

TELECHIPS TRANS-CODER MANUAL

Telechips_Trans-Coder_Manual

Rev. 0.01

Nov, 2011

TeleChips

DISCLAIMER

All information and data contained in this material are without any commitment, are not to be considered as an offer for conclusion of a contract, nor shall they be construed as to create any liability. Any new issue of this material invalidates previous issues. Product availability and delivery are exclusively subject to our respective order confirmation form; the same applies to orders based on development samples delivered. By this publication, Telechips, Inc. does not assume responsibility for patent infringements or other rights of third parties that may result from its use.

Further, Telechips, Inc. reserves the right to revise this publication and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes.

No part of this publication may be reproduced, photocopied, stored on a retrieval system, or transmitted without the express written consent of Telechips, Inc.

This product is designed for general purpose, and accordingly customer be responsible for all or any of intellectual property licenses required for actual application. Telechips, Inc. does not provide any indemnification for any intellectual properties owned by third party.

Telechips, Inc. can not ensure that this application is the proper and sufficient one for any other purposes but the one explicitly expressed herein. Telechips, Inc. is not responsible for any special, indirect, incidental or consequential damage or loss whatsoever resulting from the use of this application for other purposes.

COPYRIGHT STATEMENT

Copyright in the material provided by Telechips, Inc. is owned by Telechips unless otherwise noted.

For reproduction or use of Telechips' copyright material, permission should be sought from Telechips. That permission, if given, will be subject to conditions that Telechips' name should be included and interest in the material should be acknowledged when the material is reproduced or quoted, either in whole or in part. You must not copy, adapt, publish, distribute or commercialize any contents contained in the material in any manner without the written permission of Telechips. Trade marks used in Telechips' copyright material are the property of Telechips.

Important Notice

This product may include technology owned by Microsoft Corporation and in this case it cannot be used or distributed without a license from Microsoft Licensing, GP.

For customers who use licensed Codec ICs and/or licensed codec firmware of mp3:

"Supply of this product does not convey a license nor imply any right to distribute content created with this product in revenue-generating broadcast systems (terrestrial. Satellite, cable and/or other distribution channels), streaming applications(via internet, intranets and/or other networks), other content distribution systems(pay-audio or audio-on-demand applications and the like) or on physical media(compact discs, digital versatile discs, semiconductor chips, hard drives, memory cards and the like). An independent license for such use is required. For details, please visit <http://mp3licensing.com>".

For customers who use other firmware of mp3:

"Supply of this product does not convey a license under the relevant intellectual property of Thomson and/or Fraunhofer Gesellschaft nor imply any right to use this product in any finished end user or ready-to-use final product. An independent license for such use is required. For details, please visit <http://mp3licensing.com>".

For customers who use Digital Wave DRA solution:

"Supply of this implementation of DRA technology does not convey a license nor imply any right to this implementation in any finished end-user or ready-to-use terminal product. An independent license for such use is required."

For customers who use DTS technology:

"Supply of this implementation of DTS technology does not convey a license, exhaust DTS' rights in the implementation, or imply a right under any patent, or any other industrial or intellectual property right of DTS to use, offer for sale, sell, or import such implementation in any finished end-user or ready-to-use final product. Notice is hereby provided that a license from DTS is required prior to such use."

"This product made under license to U.S. Patents 5,451,942; 5,956,674; 5,974,380; 5,978,762; 6,487,535; 6,226,616 and/or foreign counterparts."

"© 1996 – 2010 DTS, Inc."

For customers who use Dolby technology:

"Supply of this Implementation of Dolby technology does not convey a license nor imply a right under any patent, or any other industrial or intellectual property right of Dolby Laboratories, to use this Implementation in any finished end-user or ready-to-use final product. It is hereby notified that a license for such use is required from Dolby Laboratories."

Revision History

Date	Version	Description

TABLE OF CONTENTS

Contents

1 Introduction.....	1-1
2 TCC TRANS-CODER APIS.....	2-1
2.1 TCC_HLS_Trscoder_Init.....	2-1
2.2 TCC_HLS_Trscoder_Start.....	2-2
2.3 TCC_HLS_Trscoder_Stop.....	2-3
2.4 TCC_HLS_SendAudioInfo	2-4
2.5 TCC_HLS_SendVideoInfo	2-5
2.6 TCC_HLS_WriteVideoFrame	2-6
2.7 TCC_HLS_WriteAudioFrame	2-7
2.8 TCC_HLS_SetOutputCallback	2-8
3 How to use TCC Trans-Coder in android media framework.....	3-9
3.1 Examples with TCC Trans-Coder library	3-10

1 Introduction

This document show how to use Telechips Trans-coder library(libtrscoder.so). Currently TCC Trans-Coder(Telechips Trans-Coder) supports H264 and AAC only. The output of TCC Trans-Coder is TS stream format. User should register output callback function. TCC Trans-Coder call User output callback function if there is proper TS data. To use this library properly, User should combine TCC Trans-Coder with android media framework. We added member functions on awesome player for hooking video and audio output. **If User uses DDR3 memory, The performance might be bad. We recommend User should use DDR2 memory for better performance.**

2 TCC TRANS-CODER APIS

2.1 TCC_HLS_Trscoder_Init

It initializes Trans-Coder module(Initialize Audio/Video encoder, TS muxer). User can select bitrate(bps), frame-rate(fps), key-frame interval and video out width/height. If keyframe_interval is 0, TCC Trans-Coder encode according to source frame rate(uiPicType@TCC_HLS_WriteVideoFrame).

