Android10 多媒体模块说明书

1.0

2020.02.27

文档履历

版本号	日期	制/修订人	内容描述
1.0	2020.02.27		正式版本

目录

1. 前言

1.1 编写目的

为了让多媒体开发人员熟悉 A100/A133 Android 10 产品的多媒体框架，实现多媒体功能定制和简单调试。

1.2 适用范围

本模块说明适用于全志科技 A100 平台 Android 10 系统产品，其他 Android 版本系统也可参考。

1.3 相关人员

多媒体开发人员。

1.4 相关术语

- OpenMAX: 开放多媒体加速层（英语：Open Media Acceleration，缩写为 OpenMAX），一个不需要授权、跨平台的软件抽象层，以 C 语言实现的软件接口，用来处理多媒体。它是由 KhronosGroup 提出的标准，目标在于创造一个统一的接口，加速多媒体的处理。
- CedarX：全志多媒体中间件，通过 AwPlayer 对接到 Android 系统中，架构详见 "2.1 Android 多媒体框架" 。
- CedarC：全志多媒体视频编解码驱动以及 openMAX IL 层实现。
- Stream：CedarX 对多媒体协议类型访问的统一接口，支持的媒体协议类型包括：本地文件、文件描述符、RTSP、HTTP、SSL、TCP、RTMP、MMS、MMSH、MMST、MMSHTTP 等。

- Parser：CedarX 对封装格式的解析的统一接口，支持的媒体封装类型包括：ASF、TS、AVI、FLV、MKV、MOV、DASH、RTSP、HLS、PMP、OGG、MPG、MMS、MMSHTTP、M3U9、PLAYLIST、MP3、APE、FLAC、AMR、ATRAC、REMUX 等。
- Demuxer：CedarX 对媒体的 Stream 和 Parser 解析的统一接口。
- Decoder：音频，视频，字幕的解码器。
- Render：音频，视频，字幕渲染。

