IH265DEC_OUTARGS *pstOutArgs);

【描述】

本函数支持流式解码，对于以“00 00 01”为 nalu 分隔符的连续、线性 H.265 码流，用户可以任意长度配置给解码器解码。

【参数】

参数	成员	取值范围	输入/输出	描述
hDecoder	-	-	输入	解码器句柄。
pstInArgs	pStream	-	输入	码流起始地址。
	uiStreamLen	-	输入	码流长度（字节为单位）。
	uiTimeStamp	-	输入	时间戳信息。
	eDecodeMode	-	输入	解码模式： 通常使用 IH265D_DECODE； 当解码到文件尾，没有后续码流时使用 IH265D_DECODE_END
pstOutArgs	uiChannelID	-	输出	通道号。
	uiBytsConsumed	-	输出	消耗的字节数。
	uiTimeStamp	-	输出	时间戳。
	eFrameType	[0,2]	输出	帧类型。

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

参数	成员	取值范围	输入/输出	描述
	eDecodeStatus	-	输出	解码器状态。
	uiDecWidth	-	输出	图像宽度。
	uiDecHeight	-	输出	图像高度。
	uiYStride	-	输出	亮度分量的跨度。
	uiUVStride	-	输出	色度分量的跨度
	pucOutYUV	-	输出	输出 YUV 3 个分量的地址。
	pstUserData	-	输出	用户数据。
	uiCodingBytesOfCurFrm	-	输出	原始码流长度（字节数）。
	uiAspectRatioIdc	-	输出	幅形比信息
	uiSarWidth	-	输出	幅形比宽度，当 uiAspectRatioIdc 为 255 时有效。
	uiSarHeight	-	输出	幅形比高度，当 uiAspectRatioIdc 为 255 时有效。
	uiVpsNumUnitsInTick	-	输出	帧率信息
	uiVpsTimeScale	-	输出	
	stCuOutInfo	-	输出	输出当前帧信息，包含一帧中各种类型 CU 的个数。
	bIsError	[0,1]	输出	帧错误标志。

【返回值】

返回值	宏定义	含义
0	IHW265D_OK	函数执行成功，有一帧图像输出。
1	IHW265D_NEED_MORE_BITS	剩余码流不够解码一帧的数据，需要重新配置更多的码流。 解码模式为 IH265D_DECODE 时才会返回此值。
0xF0401000	IHW265D_INVALID_ARGUMENT	无效参数。

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

返回值	宏定义	含义
0xF0401001	IHW265D_DECODER_NOT_CREATE	解码器未创建。
0xF0402008	IHW265D_STREAMBUF_NULL	码流地址为空。

【注意】

在调用本函数过程中需要注意以下几点：

- 解码一段任意长度的码流。输入参数有码流缓冲区的地址 pStream 和码流字节数 uiStreamLen。正常情况下解码模式配置为 IH265D_DECODE。如果函数返回 IH265D_GETDISPLAY 则表明有一帧图像输出，用户必须在循环调用内部及时处理存储在 pstOutArgs 中的图像。
- 如果这段码流不足以解码出一帧，需要继续输入码流进行解码。
- 如果这段码流包括若干帧，需要在解码出一帧图像后反复调用此函数解码剩余码流。
- 在码流结束后，解码模式配置为 IH265D_DECODE_END 进入 flush 模式清空解码器中的残留图像，循环调用此函数直到函数返回 IH265D_NO_PICTURE，表示解码器中没有残留图像为止。
- 解码函数提供时间戳透传功能，输入的时间戳将保存在当前码流解码后的图像结构体中，并随解码图像一起输出。

2.5 IHW265D_DecodeAU

【目的】

对输入的一帧码流进行解码并立即输出图像。

【语法】

```
INT32 IHW265D_DecodeAU( IH265DEC_HANDLE hDecoder, IH265DEC_INARGS
*pstInArgs, IH265DEC_OUTARGS *pstOutArgs);
```

