```
#include "mvc2api.h"
#include "CMXFParserModule.h"
#include "mvc2api_securitymanager.h"

#pragma warning(disable : 4996)

#include <iostream>
#include <cstdio>
#include <cstdlib>
#include <cstring>
using namespace std;
using namespace mvc2;

#define VIDEO_MXF "/home/zhang/sdk_32bit/movie/dieying3_xyz_ct/dieying3_xyz_video_ct.mxf"
#define AUDIO_MXF "/home/zhang/sdk_32bit/movie/dieying3_xyz_ct/dieying3_xyz_audio_ct.mxf"
#define SUBTITLE_FILE "/home/zhang/sdk_32bit/movie/dieying3_xyz_ct/a00ccb11-62d4-46b1-bd05-c14b86bea9d7/dieying3_chinese_subtitle.xml"
#define FONT_FILE "/home/zhang/sdk_32bit/movie/dieying3_xyz_ct/a00ccb11-62d4-46b1-bd05-c14b86bea9d7/simhei-C.ttf"

#define VIDEO_BUFFER_LENGTH 2 * 1024 * 1024
#define AUDIO_BUFFER_LENGTH 36000 * 10
/// XLQ:
#define MM_LINUX
#define MVC2API_NETWORK_ONLY
#define IMB_IP_ADDRESS "10.7.75.1"
/// XLQ

const char* bin2hex(unsigned char* bin_buf, unsigned int bin_len, char* str_buf, unsigned int str_len)
{
    if ( bin_buf == 0
        || str_buf == 0
        || ((bin_len * 2) + 1) > str_len )
        return 0;

    //#ifdef CONFIG_RANDOM_UUID
    //  const char* use_random_uuid  = getenv("KM_USE_RANDOM_UUID");
    //  if ( use_random_uuid != 0 && use_random_uuid[0] != 0 && use_random_uuid[0] != '0' )
    //    return bin2hex_rand(bin_buf, bin_len, str_buf, str_len);
    //#endif

    char* p = str_buf;

    for ( unsigned int i = 0; i < bin_len; i++ )
    {
        *p = (bin_buf[i] >> 4) & 0x0f;
        *p += *p < 10 ? 0x30 : 0x61 - 10;
        p++;

        *p = bin_buf[i] & 0x0f;
        *p += *p < 10 ? 0x30 : 0x61 - 10;
        p++;
    }

    *p = '\0';
    return str_buf;
```

```
}

TMmRc TransferAudio_CT(MvcDecoder& dec, const unsigned char *audioDataBuffer, unsigned long &audioDataLength, unsigned int &m_uPlaintextOffset,
    unsigned int &m_uSourceLength, bool &m_bHmacFlag,char *m_cKeyID)
{
    DataBuffer dataBuffer;
    TMmRc ret = MMRC_Ok;

    ret = dec.getDataBuffer(dataBuffer, AUDIO_BUFFER_LENGTH);
    if (MM_IS_ERROR(ret))
    {
        printf("could not get audio databuffer -> abort\n");
        return(ret);
    }

    uint32_t readbytes = 0;

    /// XLQ:从IV段、CV段这32个字节的后面PlainText Offset段的开始取值，到E(V)的最后一个字节！！！
    if(true == m_bHmacFlag)
    {
        memcpy( dataBuffer.getBufferAddress(),
            (audioDataBuffer + 32),
            (audioDataLength - 32 - 56) );

        readbytes = audioDataLength - 32 - 56;

        dataBuffer.setMicValue( (audioDataBuffer + (audioDataLength - 56) ), 56);
    }
    else
    {
        memcpy( dataBuffer.getBufferAddress(),
            (audioDataBuffer + 32),
            (audioDataLength - 32) );

        readbytes = audioDataLength - 32;
    }

    dataBuffer.setDecryptionSize(m_uPlaintextOffset, m_uSourceLength);

    char keyId[64] = "";
    bin2hex((unsigned char *)m_cKeyID, 16, keyId, 64);
    //printf("%s", keyId);

    mvc2::UuidValue keyid(keyId);
    //mvc2::UuidValue keyid = "25091308b27ed5bf19daec4a1b32a6be";

    dataBuffer.setKeyId(keyid, audioDataBuffer, audioDataBuffer + 16);

    //dataBuffer.setKeyIndex(1, audioDataBuffer, audioDataBuffer + 16);

    uint32_t padding = (16 - (readbytes & 0x0f)) & 0x0f;
