



Hi3520/Hi3515 媒体处理软件

## FAQ

文档版本	01
发布日期	2009-12-23
部件编码	N/A

深圳市海思半导体有限公司为客户提供全方位的技术支持，用户可与就近的海思办事处联系，也可直接与公司总部联系。

## 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)

**版权所有 © 深圳市海思半导体有限公司 2009。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

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。



## 目 录

前 言.....	1
1 FAQ.....	1-1
1.1 系统控制.....	1-1
1.1.1 如何获取当前MPP平台的版本号 .....	1-1
1.1.2 如何查看MPP的日志信息 .....	1-1
1.2 视频编码.....	1-2
1.2.1 NTSC制QCIF小码流图像被裁剪问题 .....	1-2
1.2.2 如何解决H.264 码率波动的问题 .....	1-3
1.2.3 如何通过VBR调节图像质量 .....	1-3
1.2.4 如何设置MJPEG的码率 .....	1-4
1.2.5 JPEG压缩率与图像质量的关系 .....	1-5
1.3 视频解码.....	1-6
1.3.1 如何解决H.264 解码图像输出延迟的问题 .....	1-6
1.3.2 如何解决H.264 解码器内码流残留的问题 .....	1-7
1.4 音频.....	1-7
1.4.1 Hi3520/Hi3515 音频有哪些软件调试手段 .....	1-7
1.4.2 ADPCM（IMA）音频编码时，为什么声音有异常.....	1-8
1.4.3 如何解决音频回放或编码时有少许杂音或丢帧的问题 .....	1-8



## 表格目录

表 1-1 编码图像crop示例.....	1-3
表 1-2 不同bitrate、picLevel下的VBR效果值(示例).....	1-4
表 1-3 不同规格图像的码率.....	1-5
表 1-4 JPEG图像压缩率与图像质量关系.....	1-5
表 1-5 不同解码输出方式的参数配置.....	1-6



# 前言

## 概述

本文为使用 Hi3520/Hi3515 媒体处理软件开发的程序员而写，目的是为您在开发过程中遇到的问题提供解决办法和帮助。

## 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
Hi3520 H.264 编解码处理器	V100
Hi3515 H.264 编解码处理器	V100

## 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。



符号	说明
 注意	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。
“ ”	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。
{ x   y   ... } *	表示从两个或多个选项选取多个, 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选取多个或者不选。



## 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
2009-12-23	01	第一次发布。



# 1 FAQ

## 1.1 系统控制

### 1.1.1 如何获取当前 MPP 平台的版本号

**【现象】**

需要获取当前 MPP 平台的版本号。

**【分析】**

无。

**【解决】**

- 定义一个 MPP 版本数据结构（MPP\_VERSION\_S）的变量 sVerMpp，MPP\_VERSION\_S 数据结构请参见 hi\_common.h 中的具体描述。
- 包含头文件 hi\_common.h、hi\_comm\_sys.h、mpi\_sys.h。
- 调用函数 HI\_MPI\_SYS\_GetVersion，输入该 MPP 版本数据结构变量指针 &sVerMpp，将返回表示当前 MPP 版本的字符串。
- 使用函数 `printf(“%s \n”, sVerMpp.aVersion);` 可以看到以下类似例子的内容：  
HI\_VERSION=Hi3520 MPP V1.0.3.0。

### 1.1.2 如何查看 MPP 的日志信息

**【现象】**

需要查看日志和调整 log 日志的等级。

**【分析】**

Log 日志记录 SDK 运行时错误的原因、大致位置以及一些系统运行状态等信息。因此可通过查看 log 日志，辅助错误定位。

目前日志分为 7 个等级，默认设置为等级 3。等级设置的越高，记录到日志中的信息量就越多，当等级为 7 时，会把系统的整个运行状态实时的记录到日志中，此时的信息量非常庞大，会大大降低系统的整体性能。因此，通常情况下，推荐设置为等级 3，因





为此时只有发生错误的情况下，才会将信息记录到日志中，辅助定位绝大多数的错误。

#### 【解决】

获取日志记录或修改日志等级时用到的命令如下：

- 查看各模块的日志等级，可以使用命令 **cat /proc/umap/log**，此命令会列出所有模块日志等级。
- 修改某个模块的日志等级，可使用命令 **echo "venc=4" > /proc/umap/log**，其中 **venc** 是模块名，与 **cat** 命令列出的模块名一致即可。
- 修改所有模块的日志等级，可以使用命令 **echo "all=4" > /proc/umap/log**。
- 获取日志记录，可以使用命令 **cat /dev/log**，此命令将打印出所有的日志信息；如果日志已读空，命令会阻塞并等待新的日志信息，可以使用 **Ctrl+C** 退出。也可以使用 **open**、**read** 等系统调用来操作 **/dev/log** 这个设备节点。
- 获取 Hi3520 ARM11 上的日志记录，则需要使用命令 **cat /dev/mstlog**。