【描述】

本函数支持按帧解码，输入码流必须包含且仅包含一帧 H.265 码流，并且按照解码序立即输出。

【参数】

参数	成员	取值范围	输入/输出	描述
hDecoder	-	-	输入	解码器句柄。
pstInArgs	pStream	-	输入	码流起始地址。

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

参数	成员	取值范围	输入/输出	描述
	uiStreamLen	-	输入	码流长度（字节为单位）。
	uiTimeStamp	-	输入	时间戳。
	eDecodeMode	-	输入	解码模式，无效
pstOutArgs	uiChannelID	-	输出	通道号。
	uiBytsConsumed		输出	消耗的字节数。
	uiTimeStamp		输出	时间戳。
	eFrameType		输出	帧类型。
	eDecodeStatus		输出	解码器状态。
	uiDecWidth		输出	图像宽度。
	uiDecHeight		输出	图像高度。
	uiYStride		输出	亮度分量的跨度。
	uiUVStride		输出	色度分量的跨度
	pucOutYUV		输出	输出 YUV 3 个分量的地址。
	pstUserData		输出	用户数据。
	uiCodingBytesOfCurFrm	-	输出	原始码流长度（字节数）。
	uiAspectRatioIdc	-	输出	幅形比信息
	uiSarWidth	-	输出	幅形比宽度，当 uiAspectRatioIdc 为 255 时有效。
	uiSarHeight	-	输出	幅形比高度，当 uiAspectRatioIdc 为 255 时有效。
	uiVpsNumUnitsInTick	-	输出	帧率信息
	uiVpsTimeScale	-	输出	
	stCuOutInfo	-	输出	输出当前帧信息，包含一帧中各种类型 CU 的个数。
	bIsError	[0,1]	输出	帧错误标志。

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

【返回值】

返回值	宏定义	含义
0	IHW265D_OK	函数执行成功。
0xF0401000	IHW265D_INVALID_ARGUMENT	无效参数。
0xF0401001	IHW265D_DECODER_NOT_CREATE	解码器未创建。
0xF0402008	IHW265D_STREAMBUF_NULL	码流地址为空。

【注意】

在调用本函数过程中需要注意以下几点：

- 本函数输入码流必须包含且仅包含一帧视频。对超出或不足一帧视频的情况视为错误码流。
- 本函数仅支持解码序输出，并在解码之后立即输出。对于解码序和显示序不同的视频流，不能使用本函数。
- 本函数提供时间戳透传功能，输入的时间戳将保存在当前码流解码后的图像结构体中，并随解码图像一起输出。

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

3 数据类型与数据结构

3.1 通用数据类型描述

API 用到的主要数据类型定义如下：

```
typedef signed char         INT8;
typedef signed short        INT16;
typedef signed int          INT32;
typedef unsigned char       UINT8;
typedef unsigned short      UINT16;
typedef unsigned int        UINT32;
typedef           __int64   INT64;
typedef unsigned __int64    UINT64;
typedef void*               IH265DEC_HANDLE;
```

API 用到的函数指针类型定义如下：

```
typedef void *(* IHWVIDEO_ALG_MALLOC_FXN)( UINT32 uiChannelID, UINT32
uiSize);
typedef void  (* IHWVIDEO_ALG_FREE_FXN)( UINT32 uiChannelID, void *pMem);
typedef void  (* IHWVIDEO_ALG_LOG_FXN)( UINT32 uiChannelID,
IHWVIDEO_ALG_LOG_LEVEL eLevel, INT8 *pszMsg, ...);
```

3.2 数据结构描述

3.2.1 HW265D_THREADTYPE

【说明】

线程类型枚举。

【定义】

```
typedef enum tagHW265D_THREADTYPE
{
```


海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

```
    IH265D_SINGLE_THREAD = 0,   /* 单线程 */
    IH265D_MULTI_THREAD         /* 多线程 */
} HW265D_THREADTYPE;
```