2. 多媒体模块框架和支持列表

2.1 Android 多媒体框架

Android 原生多媒体框架，以及全志科技的移植框架图，下图中 CedarX 是全志多媒体中间件。

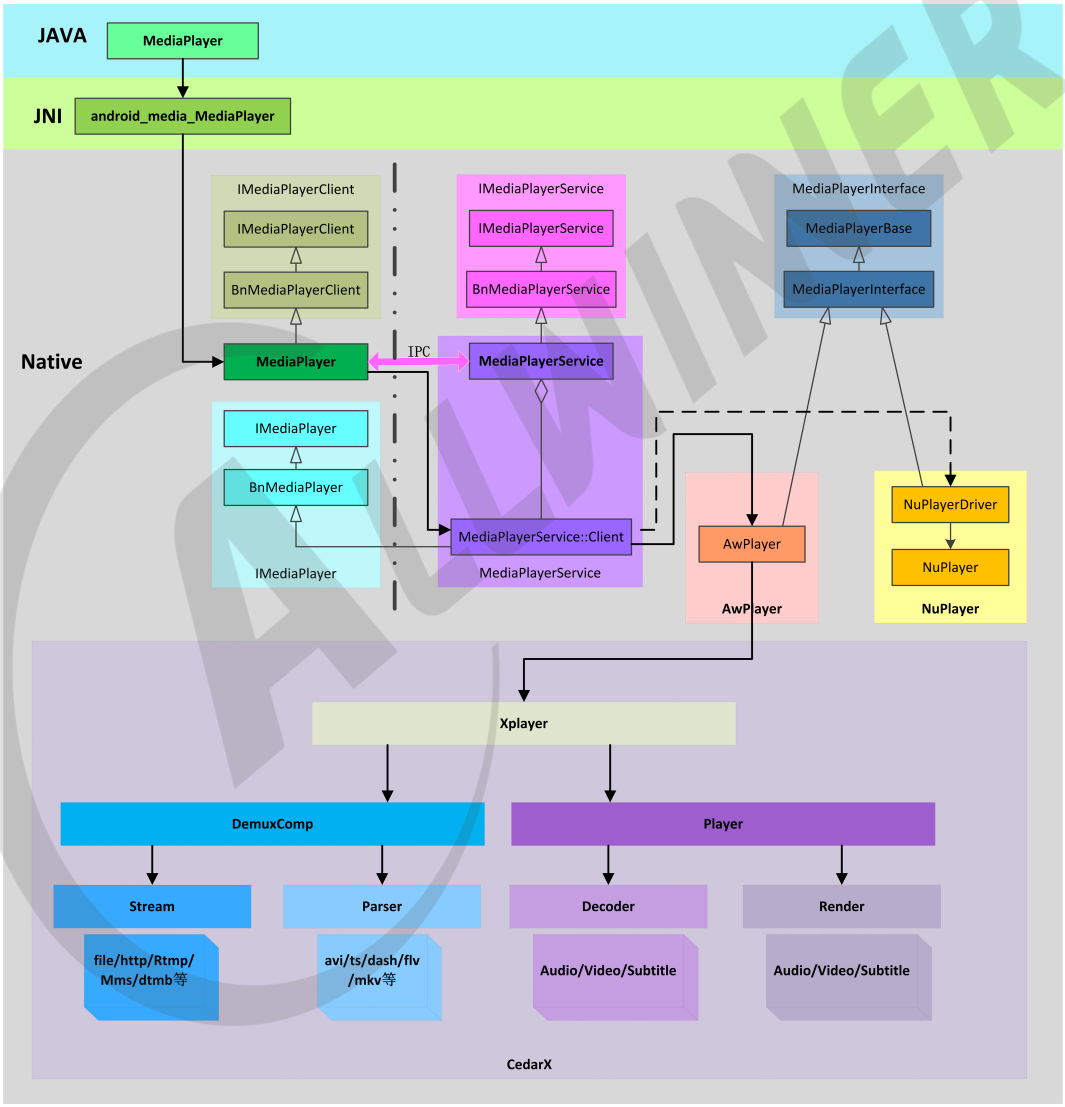

图 1: Android CedarX flows

2.2 代码结构

1、Android 多媒体模块，java 层和 jni 层代码目录:

```
android/frameworks/base/media
├── java
├── jni
├── lib
├── mca
├── native
├── OWNERS
├── packages
└── tests
```

2、Android 多媒体模块，Native 层代码目录:

```
android/frameworks/av/media
├── audioserver
├── common_time
├── extractors
├── img_utils
├── libaaudio
├── libaudioclient
├── libaudiohal
├── libaudioprocessing
├── libcedarc
├── libcedarx
├── libcpustats
├── libeffects
├── libheif
├── libmedia
├── libmediaextractor
├── libmediametrics
├── libmediaplayer2
├── libmediaplayerservice
├── libnbaio
├── libnblog
├── libstagefright
├── mediaserver
├── mtp
├── ndk
├── OWNERS
└── utils
```

3、CedarX 多媒体中间件目录:

```
android/frameworks/av/media/libcedarx
├── android_adapter
│   ├── awplayer
│   ├── base
│   ├── iptv
│   ├── metadataretriever
│   └── output
├── awrecorder
├── conf
├── config
├── demo
├── external
├── libcore
│   ├── base
│   ├── common
│   ├── muxer
│   ├── parser
│   ├── playback
│   └── stream
├── xmetadataretriever
└── xplayer
```

4、CedarC 多媒体编解码库目录:

```
android/frameworks/av/media/libcedarc
├── base
├── conf
├── config
├── include
├── library
├── memory
├── openmax
│   ├── include
│   ├── libstagefrighthw
│   ├── omxcore
│   ├── vdec
│   └── venc
├── vdecoder
└── vencoder
```

3. 多媒体支持说明

3.1 视频容器格式支持说明

默认支持如下封装格式：asf, avi, flv, f4v, mkv, mov, mp4/m4v, vob, mpg, pmp, ts/tp, m2ts, mts, wmv, webm, 3GP。

3.2 视频解码格式支持说明

默认支持如下视频格式：H.265 MP/Level5.0、H.264 Baseline/HP/MP Level5.1、MPEG1/MPEG2/MPEG4、VC-1/WMV9 SP/MP/AP、MJPEG、H.263 Baseline、VP6、VP8、VP9、WMV7/8、AVS/AVS+。具体如下图所示:

Video				
	Format	Profile	Max Resolution Ratio	Max Frame Rate
Video Decode	H.265	MP/Level5.0	4096x2048	4K@30fps@8bit
	H.264	BP, MP, HP/Level5.1	4096x2048	4K@30fps@8bit
	AVS/AVS+	JiZhun/Level6.0	1920x1080	1080p@60fps
	VP9(Soft)	/	1280x720	720p@30fps
	VP8	/	1920x1080	1080p@60fps
	MPEG1	MP/HL	1920x1080	1080p@60fps
	MPEG2	MP/HL	1920x1080	1080p@60fps
	MPEG4-XVID	SP/ASP	1920x1080	1080p@60fps
	DIVX4/5	HD	1920x1080	1080p@60fps
	DIVX3.11(Soft)	/	1280x720	720p@30fps
	H.263	BP	1920x1080	1080p@60fps
	VC-1	SP/MP/AP	1920x1080	1080p@30fps
	VP6 (Soft)	6.0/6.1/6.2	720x576	576p@30fps
	WMV7/8(Soft)	/	720x576	576p@30fps
	MJPEG	/	1920x1080	1080p@60fps
Video Encode	MJPEG	/	4096x4096	1080p@30fps
	H.264	Main/Level4.1	3840x2160	1080p@60fps