Syntax

```
int TCC_HLS_Trscoder_Init(  
    TrscoderInfo *stream  
)
```

Parameters

stream

Initial setting value for TCC Trans-Coder. It defined at tcc_mediastream.h

Return

If there is no error, it returns 0..

Remarks

```
struct _TrscoderInfo  
{  
    unsigned int trscoder_status;  
    unsigned int bps;  
    unsigned int fps;  
    unsigned int keyframe_interval;  
    unsigned int target_width;  
    unsigned int target_height;  
};  
trscoder_status  
    for the future purpose (don't use currently)  
bps  
    target bitrate, unit is bps(if you want to target bitrate as 1Mbps, You should set bps = 1000000.  
fps  
    for the future purpose. target fps(frame-rate) is same as source fps.  
keyframe_interval  
    target key frame interval. If this value set to 0, target key frame interval is same as source.  
target_width  
    target video width for ouput  
target_height  
    target video height for output
```

2.2 TCC_HLS_Trscoder_Start

Start TCC Trans-Coder. The Trans-Coder scheduler is starting inside TCC Trans-Coder.

Syntax

```
int TCC_HLS_Trscoder_Start(void)
```

Parameters**Return**

If there is no error, it returns 0.

Remarks

2.3 TCC_HLS_Trscoder_Stop

Stop TCC Trans-Coder. The Trans-Coder scheduler is stopping inside TCC Trans-Coder.

Syntax

```
int TCC_HLS_Trscoder_Stop(void)
```

Parameters

Return

If there is no error, it returns 0.

Remarks

2.4 TCC_HLS_SendAudioInfo

Inform source video information. This information is needed by trnas-coder.

Syntax

```
int TCC_HLS_SendAudioInfo(  
    unsigned int uiSampleRate,  
    unsigned int uinChannelCounts,  
)
```

Parameters

uiSampleRate
 source sampling rate
uinChannelCounts
 source channel numbers

Return

If there is no error, it returns 0.

Remark

2.5 TCC_HLS_SendVideoInfo

Inform source video information. This information is needed by trnas-coder.

Syntax

```
int TCC_HLS_SendVideoInfo(  
    unsigned int width,  
    unsigned int height,  
    unsigned int fps  
)
```

Parameters

width
 source video width
height
 source video height
fps
 source video framerate

Return

If there is no error, it returns 0.

Remarks

2.6 TCC_HLS_WriteVideoFrame

Inform source video information. This information is needed by trnas-coder.

Syntax

```
int TCC_HLS_WriteVideoFrame(  
    unsigned char *pucData,  
    unsigned int uiSize,  
    unsigned int uiPTS,  
    unsigned int uiPicType  
)
```

Parameters

pucData

Physical address of video output buffer. Telechips video decoder needs several video output buffer. This is one of them.

uiSize

The pucData size. Typically uiSize is 3*4byte, because pucData is YUV address.

uiPTS

PTS(Presentation Time Stamp) of current video frame

uiPicType

Picture type of current frame. Trans-coder should know this information, because trans-coder decides proper key frame interval. Telechips video decoder also gives this information.

Return

If there is no error, it returns 0.

Remarks

2.7 TCC_HLS_WriteAudioFrame

Inform source video information. This information is needed by trnas-coder.

Syntax

```
int TCC_HLS_WriteAudioFrame(  
    unsigned char *pcmdata,  
    int len,  
    unsigned int ts  
)
```

Parameters

pcmdata

Audio output pcm data pointer

len

pcmdata size

ts

PTS(Presentation Time Stamp) of current audio frame

Return

If there is no error, it returns 0.

Remarks

2.8 TCC_HLS_SetOutputCallback

Set user output callback function. TCC Trans-Coder call this callback function if it is registered. User can get ts stream data by this callback function.

Syntax

```
void TCC_HLS_SetOutputCallback(  
    void (*out_callback)(  
        unsigned char *pbuffer,  
        long len,  
        long pts,  
        unsigned long estype)  
)
```

Parameters

Out_callback

User output callback for passing ts output data inside TCC Trans-Coder

pbuffer

ts output data pointer

pts

PTS(Presentation Time Stamp) of current ts output. Unit is msec.

estype

Indication for Audio/Video. Audio is 2. Video is 1.