## 1.2 视频编码

### 1.2.1 NTSC 制 QCIF 小码流图像被裁剪问题

#### 【现象】

无法编码 N 制 CIF 大码流（352×240）、QCIF 小码流（176×120）。

#### 【分析】

H.264 编码图像宽高需要 16 像素对齐，QCIF 小码流高度不是 16 像素对齐，所以创建通道失败，无法进行编码。

#### 【解决】

视频输入源仍然为 352×240，但创建编码通道时，大小码流分别多编码 16 像素和 8 像素（即大码流创建为 352×256、小码流创建为 176×128），这样，大码流 1/2 缩放后即可得到完整的 QCIF 小码流图像。同时依据 H.264 的 **crop** 语法，设置显示区域为（352×240）和（176×120）；用户 PC 端解码时，需要支持 **crop** 语法，否则码流图像底部会有多余无效数据。

基于同样原理，N 制 QQVGA 小码流时，也可以采用上述方案，即 QVGA 大码流创建为 320×256，QQVGA 小码流创建为 160×128。



### 注意

- 仅使用主次码流，且次码流高度为 128 时，才对大小码流同时进行剪裁操作。请根据表 1-1 进行剪裁的特殊设置。
- 推荐流程：创建通道组—>创建并注册主码流—>创建并注册次码流—>启动主次通道接收图像—>通道组绑定 VI。

表1-1 编码图像 crop 示例

编码通道尺寸		VI 尺寸	实际图像尺寸		备注
主	次		主	次	
352×256	176×128	352×240	352×240	176×120	特殊设置
704×512	176×128	704×480	704×480	176×120	特殊设置
704×480	352×240	704×480	704×480	352×240	正常设置

## 1.2.2 如何解决 H.264 码率波动的问题

### 【现象】

CBR 模式下，码率波动较大，难控制。

### 【分析】

H.264 编码提供了灵活的 CBR 码率控制策略：支持完全由 SDK 来控制码率；也支持通过设置码率波动范围的方式来控制码率。

CBR 模式下，可通过 PicLevel 的不同取值来选择控制策略：

- 0：完全由 SDK 来控制码率，码率波动范围基本为 $[-30\%, +30\%]$ 。
- 1~5：对应的码率波动范围分别为 $\pm 10\% \sim \pm 50\%$ （波动范围越大，图像主观质量越好）。在场景切换或大运动时码率会上冲。

### 【解决】

可以考虑下列解决办法：

- 降低码率波动范围，即减小 piclevel。
- 由 SDK 控制码率，即令 piclevel 等于 0，推荐使用。

## 1.2.3 如何通过 VBR 调节图像质量

### 【现象】

需要通过 VBR 调节图像质量。

### 【分析】



VBR 效果可通过 bitrate 和 picLevel 调节。VBR 效果值越大，表示效果越好。

#### 【解决】

VBR 效果的计算公式为：bitrate×[1-(picLevel/10)]

表 1-2 给出了不同 bitrate、picLevel 下的 VBR 效果值，可根据实际需要选择其他的 bitrate 和 picLevel 值（不局限于表 1-2），通过公式得到相应的 VBR 效果值，然后排序。例如选择 bitrate=6000，picLevel=0~5 等。



#### 注意

若等级较多，则相邻等级的 VBR 效果的差异并不明显。

表1-2 不同 bitrate、picLevel 下的 VBR 效果值(示例)

Bitrate (bps)	Piclevel					
	0	1	2	3	4	5
8000	8000	$8 \times (1-1/10)$	$8 \times (1-2/10)$	$8 \times (1-3/10)$	$8 \times (1-4/10)$	$8 \times (1-5/10)$
4000	4000	bitrate×[1-(picLevel/10)]（根据公式具体计算）				2000
2000	2000	bitrate×[1-(picLevel/10)]（根据公式具体计算）				1000
1000	1000	bitrate×[1-(picLevel/10)]（根据公式具体计算）				512
512	512	bitrate×[1-(picLevel/10)]（根据公式具体计算）				256
256	256	bitrate×[1-(picLevel/10)]（根据公式具体计算）				128
128	128	bitrate×[1-(picLevel/10)]（根据公式具体计算）				64