图 2: video decoder

3.3 音频容器格式支持说明

默认支持如下音频封装格式: aac, aiff, amr, ape, atrac, caf, dsd, flac, g729, mp3, ogg, wav 等。

3.4 音频解码格式支持说明

默认支持如下音频解码格式: AMR, MP1/MP2/MP3, OGG, WAV, AAC, APE, FLAC, DSD, G729, ALAC 等。

3.5 流媒体协议支持说明

默认支持如下流媒体协议：rtsp、http、https、rtmp、mms、hls。

3.6 多屏互动支持说明

默认支持 Miracast 多屏互动协议。

Miracast 是由 Wi-Fi 联盟于 2012 年所制定，以 Wi-Fi 直连为基础的无线显示标准。支持此标准的设备可通过无线方式分享视频画面，例如手机可通过 Miracast 将影片或照片直接在电视或其他装置播放而无需受到连接线缆长度的影响。

3.7 支持第三方使用 openMAX 的播放器

默认支持使用 openMAX IL 层和 openMAX AL 层 Android 原生标准接口的视频播放 apk，比如 Kodi 等。

3.8 支持多音轨切换

默认支持多音轨切换。

3.9 支持的其他播放特性

- H.265 4K@30fps@8bit
- H.264 4K@30fps@8bit
- VP9 720p@30fps

4. 模块配置使用说明

4.1 最后一帧显示黑屏配置说明

最后一帧显示黑屏是指在切台场景，切换中屏幕保持黑屏。修改方案的 CedarX 配置文件：android/frameworks/av/media/libcedarx/config/cedarx_config.go

```
var ceres_cflags = []string {
   "-DCONF_SEND_BLACK_FRAME_TO_GPU",
      ...
```

此功能配置默认关闭。

4.2 开机视频、动画使用说明

4.2.1 开机视频使用说明

将视频命名为 boot.mp4，放到/system/media/下或者/data/local/下。系统启动优先从/data/local/检测视频文件，如果没有则从/system/media/下获取，如果两个路径件都没有，则使用默认启动动画。