【注意事项】

无。

3.2.2 HW265D_OUTPUTORDER

【说明】

输出顺序枚举。

【定义】

```
typedef enum tagHW265D_OUTPUTORDER
{
    IH265D_DECODE_ORDER = 0,    /* 按解码序输出 */
    IH265D_DISPLAY_ORDER,           /* 按显示序输出 */
} HW265D_OUTPUTORDER;
```

【注意事项】

无。

3.2.3 HW265D_DECODEMODE

【说明】

解码模式，正常解码过程中解码器内部会残留一部分码流和未输出图像，解码完毕后这部分图像要强制输出。

【定义】

```
typedef enum tagHW265D_DECODEMODE
{
    IH265D_DECODE = 0,          /* 正常解码 */
    IH265D_DECODE_END           /* 解码完毕并要求解码器输出残留图像 */
} HW265D_DECODEMODE;
```

【注意事项】

无。

3.2.4 HW265D_FRAMETYPE

【说明】

帧类型。

【定义】

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

```
typedef enum tagHW265D_FRAMETYPE
{
    IH265D_FRAME_I = 0,
    IH265D_FRAME_P,
    IH265D_FRAME_B,
    IH265D_FRAME_UNKNOWN
} HW265D_FRAMETYPE;
```

【注意事项】

无。

3.2.5 HW265D_DECODESTATUS

【说明】

解码器状态。

【定义】

```
typedef enum tagHW265D_DECODESTATUS
{
    IH265D_GETDISPLAY = 0,      /* 已经解码出一帧并输出，可用于显示 */
    IH265D_NEED_MORE_BITS,      /* 解码器没有图像输出，需要更多的码流 */
    IH265D_NO_PICTURE,          /* 解码完毕，已经输出全部图像 */
    IH265D_ERR_HANDLE           /* 句柄错误 */
} HW265D_DECODESTATUS;
```

【注意事项】

无。

3.2.6 IHW265D_INIT_PARAM

【说明】

解码器初始化参数。

【定义】

```
/* 解码器初始化参数 */
typedef struct tagIHW265D_INIT_PARAM
{
    UINT32  uiChannelID;                /* 通道号 */
    INT32   iMaxWidth;                  /* 最大宽度 */
    INT32   iMaxHeight;                 /* 最大高度 */
    INT32   iMaxRefNum;                 /* 最大参考帧个数 */
    HW265D_THREADTYPE  eThreadType;     /* 线程类型 */
    HW265D_OUTPUTORDER eOutputOrder;    /* 输出顺序，仅在DecodeFrame有效，
```


海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

```
DecodeAU默认使用解码序 */
    HW265D_USERDATA    *pstUserData;     /* 用户数据 */
    IHWVIDEO_ALG_MALLOC_FXN  MallocFxn; /* 分配内存回调函数*/
    IHWVIDEO_ALG_FREE_FXN    FreeFxn;   /* 释放内存回调函数 */
    IHWVIDEO_ALG_LOG_FXN     LogFxn;    /* 日志回调函数 */
}IHW265D_INIT_PARAM;
```

【注意事项】

无。

3.2.7 IH265DEC_INARGS

【说明】

解码器输入信息。

【定义】

```
/* 解码器输入的码流、时间戳、解码模式信息 */
typedef struct tagIH265DEC_INARGS
{
    UINT8  *pStream;                /* 码流地址 */
    UINT32 uiStreamLen;             /* 码流长度 */
    UINT64 uiTimeStamp;             /* 时间戳 */
    HW265D_DECODEMODE eDecodeMode;  /* 解码模式 0：正常模式；1：Flush模式 */
}IH265DEC_INARGS;
```

【注意事项】

无。

3.2.8 IH265DEC_OUTARGS

【说明】

解码器输出信息。

【定义】

```
/* 解码器输出信息 */
typedef struct tagIH265DEC_OUTARGS
{
    UINT32            uiChannelID;      /* 通道号 */
    UINT32            uiBytsConsumed;   /* 已经消耗的码流字节数 */
    UINT64            uiTimeStamp;      /* 时间戳 */
    HW265D_FRAMETYPE  eFrameType;       /* 输出帧类型 */
    HW265D_DECODESTATUS eDecodeStatus;  /* 解码器状态 */
```