    uint8_t *buf = dataBuffer.getBufferAddress();
    for (uint32_t i = 0; i < padding; i++)
    {
        buf[readbytes + i] = 0;
```

```
    }

    dataBuffer.send(readbytes + padding);

    return(ret);
}

TMmRc TransferVideo_CT(MvcDecoder& dec, const unsigned char *videoDataBuffer, unsigned long &videoDataLength, unsigned int &m_uPlaintextOffset,
    unsigned int &m_uSourceLength, bool &m_bHmacFlag, char *m_cKeyID)
{
    DataBuffer dataBuffer;
    TMmRc ret = MMRC_Ok;

    ret = dec.getDataBuffer(dataBuffer, VIDEO_BUFFER_LENGTH);
    if (MM_IS_ERROR(ret))
    {
        printf("could not get databuffer -> abort\n");
        return(ret);
    }

    uint32_t readbytes = 0;

    /// XLQ:从IV段、CV段这32个字节的后面PlainText Offset段的开始取值，到E(V)的最后一个字节！！！
    if(true == m_bHmacFlag)
    {
        memcpy( dataBuffer.getBufferAddress(),
            (videoDataBuffer + 32),
            (videoDataLength - 32 - 56) );

        readbytes = videoDataLength - 32 - 56;

        dataBuffer.setMicValue( (videoDataBuffer + (videoDataLength - 56) ), 56);
    }
    else
    {
        memcpy( dataBuffer.getBufferAddress(),
            (videoDataBuffer + 32),
            (videoDataLength - 32) );

        readbytes = videoDataLength - 32;
    }

    dataBuffer.setDecryptionSize(m_uPlaintextOffset, m_uSourceLength);

    char keyId[64] = "";
    bin2hex((unsigned char *)m_cKeyID, 16, keyId, 64);
    //printf("%s", keyId);

    mvc2::UuidValue keyid(keyId);
    //mvc2::UuidValue keyid = "007203b983345b84bcb0c928e8bab01b";

    dataBuffer.setKeyId(keyid, videoDataBuffer, videoDataBuffer + 16);

    //dataBuffer.setKeyIndex(0, videoDataBuffer, videoDataBuffer + 16);
```

```
    uint32_t padding = (16 - (readbytes & 0x0f)) & 0x0f;
    uint8_t *buf = dataBuffer.getBufferAddress();
    for (uint32_t i = 0; i < padding; i++)
    {
        buf[readbytes + i] = 0;
    }
    //  datbuf.setUserData(framecount);
    //printf("sending pic %d\n",framecount);
    //printf("readbytes %d\n", readbytes);
    //printf("padding %d\n", padding);
    dataBuffer.send(readbytes + padding);
    //fclose(infile);

    //datbuf.wait(100);

    return(ret);
}


TMmRc TransferVideo_PT(MvcDecoder& dec, char *videoDataBuffer, unsigned long &videoDataLength)
{
    DataBuffer dataBuffer;
    TMmRc ret = MMRC_Ok;

    ret = dec.getDataBuffer(dataBuffer, VIDEO_BUFFER_LENGTH);
    if (MM_IS_ERROR(ret))
    {
        printf("could not get databuffer -> abort\n");
        return(ret);
    }

    memcpy(dataBuffer.getBufferAddress(), videoDataBuffer, videoDataLength);
    uint32_t readbytes = videoDataLength;

    // copy one frame here (as an example we fill the buffer with 1's)
    //uint32_t readbytes = static_cast<uint32_t>(fread(datbuf.getBufferAddress(),1,(size_t)(datbuf.getFreeSize()),infile));
    //int readbytes = fread(datbuf.getBufferAddress(), 1, 1301870, infile);
    //printf("datbuf.getFreeSize() %d\n", datbuf.getFreeSize());
    //printf("readbytes1 %d\n", readbytes);
    uint32_t padding = (16 - (readbytes & 0x0f)) & 0x0f;
    uint8_t *buf = dataBuffer.getBufferAddress();
    for (uint32_t i = 0; i < padding; i++)
    {