4.2.2 开机动画使用说明

1. 文件存放位置

开机画面都是以附件 bootanimation.zip 的形式存放在机子中。存放位置有两个：

- /system/media/bootanimation.zip
- /data/local/bootanimation.zip

读取顺序：机子开机画面或视频，优先读取/data/local/bootanimation.zip，如果发现这个 zip 包没

有或者解压有问题或者包中的文件有异常或损坏，则使用/system/media/bootanimation.zip 作为开机画面或开机视频展现。

2. 开机画面的内部结构

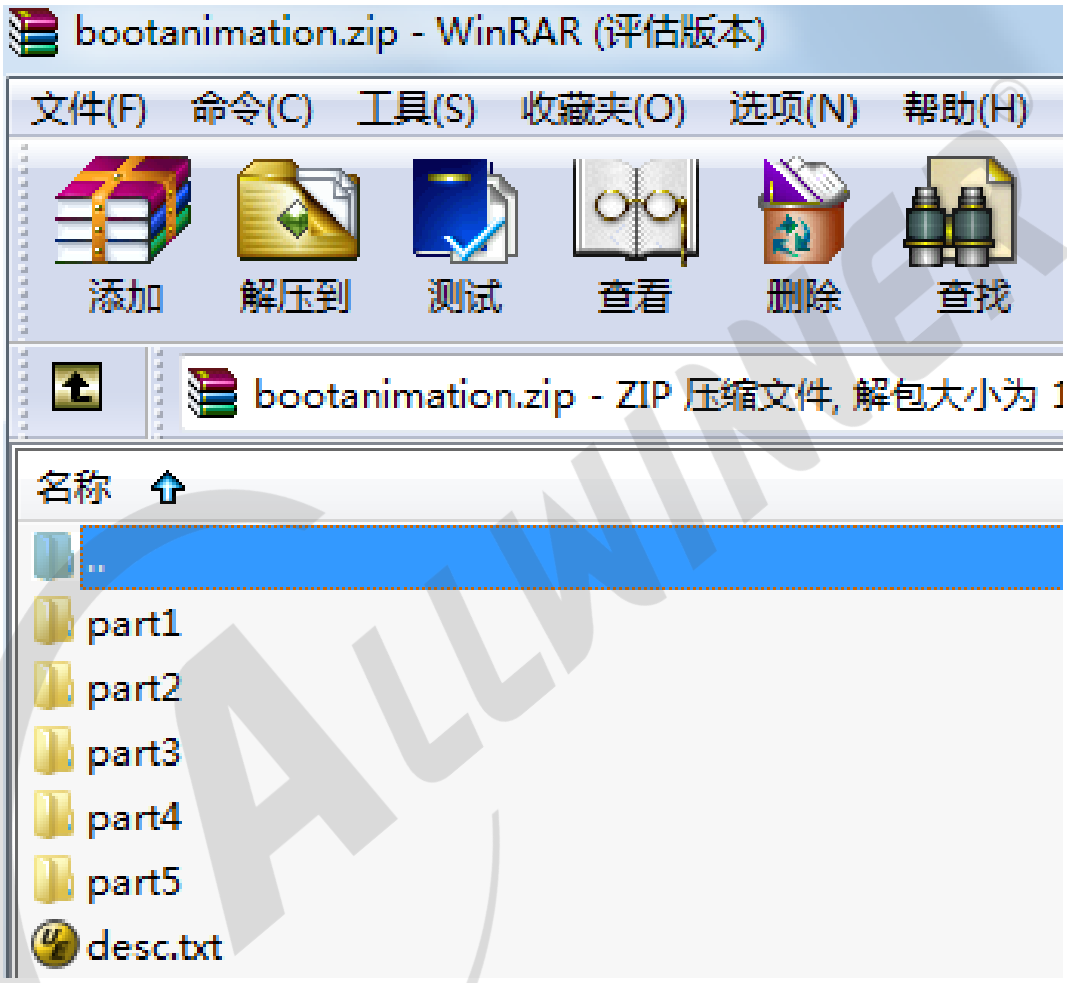

图 3: bootanim01

文件夹中存放图片，/part1 文件夹内部如下：

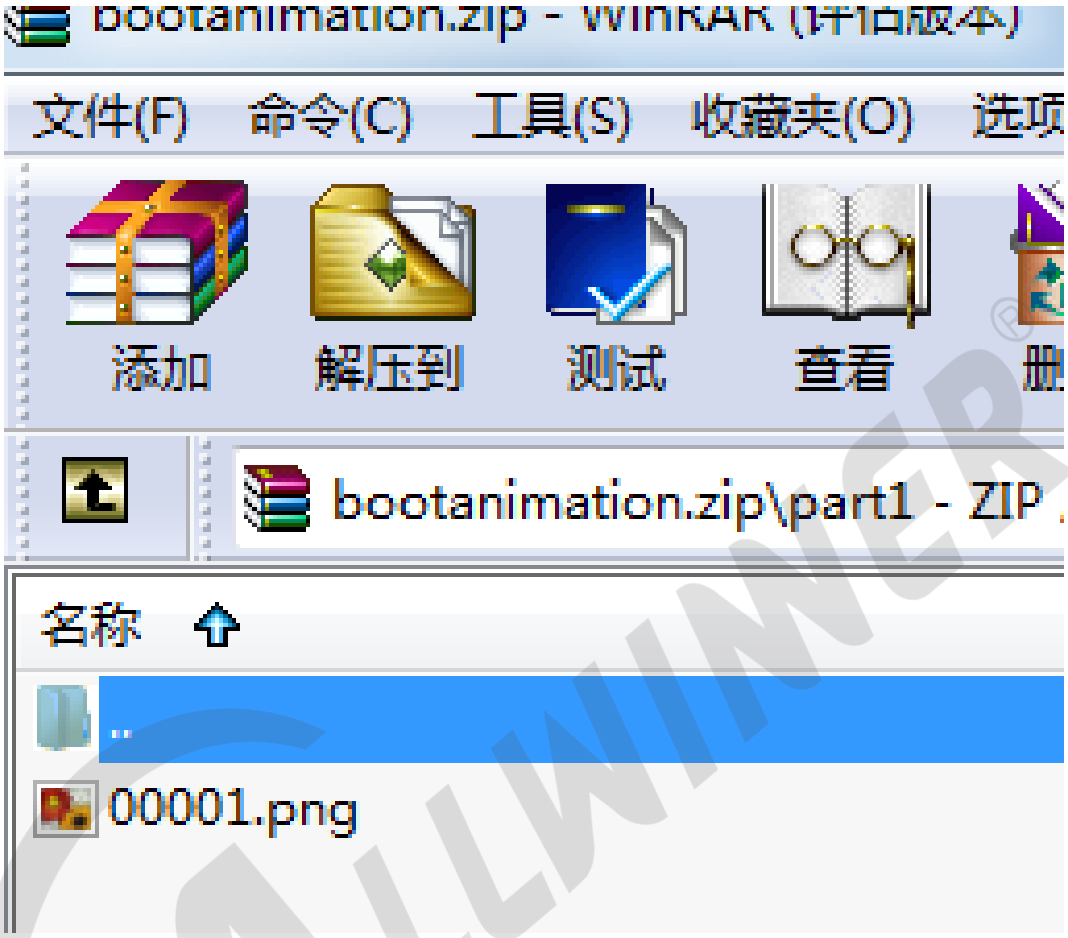

图 4: bootanim02

/desc.txt 文件及内容分析：

```
1 1280 720 1↓
2 ↓
3 p 1 5 part1↓
4 p 1 5 part2↓
5 p 1 5 part3↓
6 p 1 5 part4↓
7 p 0 0 part5↓
8 //
```

图 5: bootanim03

1280 720 是指前面文件夹里 png 的分辨率，1 是指每秒播放帧数；p 是标识符，1 5 两个数字分别指循环次数和阶段间隔时间，0 0 就代表循环播放；part1 就是文件夹的名字，文件夹的名称和存放图片的目录名一致，设计结构：上图/desc.txt 文件设置为：part1-part4 播放一次，间隔 5 秒，part5 循环播放。

4.2.3 客户开机动画视频定制使用说明

设置或者更新开机动画视频时，客户需要自行实现开机动画视频文件下载逻辑（开机动画 bootanimation.zip 或开机视频 boot.mp4），把文件存放到制定目录即可。如果已经有开机动画视频文件，覆盖即可。

5. 调试说明

5.1 如何修改多媒体中间件打印等级

针对多媒体音视频播放问题，客户可以修改 CedarX 配置文件打印等级，增加打印：android/frameworks/av/media/libcedarx/conf/ceres_cedarx.conf

```
# log will output if level >= log_level
#VERBOSE = 2,
#DEBUG = 3,
#INFO = 4,
#WARNING = 5,
#ERROR = 6,
log_level = 3
```

5.2 如何简单定位 MediaPlayer 播放问题

遇到播放问题，可以先将 log_level 改为 VERBOSE，反馈给 FAE。

5.2.1 不能播放问题的定位

第一步：播放器有没有创建成功

查看 adb logcat，搜集 CedarX 打印

```
D/awplayer( 7669): <:34>:
D/awplayer( 7669): >>>>>>>>>>>>>>>>>>>>>>>>>>> CedarX <<<<<<<<<<<<<<<<<<<<<<<<<<
D/awplayer( 7669): tag  : CedarX-2.8.0