海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

```
    UINT32            uiDecWidth;       /* 图像宽度 */
    UINT32            uiDecHeight;      /* 图像高度 */
    UINT32            uiYStride;        /* 图像亮度跨度 */
    UINT32            uiUVStride;       /* 图像色度跨度 */
    UINT8             *pucOutYUV[3];    /* 图像YUV 3个分量地址 */
    HW265D_USERDATA   *pstUserData;     /* 用户数据 */

    UINT32            uiCodingBytesOfCurFrm; /* 当前帧原始码流长度（字节数）*/
    // vui information
    UINT32            uiAspectRatioIdc;     /* 幅形比信息 */
    UINT32            uiSarWidth;
    UINT32            uiSarHeight;
    // vps information
    UINT32            uiVpsNumUnitsInTick;  /* 帧率信息 */
    UINT32            uiVpsTimeScale;
    // cuinfo
    CU_OUTPUT_INFO    stCuOutInfo;          /*  */
    // errorinfo
    BOOL32            bIsError;             /* 帧错误标志 */
}IH265DEC_OUTARGS;
```

3.2.9 HW265D_USERDATA

【说明】

用户数据。

【定义】

```
/* 用户数据 */
typedef struct tagHW265D_USERDATA
{
    UINT32  uiUserDataType;     /* 用户数据类型  */
    UINT32  uiUserDataSize;     /* 用户数据长度  */
    UINT8   *pucData;           /* 用户数据缓冲区*/

    struct tagHW265D_USERDATA  *pNext;  /* 指向下一段用户数据 */
}HW265D_USERDATA;
```

【注意事项】

无。

海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

3.2.10 IHWVIDEO_ALG_VERSION_STRU

【说明】

解码库版本信息。

【定义】

```
/* 解码库版本信息 */
typedef struct tagIHWVIDEO_ALG_VERSION
{
    INT8    cVersionChar[IHWVIDEO_ALG_VERSION_LENGTH];  /* 版本号 */
    INT8    cReleaseTime[IHWVIDEO_ALG_TIME_LENGTH];     /* 编译时间 */
    UINT32  uiCompileVersion;                           /* 编译器版本号 */
}IHWVIDEO_ALG_VERSION_STRU;
```

【注意事项】

无。

3.2.11 CU_OUTPUT_INFO

【说明】

每帧图像各种 CU 数量信息。

【定义】

```
/* 每帧图像各种CU数量信息 */
typedef struct tagCU_OUTPUT_INFO
{
    UINT32 uiCuNumIntra4;       /* intra 4x4 CU number */
    UINT32 uiCuNumIntra8;       /* intra 8x8 CU number */
    UINT32 uiCuNumIntra16;      /* intra 16x16 CU number */
    UINT32 uiCuNumIntra32;      /* intra 32x32 CU number */
    UINT32 uiCuNumIntra64;      /* intra 64x64 CU number */
    UINT32 uiCuNumPcm4;         /* IPCM 4x4 CU number */
    UINT32 uiCuNumPcm8;         /* IPCM 8x8 CU number */
    UINT32 uiCuNumPcm16;        /* IPCM 16x16 CU number */
    UINT32 uiCuNumPcm32;        /* IPCM 32x32 CU number */
    UINT32 uiCuNumPcm64;        /* IPCM 64x64 CU number */
    UINT32 uiCuNumInter8;       /* inter 8x8 CU number */
    UINT32 uiCuNumInter16;      /* inter 16x16 CU number */
    UINT32 uiCuNumInter32;      /* inter 32x32 CU number */
    UINT32 uiCuNumInter64;      /* inter 64x64 CU number */
    UINT32 uiCuNumSkip8;        /* skip 8x8 CU number */
    UINT32 uiCuNumSkip16;       /* skip 16x16 CU number */
    UINT32 uiCuNumSkip32;       /* skip 32x32 CU number */
```