        buf[readbytes + i] = 0;
    }
    //  datbuf.setUserData(framecount);
    //printf("sending pic %d\n",framecount);
    //printf("readbytes %d\n", readbytes);
    //printf("padding %d\n", padding);
    dataBuffer.send(readbytes + padding);
    //fclose(infile);

    //datbuf.wait(100);

    return(ret);
```

```
}

TMmRc TransferVideo_PT(MvcDecoder& dec, MvcDecoder& dec_right, char *videoDataBuffer, unsigned long &videoDataLength, char *videoDataBuffer_right,
    unsigned long &videoDataLength_right)
{
    DataBuffer dataBuffer;
    TMmRc ret = MMRC_Ok;

    ret = dec.getDataBuffer(dataBuffer, VIDEO_BUFFER_LENGTH);
    if (MM_IS_ERROR(ret))
    {
        printf("could not get databuffer -> abort\n");
        return(ret);
    }

    memcpy(dataBuffer.getBufferAddress(), videoDataBuffer, videoDataLength);
    uint32_t readbytes = videoDataLength;

    // copy one frame here (as an example we fill the buffer with 1's)
    //uint32_t readbytes = static_cast<uint32_t>(fread(datbuf.getBufferAddress(),1,(size_t)(datbuf.getFreeSize()),infile));
    //int readbytes = fread(datbuf.getBufferAddress(), 1, 1301870, infile);
    //printf("datbuf.getFreeSize() %d\n", datbuf.getFreeSize());
    //printf("readbytes1 %d\n", readbytes);
    uint32_t padding = (16 - (readbytes & 0x0f)) & 0x0f;
    uint8_t *buf = dataBuffer.getBufferAddress();
    for (uint32_t i = 0; i < padding; i++)
    {
        buf[readbytes + i] = 0;
    }
    //  datbuf.setUserData(framecount);
    //printf("sending pic %d\n",framecount);
    //printf("readbytes %d\n", readbytes);
    //printf("padding %d\n", padding);
    dataBuffer.send(readbytes + padding);
    //fclose(infile);

    //datbuf.wait(100);


    /// XLQ:右眼
    DataBuffer dataBuffer_right;
    TMmRc ret_right = MMRC_Ok;

    ret_right = dec_right.getDataBuffer(dataBuffer_right, 2 * 1024 * 1024);
    if (MM_IS_ERROR(ret_right))
    {
        printf("could not get databuffer -> abort\n");
        return(ret_right);
    }

    memcpy(dataBuffer_right.getBufferAddress(), videoDataBuffer_right, videoDataLength_right);
    uint32_t readbytes_right = videoDataLength_right;

    // copy one frame here (as an example we fill the buffer with 1's)
    //uint32_t readbytes = static_cast<uint32_t>(fread(datbuf.getBufferAddress(),1,(size_t)(datbuf.getFreeSize()),infile));
```

```
    //int readbytes = fread(datbuf.getBufferAddress(), 1, 1301870, infile);
    //printf("datbuf.getFreeSize() %d\n", datbuf.getFreeSize());
    //printf("readbytes1 %d\n", readbytes);
    uint32_t padding_right = (16 - (readbytes_right & 0x0f)) & 0x0f;
    uint8_t *buf_right = dataBuffer_right.getBufferAddress();
    for (uint32_t i_right = 0; i_right < padding_right; i_right++)
    {
        buf_right[readbytes_right + i_right] = 0;
    }
    //  datbuf.setUserData(framecount);
    //printf("sending pic %d\n",framecount);
    //printf("readbytes %d\n", readbytes);
    //printf("padding %d\n", padding);
    dataBuffer_right.send(readbytes_right + padding_right);
    //fclose(infile);

    //datbuf.wait(100);

    return(ret);
}

TMmRc TransferAudio_PT(MvcDecoder& dec, char *audioDataBuffer, unsigned long &audioDataLength)