D/awplayer( 7669): branch: master
D/awplayer( 7669): commit: 967535b8ff6a073cb4f38e85a4ae5fa6008014d8
D/awplayer( 7669): date  : Mon, 15 May 2017 01:30:22 +0000 (09:30 +0800)
D/awplayer( 7669): author:
D/awplayer( 7669): ------------------------------------------------------------
```

查看 adb logcat，搜集 CedarC 打印

```
D/cedarc  ( 7669): >>>>>>>>>>>>>>>>>>>>>>>> Cedar Codec <<<<<<<<<<<<<<<<<<<<<<<<
D/cedarc  ( 7669): tag   : CedarC-v1.2.0
D/cedarc  ( 7669): branch: master
D/cedarc  ( 7669): commit: 15195bdd453409acd8c6b3906389325650800c22
D/cedarc  ( 7669): date  : Tue May 7 11:10:20 2019 +0800
D/cedarc  ( 7669): author: jenkins8080
D/cedarc  ( 7669): patch :
D/cedarc  ( 7669): ------------------------------------------------------------
```

第二步：parser 有没有解析成功

```
D/demuxComponent( 1515): <PrintMediaInfo:426>: *****PrintMediaInfo begin*****
D/demuxComponent( 1515): <PrintMediaInfo:440>: fileSize = 47174592, bSeekable = 1,
duration = 202996, audioNum = 1, videoNum = 1, subtitleNum = 0
D/demuxComponent( 1515): <PrintMediaInfo:458>: **Video[0]** eCodecFormat = 0x115,
nWidth = 720, nHeight = 480, nFrameRate = 24000,
nFrameDuration = 0, bIs3DStream = 0
D/demuxComponent( 1515): <PrintMediaInfo:476>: ***Audio[0]*** eCodecFormat = 0x3,
eSubCodecFormat = 0x0, nChannelNum = 2,
nBitsPerSample = 0, nSampleRate = 44100
D/demuxComponent( 1515): <PrintMediaInfo:492>: ****PrintMediaInfo end*****
```