海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

```
    UINT32 uiCuNumSkip64;      /* skip 64x64 CU number */
}CU_OUTPUT_INFO;
```

【注意事项】

无。

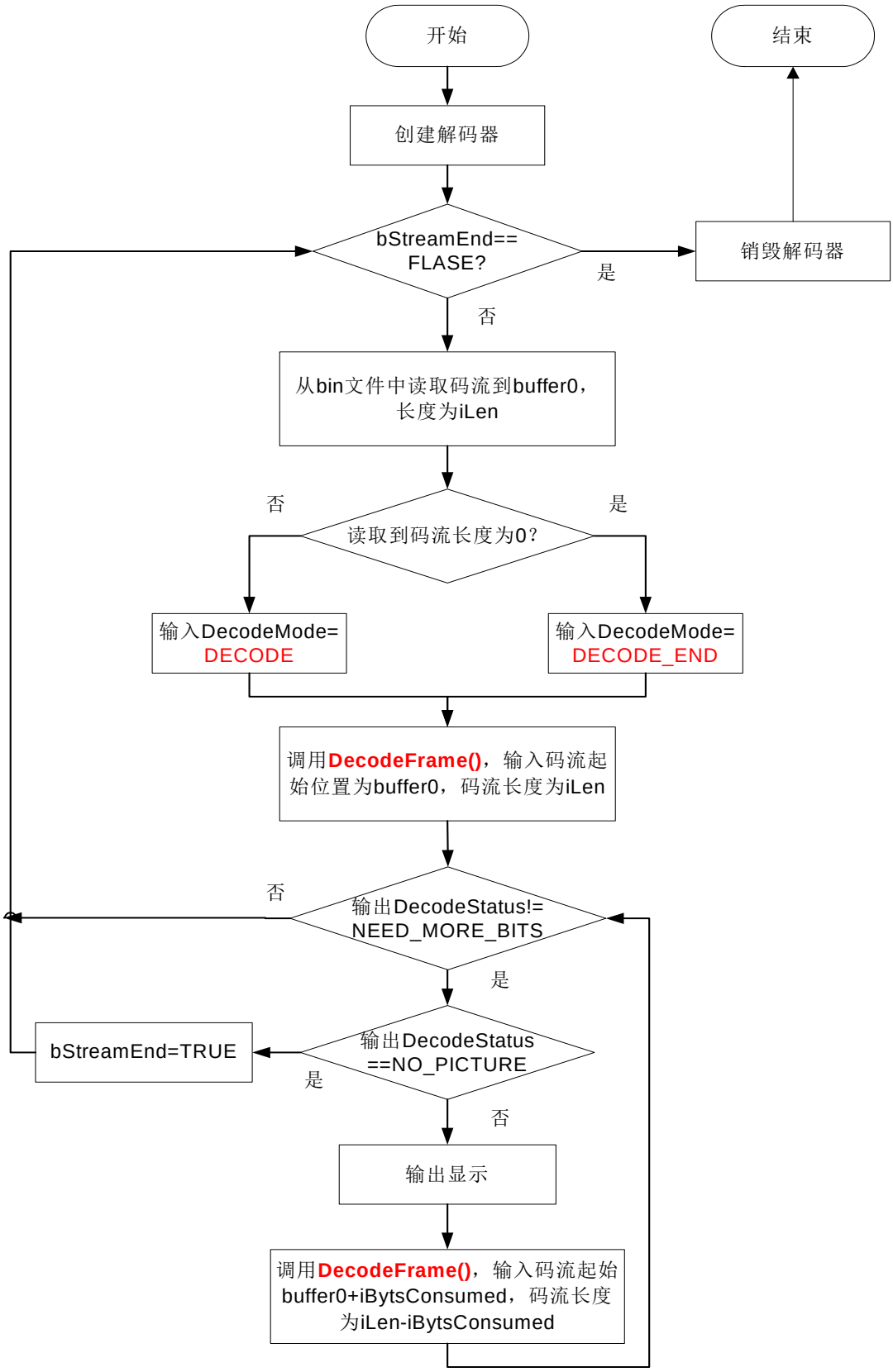
海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