{
    DataBuffer dataBuffer;
    TMmRc ret = MMRC_Ok;

    ret = dec.getDataBuffer(dataBuffer, AUDIO_BUFFER_LENGTH);
    if (MM_IS_ERROR(ret))
    {
        printf("could not get audio databuffer -> abort\n");
        return(ret);
    }

    memcpy(dataBuffer.getBufferAddress(), audioDataBuffer, audioDataLength);
    uint32_t readbytes = audioDataLength;

    uint32_t padding = ( 16 - (readbytes & 0x0f) ) & 0x0f;
    uint8_t *buf = dataBuffer.getBufferAddress();
    for (uint32_t i = 0; i < padding; i++)
    {
        buf[readbytes + i] = 0;
    }
    //  datbuf.setUserData(framecount);
    //printf("sending pic %d\n",framecount);
    //printf("readbytes %d\n", readbytes);
    //printf("padding %d\n", padding);
    dataBuffer.send(readbytes + padding);

    //datbuf.send(audiosDataLength);
    return(ret);
}


int main(int argc, char **argv)
{
```

```
    printf("This is a example of Playing Plaintext movie!这是一个播放Jpeg2K明文2D影片的例子! \n");

    TMmRc ret;

    MvcDevice mvcdevice;

#ifndef MVC2API_NETWORK_ONLY
    if (! ( mvcdevice = MvcDeviceIterator().getIndex(0) ) )
#else
    if (! ( mvcdevice = MvcNetDeviceIterator(IMB_IP_ADDRESS).getIndex(0) ) )
#endif

    {
        printf("MVC card not found\n");
        exit(0);
    }
    else
    {
        printf("MVC card is found!\n");
    }

    //ret = mvcdevice.resetCard();
    //if (MM_IS_ERROR(ret))
    //{
    //    printf("failed to resetCard: %d\n", ret);
    //    exit(-1);
    //}
    //else
    //{
    //    printf("succeed to resetCard: %d\n", ret);
    //    exit(0);
    //}
        ///video of MXF Parser
    char *videoDataBuffer = new char[VIDEO_BUFFER_LENGTH];
    unsigned long videoDataLength = 0;
    char *audioDataBuffer = new char[AUDIO_BUFFER_LENGTH];
    unsigned long audioDataLength = 0;

    CMXFParserModule cmxfParserModule;
    unsigned long videoFrameSum = 0;
    unsigned long audioFrameSum = 0;
#ifdef WIN32
    cmxfParserModule.InitVideoParser("d:\\video2.mxf", videoFrameSum);
    cmxfParserModule.InitAudioParser("d:\\audio2.mxf", audioFrameSum);
#else
    //cmxfParserModule.InitVideoParser("/home/zhang/sdk_32bit/movie/dieying3_xyz_sub/dieying3_xyz_video.mxf", videoFrameSum);
   // cmxfParserModule.InitAudioParser("/home/zhang/sdk_32bit/movie/dieying3_xyz_sub/dieying3_xyz_audio.mxf", audioFrameSum);
        cmxfParserModule.InitVideoParser(VIDEO_MXF, videoFrameSum);
    cmxfParserModule.InitAudioParser(AUDIO_MXF, audioFrameSum);

#endif
    printf("video sum frame:%lu\n", videoFrameSum);
    printf("audio sum frame:%lu\n", audioFrameSum);

    int m_iVideoType = -1;
```

```
double m_dAspectRatio = 0;
unsigned long m_uWidthSize = 0;
unsigned long m_uHeightSize = 0;
unsigned long m_uFrameRate = 0;
bool m_bHmacFlag1 = false;
bool m_bCryptoFlag1 = false;
char m_cKeyID1[16] = "";
cmxfParserModule.GetVideoInfo(m_iVideoType, m_dAspectRatio, m_uWidthSize, m_uHeightSize, m_uFrameRate, m_bHmacFlag1, m_bCryptoFlag1, m_cKeyID1);

    printf("VideoType(1为Mpeg2, 2为2D Jpeg2k):[%d]\n",m_iVideoType); ///video类型
    printf("AspectRatio:[%f]\n",m_dAspectRatio);                     ///纵横比
    printf("WidthSize:[%lu]\n",m_uWidthSize);                         ///宽度