没有如上类似的打印，说明 parser 没有解析成功。如果需要的话，搜集 CedarC 和 CedarX 打印，反馈给 FAE 定位问题。

5.2.2 播放异常定位

播放的处理流程如下：

```
        ——> AudioDecoder ——> AudioRender
Demuxer ——|
        ——> VideoDecoder ——> VideoRender
```

播放异常的分析流程与之相反：Render —> Decoder —> Demuxer

1、查看 Audio/Video Render 是否正常

android/frameworks/av/media/libcedarx/libcore/playback/player.c 文件

```
diff --git a/media/libcedarx/libcore/playback/player.c b/media/libcedarx/libcore/playback/player.c
index 0e0baa0..304ae85 100644
--- a/media/libcedarx/libcore/playback/player.c
+++ b/media/libcedarx/libcore/playback/player.c
@@ -3415,7 +3415,7 @@ static int CallbackProcess(void* pSelf, int eMessageId, void* param)
         nCurTime  = p->pAvTimer->GetTime(p->pAvTimer);
         nTimeDiff = nVideoPts - nCurTime;

-        logv("notify video pts = %" PRId64 " ms, curTime = %" PRId64 " ms, diff = %"
+        logd("notify video pts = %" PRId64 " ms, curTime = %" PRId64 " ms, diff = %"
           PRId64 " ms", nVideoPts/1000, nCurTime/1000, nTimeDiff/1000);

         if (p->onResetNotSync)
@@ -3614,7 +3614,7 @@ static int CallbackProcess(void* pSelf, int eMessageId, void* param)

         }

-        logv("notify audio pts %" PRId64 " ms, curTime %" PRId64 " ms, diff %" PRId64
+        logd("notify audio pts %" PRId64 " ms, curTime %" PRId64 " ms, diff %" PRId64
           " ms, cacheTime %" PRId64 " ms", nAudioPts/1000, nCurTime/1000,
           nTimeDiff/1000, nCachedTimeInSoundDevice/1000);
```

分析打印，看看 Audio 和 video 的 pts 是否正常。

2、查看 Audio/Video Decoder 是否正常

修改 android/frameworks/av/media/libcedarx/libcore/playback/audioDecComponent.c 文件以及 android/frameworks/av/media/libcedarx/libcore/playback/videoDecComponent.c 文件。

```
diff --git a/media/libcedarx/libcore/playback/audioDecComponent.c b/media/libcedarx/libcore/playback/audioDecComponent.c
index da72249..ab00299 100755
--- a/media/libcedarx/libcore/playback/audioDecComponent.c
+++ b/media/libcedarx/libcore/playback/audioDecComponent.c
@@ -1312,7 +1312,7 @@ static void doDecode(AwMessage *msg, void *arg)
                     &p->pStreamInfoArr[p->nStreamSelected],
                     pOutputBuf,
                     &nPcmDataLen);
-    logv("DecodeAudioStream, ret = %d",ret);
+    logd("DecodeAudioStream, ret = %d",ret);
     if(ret == ERR_AUDIO_DEC_NONE)
     {
         if(p->pStreamInfoArr[p->nStreamSelected].nSampleRate != p->bsInfo.out_samplerate ||
diff --git a/media/libcedarx/libcore/playback/videoDecComponent.c b/media/libcedarx/libcore/playback/videoDecComponent.c
index 8b050ad..6bec2e4 100755
--- a/media/libcedarx/libcore/playback/videoDecComponent.c
```

```
+++ b/media/libcedarx/libcore/playback/videoDecComponent.c
@@ -820,7 +820,7 @@ static void doDecode(AwMessage *msg, void *arg)
                         p->bConfigDropDelayFrames,
                         nCurTime);

-    logv("DecodeVideoStream return = %d, p->bCrashFlag(%d)", ret, p->bCrashFlag);
+     logd("DecodeVideoStream return = %d, p->bCrashFlag(%d)", ret, p->bCrashFlag);

     if(ret == VDECODE_RESULT_NO_BITSTREAM)
     {
```

分析解码库的返回值意义：

- VDECODE_RESULT_FRAME_DECODED(1): 解码成功，输出了一帧图像；
- VDECODE_RESULT_CONTINUE(2): 码流被解码，但没有图像输出，需继续解码；
- VDECODE_RESULT_KEYFRAME_DECODED(3): 解码成功，输出了一帧关键帧图像；
- VDECODE_RESULT_NO_FRAME_BUFFER(4): 当前无法获取到图像 Buffer；
- VDECODE_RESULT_NO_BITSTREAM(5): 当前无法获取到码流数据；
- VDECODE_RESULT_RESOLUTION_CHANGE(6): 视频分辨率发生变化；
- VDECODE_RESULT_UNSUPPORTED(-1): 不能支持的格式或申请内存失败，无法继续解码；