4 API 应用实例

4.1 流式解码流程图

图4-1 按流方式解码 API 函数使用流程图

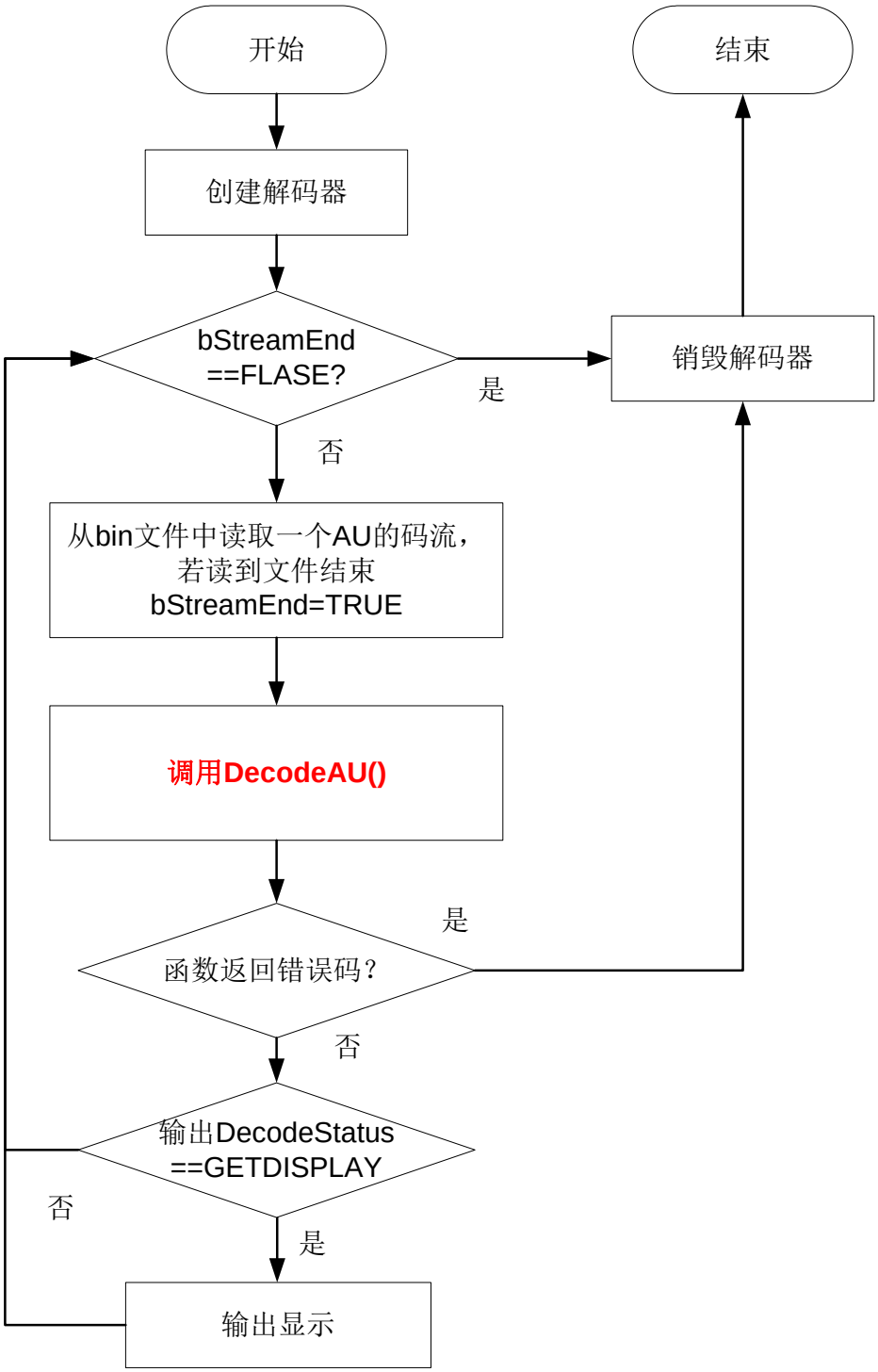
海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司