    printf("HeightSize:[%lu]\n",m_uHeightSize);                      ///高度
    printf("FrameRate:[%lu]\n",m_uFrameRate);                                    ///帧率
    printf("HmacFlag1:[%d]\n",m_bHmacFlag1);                                     ///是否有mic值
    printf("CryptoFlag1:[%d]\n",m_bCryptoFlag1);                               /// 是否加密
    printf("KeyID1:[%s]\n",m_cKeyID1);                                                ///keyID


if(true == m_bCryptoFlag1)
{
            printf("播放的是密文\n");
    return -1;
}

unsigned long m_uSamplingRate = 0;
unsigned long m_uChannelCount = 0;
unsigned long m_uBitsPerSample = 0;
bool m_bHmacFlag2 = false;
bool m_bCryptoFlag2 = false;
char m_cKeyID2[16] = "";
cmxfParserModule.GetAudioInfo(m_uSamplingRate, m_uChannelCount, m_uBitsPerSample, m_bHmacFlag2, m_bCryptoFlag2,  m_cKeyID2);

    printf("SamplingRate:[%lu]\n",m_uSamplingRate);  ///采样率（频率）
    printf("ChannelCount:[%lu]\n",m_uChannelCount);  ///通道数
    printf("BitsPerSample:[%lu]\n",m_uBitsPerSample); ///采样位数
    printf("HmacFlag2:[%d]\n",m_bHmacFlag2);         ///是否有mic值
    printf("CryptoFlag2:[%d]\n",m_bCryptoFlag2);     ///是否加密
    printf("KeyID2:[%s]\n",m_cKeyID2);               ///keyID

if(true == m_bCryptoFlag2)
{
    printf("播放的是密文\n");
    return -1;
}

/// XLQ:视频部分为Jpeg2k格式
    Jpeg2kDecoder j2kdec(&ret, mvcdevice);
    if (MM_IS_ERROR(ret))
    {
            printf("failed to create Jpeg2k video decoder: %d\n",ret);
            exit(0);
    }

    j2kdec.setFrameRate(m_uFrameRate);                    // set frame rate 24/48, so we don't need to set timestamps anymore
```

```
// create the video output
    ////***********VideoOutput::VideoProperty_Dual_HDTV怎样设置？？？目前2k按照以下方式设置//////
VideoOutput videoout(&ret, mvcdevice, VideoOutput::VideoProperty_Dual_HDTV);
if (MM_IS_ERROR(ret))
{
    printf("failed to create video output: %d\n",ret);
    exit(0);
}
// setting a video mode (optional)
///***VideoMode::Mode_2048_1080_2400_p根据宽和高设置？？？目前2k按照以下方式设置***//
if (MM_IS_ERROR(ret = videoout.setVideoMode(VideoMode::Mode_2048_1080_2400_p)))
    //if (MM_IS_ERROR(ret = videoout.setVideoMode(VideoMode::Mode_2048_1080_4800_p)))
{
    printf("failed to set video mode: %d\n",ret);
}

// connect decoder with video output

    if (MM_IS_ERROR(ret = j2kdec.connectOutput(videoout)))
    {
            printf("failed to connect decoder with video output: %d\n",ret);
            exit(0);
    }

/// XLQ:字幕部分
SubtitleDecoder subtitleDec(&ret, mvcdevice);
if (MM_IS_ERROR(ret))
{
    printf("failed to create subtitle decoder: %d\n", ret);
    exit(0);
}

// connect decoder with video output
if (MM_IS_ERROR(ret = subtitleDec.connectOutput(videoout)))
{
    printf("failed to connect decoder with video output: %d\n",ret);
    exit(0);
}

/// XLQ:音频部分
// PCMDecoder pcmDec(&ret, mvcdevice, 24, 6);
    ///bitsPerSample     bits per sample to process (比特深度possible values: 16 and 24)
    PCMDecoder pcmDec(&ret, mvcdevice, m_uBitsPerSample, m_uChannelCount);
if (MM_IS_ERROR(ret))
{
    printf("failed to create pcm audio decoder: %d\n", ret);
    exit(0);
}

AudioOutput audioout(&ret, mvcdevice, m_uChannelCount);