3、查看 Demuxer 给解码库 submit Audio/Video 数据是否正常

android/frameworks/av/media/libcedarx/libcore/playback/player.c 文件

```
diff --git a/media/libcedarx/libcore/playback/player.c b/media/libcedarx/libcore/playback/player.c
index 0e0baa0..7a2416c 100644
--- a/media/libcedarx/libcore/playback/player.c
+++ b/media/libcedarx/libcore/playback/player.c
@@ -1365,7 +1365,7 @@ int PlayerSubmitStreamData(Player*          pl,

     p = (PlayerContext*)pl;

-    logv("submit stream data, eMediaType = %d", eMediaType);
+     logd("submit stream data, eMediaType = %d", eMediaType);

     if(p->eStatus == PLAYER_STATUS_STOPPED)
```

根据这句打印，判断 Demuxer 后的音视频数据是否送给解码库，每笔数据的 PTS。其中 eMediaType 含义如下。

```
enum EMEDIATYPE
{
  MEDIA_TYPE_VIDEO = 0,
  MEDIA_TYPE_AUDIO,
  MEDIA_TYPE_SUBTITLE
};
```

经过以上三步分析，可以初步定位出播放异常问题属于 Render —> Decoder —> Demuxer（Audio/Video/Subtitle）哪个模块。

5.3 如何保存解码前的码流和解码后的图片

目前有三种方式可以保留解码前的码流和解码后的图片。

方式一：设置属性（推荐），适用于 Android 10 系统。

保存解码前的码流操作如下：

1. adb shell 下在 data 目录下新建 camera 目录，若已存在，则可忽略；

   cd data && mkdir camera

2. adb shell 下关闭 selinux 系统，注意，只有 eng 固件才可以关闭 selinux 系统；

   setenforce 0

3. adb shell 下设置保存码流属性，此后，播放视频即可保存码流，可使用 ffplay.exe 等工具播放 avc/hevc 码流

   setprop vendor.vdec.savestream 1

保存解码后的图片操作步骤如下：

1. adb shell 下在 data 目录下新建 camera 目录，若已存在，则可忽略；

cd data && mkdir camera

2. adb shell 下关闭 selinux 系统，注意，只有 eng 固件才可以关闭 selinux 系统

   setenforce 0

3. adb shell 下设置保存图片的起始帧与总数帧的属性，譬如，设置起始帧为第 1 帧，保存的图片总数为 100 张，此后，播放视频即可保存图片，解码出来的图片，根据格式使用 YUV 或 NV 工具查看。

   setprop vendor.vdec.savepic.startnum 0

   setprop vendor.vdec.savepic.totalnum 100

   方式二：打开宏，推库，适用于 Android 版本系统

   android/frameworks/av/media/libcedarc/vdecoder/vdecoder.c 文件

```
diff --git a/media/libcedarc/vdecoder/vdecoder.c b/media/libcedarc/vdecoder/vdecoder.c
index 47c1cdd..9346fdc 100755
--- a/media/libcedarc/vdecoder/vdecoder.c
+++ b/media/libcedarc/vdecoder/vdecoder.c
@@ -32,8 +32,8 @@
 #include <sys/ioctl.h>
 #include <fcntl.h>

-#define DEBUG_SAVE_BITSTREAM    (0)
-#define DEBUG_SAVE_PICTURE      (0)
+#define DEBUG_SAVE_BITSTREAM    (1) //* 保存解码前的码流
+#define DEBUG_SAVE_PICTURE      (1) //* 保存解码出来的图片

 /* show decoder speed */
 #define AW_VDECODER_SPEED_INFO  (0)
```

修改源代码之后，编译，将编译后的/system/lib/libvdecoder.so 以及/vendor/lib/libvdecoder.so 推送打牌小机相应目录，并通过重启或 adb shell stop/start 等方式重载库，播放视频，即可保存码流与图片。

方式三：修改小机配置文件，适用于 Android 10 系统下通过 CedarX 多媒体中间件播放的情况。

修改小机/etc 或者/system/etc 目录下的 cedarc.conf 文件，如下所示：

```
diff --git a/media/libcedarc/conf/cedarc.conf b/media/libcedarc/conf/cedarc.conf
index 458668e..ef6054e 100755
--- a/media/libcedarc/conf/cedarc.conf
+++ b/media/libcedarc/conf/cedarc.conf
@@ -1,7 +1,7 @@
########### paramter ############
[paramter]
# save bitstream in vdecoder.c
-vdecoder_save_bitstream = 0
+vdecoder_save_bitstream = 1
vdecoder_save_bitstream_path = /data/camera/bs.dat

# save special bitstream in vdecoder.c
@@ -9,9 +9,9 @@ vdecoder_save_special_bitstream = 0
vdecoder_save_special_bitstream_path = /data/camera/spec.awsp

# save picture from start_num to start_num + total_num
-vdecoder_save_picture_en = 0
+vdecoder_save_picture_en = 1
vdecoder_save_picture_start_num = 0
-vdecoder_save_picture_total_num = 10
+vdecoder_save_picture_total_num = 100
vdecoder_save_picture_path = /data/camera/pic.dat

vdecoder_show_speed_info = 0
```