海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

4.2 按帧解码流程图

图4-2 按帧方式解码 API 函数使用流程图



海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

4.3 程序实例

```
void *HW265D_Malloc(UINT32 channel_id, UINT32 size)
{
    return (void *)malloc(size);
}

void HW265D_Free(UINT32 channel_id, void * ptr)
{
    free(ptr);
}

void HW265D_Log( UINT32 channel_id, IHWVIDEO_ALG_LOG_LEVEL eLevel, INT8
*p_msg, ...)
{
    printf("%s.\n", p_msg);
}

INT32 H265DecLoadAU(UINT8* pStream, UINT32 iStreamLen, UINT32* pFrameLen)
{
    UINT32 i;
    UINT32 state = 0xffffffff;
    BOOL32 bFrameStartFound=0;

    *pFrameLen = 0;
    if( NULL == pStream || iStreamLen <= 4)
    {
        return -1;
    }

    for( i = 0; i < iStreamLen; i++)
    {
        if( (state & 0xFFFFFF7E) >= 0x100 &&
            (state & 0xFFFFFF7E) <= 0x13E )
        {
            if( 1 == bFrameStartFound )
            {
                if( (pStream[i+1]>>7) == 1)
                {
                    *pFrameLen = i - 4;
                    return 0;
                }
            }
            else
```


海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

```
            {
                bFrameStartFound = 1;
            }
        }

        /*find a vps, sps, pps*/
        if(  (state&0xFFFFFF7E) == 0x140 ||
             (state&0xFFFFFF7E) == 0x142 ||
             (state&0xFFFFFF7E) == 0x144)
        {
            if(1 == bFrameStartFound)
            {
                *pFrameLen = i - 4;
                return 0;
            }
            else
            {
                bFrameStartFound = 1;
            }
        }

        state = (state << 8) | pStream[i];
    }

    *pFrameLen = i;
    return -1;
}


int main(int argc, unsigned char** argv)
{
    FILE *fpInFile = NULL;
    FILE *fpOutFile = NULL;
    INT32 iRet = 0;
    UINT8 *pInputStream = NULL, *pStream;
    UINT32 uiChannelId = 0;
    UINT32 iFrameIdx = 0;
    BOOL32 bStreamEnd = 0;
    INT32 iFileLen;

    IH265DEC_HANDLE hDecoder = NULL;
    IHW265D_INIT_PARAM stInitParam = {0};
    IH265DEC_INARGS stInArgs;
    IH265DEC_OUTARGS stOutArgs = {0};
```


海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

```
/* open input stream file and output yuv file  */
    fpInFile = fopen(argv[1], "rb");
    fpOutFile = fopen(argv[2], "wb");
    if (NULL == fpInFile || NULL == fpOutFile)
    {
        fprintf(stderr, "Unable to open h265 stream file %s or yuv
file %s.\n",
argv[1], argv[2]);
        goto exitmain;
    }

    printf("decoding file: %s...\n", argv[1]);
    printf("save yuv file: %s...\n", argv[2]);

    /* malloc stream buffer */
    fseek( fpInFile, 0, SEEK_END);
    iFileLen = ftell( fpInFile);
    fseek( fpInFile, 0, SEEK_SET);
    pInputStream = (unsigned char *) malloc(iFileLen);
    if (NULL == pInputStream)
    {
        fprintf(stderr, "Malloc failed! \n");
        goto exitmain;
    }

    /* create decode handle */
stInitParam.uiChannelID    = 0;
    stInitParam.iMaxWidth      = 1920;
    stInitParam.iMaxHeight     = 1088;
    stInitParam.iMaxRefNum     = 2;
    stInitParam.eThreadType    = IH265D_SINGLE_THREAD;    // or
IH265D_MULTI_THREAD;
    stInitParam.eOutputOrder   = IH265D_DECODE_ORDER;     // or
IH265D_DISPLAY_ORDER;
    stInitParam.MallocFxn      = HW265D_Malloc;
    stInitParam.FreeFxn        = HW265D_Free;
    stInitParam.LogFxn         = HW265D_Log;
    iRet = IHW265D_Create(&hDecoder, &stInitParam);
    if (IHW265D_OK != iRet)
    {
        fprintf(stderr, "Unable to create decoder.\n");
        goto exitmain;
    }
```