Return

Remarks

FRAMEWORK

3 How to use TCC Trans-Coder in android media framework

User should hook video/audio output to use TCC Trans-Coder with android video player. We made member functions on stagefright player. If user set below member functions as user callback, the stagefright player call user callback if audio or video output data is available. This modification in in framework/base@android PDK. If you define below member functions as your won callback, The Telechips AwesomePlayer doesn't use renderer(Audio/Video). That mean while transcoding, the audio/video output is disabled. The output is sent to TCC Trans-Coder. By this way you can control your output scenario.

```
@ include/media/stagefright/AwesomePlayer.h
```

```
typedef int (*SendAudioFrameInfoFunc)
(unsigned int SampleRate,
 unsigned int nChannelCounts);
SendAudioFrameInfoFunc funcSendAudioFrameInfoForTrscore;
```

This member function is for setting audio information(Sampling rate, Channel numbers).

Be sure that funcSendAudioFrameInfoForTrscore@include/media/stagefright/AudioPlayer.h is not used currently.

```
@ include/media/stagefright/AudioPlayer.h
```

```
typedef int (*SendAudioFrameFunc)(
 unsigned char *pcmdata,
 int len,
 unsigned int ts
);
SendAudioFrameFunc funcSendAudioForTrscore;
```

This member functions is for audio pcm data.

```
@ include/media/stagefright/AwesomePlayer.h
```

```
typedef int (*SendVideoFrameInfoFunc)(
 unsigned int width,
 unsigned int height,
 unsigned int fps
);
SendVideoFrameInfoFunc funcSendFrameInfoForTrscore;
```

This member functions is for video information.

```
@ include/media/stagefright/AwesomePlayer.h
```

```
typedef int (*SendVideoFrameFunc)(
 unsigned char *pucData,
 unsigned int uiSize,
 unsigned int uiPTS,
 unsigned int uiPicType
);
SendVideoFrameFunc funcSendFrameForTrscore;
```

This member function is for video out data(YUV physical address).

FRAMEWORK

3.1 Examples with TCC Trans-Coder library

```

int myAudioSetting(unsigned int SampleRate,unsigned int nChannelCounts)
{
    TCC_HLS_SendAudioInfo(SampleRate, nChannelCounts);
    return 0;
}
int mySendAudio( unsigned char *pcmdata, int len, unsigned int ts)
{
    TCC_HLS_WriteAudioFrame(pcmdata, len, ts);
    return 0;
}
int myVideoSetting( unsigned int width, unsigned int height, unsigned int fps )
{
    TCC_HLS_SendVideoInfo(width, height,fps);
    return 0;
}
int mySendVideo(unsigned char *pucData,unsigned int uiSize,unsigned int uiPTS,unsigned int uiPicType)
{
    TCC_HLS_WriteVideoFrame(pucData, uiSize, uiPTS, uiPicType);
    return 0;
}
void myTSStremOutput(unsigned char *pbuffer, long len, long pts, unsigned long estype)
{
    //CAUTION!!! This function should not be blocked !!

    if(estype == 1)
        //Send vidoe ts data to Streaming Server(Such as Http Live Streaming)
    else
        //Send audio ts data to Streaming Server(Such as Http Live Streaming)
}

void myHSLInit(void)
{
    TCC_HLS_Trscoder_Init(..);
    TCC_HLS_SetOutputCallback(myTSStremOutput);
    funcSendAudioFrameInfoForTrscore = myAudioSetting;
    funcSendAudioForTrscore = mySendAudio;
    funcSendFrameInfoForTrscore = myVideoSetting;
    funcSendFrameForTrscore = mySendVideo;
}

```

We also provide source sample codes which is modified from original AwesomePlayer/AudioPlayer@framework/base. Please refer USE_TELECHIPS_TRANS_CODER define at AudioPlayer.cpp and AwesomePlayer.cpp in sample_and_patch materials from Telechips. In sample codes, We save TCC Trans-Coder output as ts stream file while playing.

4 Trouble Shutting

4.1 Insufficient VPU memory error

Some Full HD contents needs more VPU memory than other lower bitrate contents. Therefore Some High bitrate contents meets Insufficient VPU memory error which user can see in logcat log. In this case you should increase VPU memory. Please refer below modification.

@ bootable/bootloader/lk/include/arch/tcc_used_mem_tcc8800st.h

```
.....  
#if (TCC_MEM_SIZE <= 256)  
/* We restrict video resolution to 720p for 256MB or DDR3 */  
#define VIDEO_MEM_TOTAL_SIZE      (36*SZ_1M)  
#else  
#define VIDEO_MEM_TOTAL_SIZE      (58*SZ_1M)  
#endif  
  
#define HDMI_DISPLAY_MAX_WIDTH     1920  
#define HDMI_DISPLAY_MAX_HEIGHT    1080
```

The default is 58Mbyte at over 256Mbyte memory. Please increase this value to proper size such as 6xMbyte.