## 1.2.4 如何设置 MJPEG 的码率

推荐码率的计算公式如下：

$$\text{码率} = \frac{W \times H \times 1.5}{\text{图像压缩率}} \times 8 \times \text{帧率}$$

其中：

- W 为图像宽度
- H 为图像高度



- 图像压缩率：原始图像与编码后图像之比。该值影响图像质量，请根据需要的图像质量选择合适压缩率，具体请参见表 1-4。

以图像压缩率为 16（图像质量一般）为例，不同图像规格的码率如表 1-3 所示。

表1-3 不同规格图像的码率

图像	规格	计算结果 = 码率(bps)
QCIF	176×112[144] @ 30fps	$176 \times 112 \times 1.5 \times 8 \times 25 / 16 = 443520$
CIF	352×240[288] @ 30fps	$352 \times 240 \times 1.5 \times 8 \times 30 / 16 = 1900800$
Half-D1	704×240[288] @ 30fps	$704 \times 240 \times 1.5 \times 8 \times 30 / 16 = 3801600$
D1	704×480[576] @ 30fps	$704 \times 480 \times 1.5 \times 8 \times 30 / 16 = 7603200$
VGA	640×480 @ 30fps	$640 \times 480 \times 1.5 \times 8 \times 30 / 16 = 6912000$
SVGA	800×592 @ 30fps	$800 \times 592 \times 1.5 \times 8 \times 30 / 16 = 10656000$
ATSC	1280×720 @ 30fps	$1280 \times 720 \times 1.5 \times 8 \times 30 / 16 = 20736000^a$
SXGA	1280×1024 @ 24fps	$1280 \times 1024 \times 1.5 \times 8 \times 24 / 16 = 23592960^a$
UXGA	1600×1200 @ 15fps	$1600 \times 1200 \times 1.5 \times 8 \times 15 / 16 = 21600000^a$

a: 此处请设置为 20M，因为目前 MJPEG 最大码率为 20M。

## 1.2.5 JPEG 压缩率与图像质量的关系

表 1-4 列出了 JPEG 图像压缩率与图像质量的关系，为了得到较好的图像质量，图像压缩率一般不超过 20:1。



### 注意

表 1-4 中的数据为针对普通场景的经验数据。真实的压缩率与图像内容密切相关，细节越少，图像压缩率越高，细节越多，图像压缩率越低。

表1-4 JPEG 图像压缩率与图像质量关系

图像压缩率	图像质量主观描述
≤10:1	图像质量很好，肉眼很难区分与原图像差异。
11:1~15:1	图像质量较好。数码相机的压缩效果。
16:1~20:1	图像质量一般。可保证基本的图像质量。
21:1~30:1	图像质量较差，细节纹理模糊。
>30:1	图像质量差，图像的块边界明显。



图像压缩率	图像质量主观描述
40:1	图像质量非常差，作为最大的压缩率的参考。

## 1.3 视频解码

### 1.3.1 如何解决 H.264 解码图像输出延迟的问题

#### 【现象】

发送码流送给解码器时，解码图像输出存在较长时间的延迟。

#### 【分析】

由于 H.264 协议自身的特点，解码完成到图像输出存在较大的延迟。为解决该问题，解码器提供了三种输出方式：

- 普通输出：完全按照 264 协议输出图像。
- 快速输出：收到下一帧码流，输出当前帧图像。
- 按帧输出：收到当前帧码流，输出当前帧图像。

以上三种方式，自上而下，输出速度越来越快。但快速输出和按帧输出方式不符合协议，需正确配置解码器，才能正确解码。

#### 【解决】

创建解码通道时，通过设置不同的通道属性参数选择输出方式，设置方法如表 1-5 所示。



#### 注意

- 设置为按帧发送方式时，必须保证每次发送完整一帧码流（即调用一次 HI\_MPI\_VDEC\_SendStream 接口，必须发送完整一帧码流），否则会出现解码错误。
- RefFrameNum 和 enMode 都是静态参数，在创建通道时指定，不可以动态修改。
- 当码流本身不支持快速输出时，会出现图像显示乱序的现象。
- 当码流有错误时，并不会按帧输出，此时需要送入新码流才能输出上一帧。