海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

```
    /* read H.265 stream to stream buffer */
    fread(pInputStream, 1, iFileLen, fpInFile);
    pStream = pInputStream;

    /* decode process */
    while (!bStreamEnd)
    {
        INT32 iNaluLen;
        H265DecLoadAU(pStream, iFileLen, &iNaluLen);

        stInArgs.eDecodeMode = (iNaluLen>0) ? IH265D_DECODE :
IH265D_DECODE_END;
        stInArgs.pStream     = pStream;
        stInArgs.uiStreamLen = iNaluLen;

        pStream += iNaluLen;
        iFileLen-= iNaluLen;

        stOutArgs.eDecodeStatus = -1;
        stOutArgs.uiBytsConsumed = 0;

        while(stOutArgs.eDecodeStatus != IH265D_NEED_MORE_BITS)
        {
            // when decoder is empty, exit loop;
            if(stOutArgs.eDecodeStatus == IH265D_NO_PICTURE)
            {
                bStreamEnd = 1;
                break;
            }

            // output one picture
            if (stOutArgs.eDecodeStatus == IH265D_GETDISPLAY)
            {
                // save yuv to output file
                if (fpOutFile != NULL)
                {
                    UINT32 i;

                    for (i=0; i<stOutArgs.uiDecHeight; i++)
                    {
fwrite(stOutArgs.pucOutYUV[0]+i*stOutArgs.uiYStride, 1,
stOutArgs.uiDecWidth, fpOutFile);
                    }
```


海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

```
                    for (i=0; i<((stOutArgs.uiDecHeight)>>1); i++)
                    {

    fwrite(stOutArgs.pucOutYUV[1]+i*stOutArgs.uiUVStride, 1,
                            stOutArgs.uiDecWidth>>1, fpOutFile);
                    }
                    for (i=0; i<((stOutArgs.uiDecHeight)>>1); i++)
                    {

    fwrite(stOutArgs.pucOutYUV[2]+i*stOutArgs.uiUVStride, 1,
                            stOutArgs.uiDecWidth>>1, fpOutFile);
                    }
                }
                iFrameIdx++;
            }
            // continue to decode the rest of stream
            stInArgs.pStream       += stOutArgs.uiBytsConsumed;
            stInArgs.uiStreamLen   -= stOutArgs.uiBytsConsumed;

            iRet = IHW265D_DecodeFrame(hDecoder, &stInArgs, &stOutArgs);
            if ((iRet != IHW265D_OK) && (iRet != IHW265D_NEED_MORE_BITS))
            {
                if (0 == iFileLen)
                {
                    bStreamEnd = 1;
                }
                break;
            }
        }
    }
exitmain:
    if (fpInFile != 0)
        fclose(fpInFile);

    if (fpOutFile != 0)
        fclose(fpOutFile);

    if (hDecoder != NULL)
{
        IHW265D_Delete(hDecoder);
hDecoder = NULL;
}
    if (pInputStream != NULL)
    {
```


海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司

```
            free(pInputStream);
            pInputStream = NULL;
        }

        return 0;
    }
```


海思专有和保密信息
版权所有 © 深圳市海思半导体有限公司