修改之后即可播放视频，保存码流与图片在相应目录下，默认为/data/camera 目录。

5.4 如何保存 MediaCodec 编码好的数据

android/frameworks/av/media/libcedarc/openmax/venc/omx_venc.cpp 文件

```
diff --git a/media/libcedarc/openmax/venc/omx_venc.c b/media/libcedarc/openmax/venc/omx_venc.c
index 51dd1f7..9d1165f 100644
--- a/media/libcedarc/openmax/venc/omx_venc.c
+++ b/media/libcedarc/openmax/venc/omx_venc.c
@@ -41,7 +41,7 @@
 #include "omx_venc.h"
 #include "omx_venc_adapter.h"

-#define SAVE_BITSTREAM (0)
+#define SAVE_BITSTREAM (1)^M
 #define ION_DEV_NAME                    "/dev/ion"
 #define DEFAULT_BITRATE (1024*1024*2)
```

```
#define OPEN_STATISTICS (0)
```

5.5 如何统计 MediaCodec 解码效率

android/frameworks/av/media/libcedarc/openmax/vdec/inc/omx_vdec_config.h 文件

```
diff --git a/media/libcedarc/openmax/vdec/inc/omx_vdec_config.h b/media/libcedarc/openmax/vdec/inc/omx_vdec_config.h
index 7746ac4..d8f431c 100644
--- a/media/libcedarc/openmax/vdec/inc/omx_vdec_config.h
+++ b/media/libcedarc/openmax/vdec/inc/omx_vdec_config.h
@@ -29,9 +29,9 @@

 #define ENABLE_SAVE_PICTURE                    0

-#define ENABLE_SHOW_BUFINFO_STATUS             0
+#define ENABLE_SHOW_BUFINFO_STATUS             1 //* 统计buffer信息

-#define ENABLE_STATISTICS_TIME                 0
+#define ENABLE_STATISTICS_TIME                 1 //* 统计解码效率
```

5.6 如何处理解码器内部问题

打开保存码流文件 special.awsp 的调试宏，把保存的数据反馈给 FAE。

注意：如果保存失败，请查看当前机子是否有"/data/camera/" 目录，权限是否正确。

```
diff --git a/media/libcedarc/vdecoder/vdecoder.c b/media/libcedarc/vdecoder/vdecoder.c
index 904d78d..6a09462 100644
--- a/media/libcedarc/vdecoder/vdecoder.c
+++ b/media/libcedarc/vdecoder/vdecoder.c
@@ -48,7 +48,7 @@
 #define DEBUG_SAVE_FRAME_TIME   (0)
 #define DEBUG_MAX_FRAME_IN_LIST 16

-#define DEBUG_MAKE_SPECIAL_STREAM (0)
+#define DEBUG_MAKE_SPECIAL_STREAM (1)
 #define SPECIAL_STREAM_FILE "/data/camera/special.awsp"
```

```
#if DEBUG_SAVE_BITSTREAM
const char* fpStreamPath = "/data/camera/bitstream.dat";
```

6. FAQ

6.1 为什么有些音视频不能播放？

1、判断封装格式、流媒体协议是否支持

2、判断音频/视频的 Codec 格式是否支持

有些音频/视频格式，属于版权视频，需要查看支持列表，看看是否支持。如果需要支持，请咨询 FAE。

3、判断视频 Codec 规格是否支持对应 size 格式播放

比如：Video 格式：H265 4K@30fps

如果 A100 规格支持，又不能播放，可以尝试使用 5.2 章节的方法定位问题，反馈给 FAE。

7. Declaration

This document is the original work and copyrighted property of Allwinner Technology (“Allwinner”). Reproduction in whole or in part must obtain the written approval of Allwinner and give clear acknowledgement to the copyright owner. The information furnished by Allwinner is believed to be accurate and reliable. Allwinner reserves the right to make changes in circuit design and/or specifications at any time without notice. Allwinner does not assume any responsibility and liability for its use. Nor for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Allwinner. This document neither states nor implies warranty of any kind, including fitness for any particular application.tates nor implies warranty of any kind, including fitness for any particular application.