表1-5 不同解码输出方式的参数配置

输出方式	通道属性		注意事项
	RefFrameNum	enMode	



输出方式	通道属性		注意事项
	RefFrameNum	enMode	
普通输出	>3 (普通输出)	H264D_MODE_STREAM (流式发送)	-
	>3 (普通输出)	H264D_MODE_FRAME (按帧发送)	用户保证按帧发送
快速输出	≤3 (快速输出)	H264D_MODE_STREAM (流式发送)	-
按帧输出	≤3 (快速输出)	H264D_MODE_FRAME (按帧发送)	用户保证按帧发送

## 1.3.2 如何解决 H.264 解码器内码流残留的问题

### 【现象】

用户采用按流方式发送码流给解码器，码流发送完毕后，调用 HI\_MPI\_VDEC\_Query 接口查询解码器内部剩余码流数目，结果一直不为 0。

### 【分析】

采用按流方式发送码流时，由于 H264 协议自身的特点，解码器需接收到新一帧的码流时，才能识别上一帧的结束。

### 【解决】

确认码流发送完毕后，发送 EOS 包给解码器，告知解码器码流已结束。具体方式请参见 sampleSendEos() 函数（包含在 sample\_vdec.c 文件中）。

## 1.4 音频

### 1.4.1 Hi3520/Hi3515 音频有哪些软件调试手段

#### 【现象】

无。

#### 【分析】

无。

#### 【解决】

- 目前主要可以通过查看 proc 信息和 DAM Buffer 信息来了解 AI、AO 的相关运行状态以及数据状态。
- 使用 cat /proc/umap/ai 或 cat /proc/umap/ao，具体信息请参见《Hi3520/Hi3515 媒体处理软件 开发参考》。



- Attribution of AI Device 区可以查看此 AI 设备的属性。
- Status of AI Device 区下的 IntCnt 表示 SIO 采集数据的 DMA 中断次数，正常情况下这个值应该持续增加，否则表示 SIO 未正确采集到音频数据。
- Status of AI Channel 区下的 intlost 表示通道数据 Buffer 满的次数，一般可以表明通道丢帧次数。
- 通过 himd 工具查看 DMA 音频 buffer 数据，物理地址可以取/proc/umap/ai 信息中得到的 DMAPhy0 或 DMAPhy1 的值，DMA buffer 中存放 SIO 一次采样的数据（64bit），按照通道排列顺序即观察各个 AI 通道的数据，工作正常的 AI 通道的数据应该会有数据上明显的变化。

## 1.4.2 ADPCM（IMA）音频编码时，为什么声音有异常

### 【现象】

使用 ADPCM（IMA）音频编码，输出后发现杂音。

### 【分析】

ADPCM 音频编码协议有两种类型：DVI4 和 IMA。使用 IMA 类型时，输入的音频帧的采样点数（帧长）应该为 81、161、241、321、481，即比 DVI4 类型时多一个采样点，否则编码后的音频码流会有异常。

### 【解决】

将 AI 的采样点数设置为 81、161、241、321 或 481。

## 1.4.3 如何解决音频回放或编码时有少许杂音或丢帧的问题

### 【现象】

解决音频回放或编码时，有少许杂音或丢帧的。

### 【分析】

音频编解码是在用户态运行，因此相对而言，更容易受到系统的整体性能影响，比如运行较多的视频编解码业务时，可能由于用户态音频线程调度不及时导致 AO 没有音频帧可播放而播放静音帧，从而产生间断性的杂音。

检查是否由于丢帧引起的杂音，可以查看 AENC 和 AO 模块的 Proc 信息中的 LostCnt 项和 BufEmp 项。

### 【解决】

- 首先要确保 ARM 上的视频编解码业务在可承受的性能范围之内，例如保证 CPU 占用率（Hi3520 时为从 ARM 的 CPU 占用率）在 85% 以内。
- 为了使音频业务承受一定的系统性能颠簸，音频各模块内部都有一定的数据缓存，可以在调用相关接口时调高其 buffer 值设置；例如将 AENC 的 u32BufSize 配置为 50，AO 的 u32FrmNum 配置为 50。



### 注意

buffer 值设置过高也可能会引起更多的延时。

- AI 的一帧采样点个数的配置（AI 公共属性中的 u32PtNumPerFrm 项），会影响到 AI 采集音频帧的中断数目，降低 u32PtNumPerFrm 会有有效的较少中断次数，进而提高音频处理的及时性。



### 注意

采样点数目过高会引入一定的延迟；G 系列语音编码时推荐配置为 320，AMR 由于算法约束只支持 160。

- 音频采集的实际采样率同样会影响到 AI 的中断次数，语音编码的推荐采样率是 8KHz，如果误配置为 32KHz，将会对音频处理性能造成严重影响。