if (MM_IS_ERROR(ret))
{
    printf("failed to create audio output: %d\n",ret);
    exit(0);
```

```
    }

//   if (MM_IS_ERROR(ret = audioout.setOutputFrequency(48000)))
       ///sample frequency in Hz, values of 48000 or 96000 are allowed
       if (MM_IS_ERROR(ret = audioout.setOutputFrequency(m_uSamplingRate)))
    {
        printf("failed to set audio output frequency: %d\n",ret);
    }

    // connect decoder with video output
    if (MM_IS_ERROR(ret = pcmDec.connectOutput(audioout)))
    {
        printf("failed to connect decoder with audio output: %d\n",ret);
        exit(0);
    }

    /// XLQ:播放控制部分
    PlaybackControl playctrl(&ret, mvcdevice);
    if (MM_IS_ERROR(ret))
    {
        printf("failed to create playback control: %d\n",ret);
        exit(0);
    }
    // connect decoder with playback
    ret = playctrl.connect(j2kdec);
    if (MM_IS_ERROR(ret))
    {
        printf("failed to connect playback control with video decoder: %d\n",ret);
        exit(0);
    }

    // connect decoder with playback
    ret = playctrl.connect(subtitleDec);
    if (MM_IS_ERROR(ret))
    {
        printf("failed to connect playback control with subtilte decoder: %d\n",ret);
        exit(0);
    }

    /// XLQ:
    ret = playctrl.connect(pcmDec);
    if (MM_IS_ERROR(ret))
    {
        printf("failed to connect playback control with audio decoder: %d\n",ret);
        exit(0);
    }

    /// XLQ:将字幕数据作为缓冲提前载入
    //uint8_t *data = new uint8_t[10 * 1024 * 1024];
    //uint32_t dataSize = 0;

#ifdef WIN32
    FILE *subtitleXmlFile = fopen("D:\\dieying3_xyz_sub\\a00ccb11-62d4-46b1-bd05-c14b86bea9d7\\dieying3_chinese_subtitle.xml", "rb");
#else
        //FILE *subtitleXmlFile = fopen("/home/zhang/sdk_32bit/movie/dieying3_xyz_sub/a00ccb11-62d4-46b1-bd05-c14b86bea9d7/dieying3_chinese_subtitle.xml", "rb");
```

```
        FILE *subtitleXmlFile = fopen(SUBTITLE_FILE, "rb");

#endif
        if (subtitleXmlFile==NULL)
    {
            printf("open subfile failed\n");
        fputs ("File error",stderr);
        exit (1);
    }
        unsigned long subtitleLength;
        unsigned char *subtitleBuffer=NULL;
        ///计算字幕文件大小
        ret=fseek(subtitleXmlFile,0L,SEEK_END);
        printf("ret:%d\n",ret);
        subtitleLength=ftell(subtitleXmlFile);
        printf("subtitleLength:%lu\n",subtitleLength);
        subtitleBuffer=new unsigned char[subtitleLength];

        if(subtitleBuffer==NULL)
        {
            fclose(subtitleXmlFile);
            printf("new subtitle memory failed\n");
            return -1;
        }
        ///读入字幕文件到buffer
        fseek(subtitleXmlFile,0L,SEEK_SET);
        fread(subtitleBuffer,1,subtitleLength,subtitleXmlFile);
        fclose(subtitleXmlFile);
        printf("Read subtitle File Success,Length=%lu\n",subtitleLength);

        // fread(data, 1, 10244, subtitleXmlFile);
   // dataSize = 10244;
   // fclose(subtitleXmlFile);

    uint32_t resourceId = 0;
   // ret = subtitleDec.sendSubtitleFile(data, dataSize, &resourceId);
         ret = subtitleDec.sendSubtitleFile(subtitleBuffer, subtitleLength, &resourceId);
    if (MM_IS_ERROR(ret))
    {
        printf("failed to sendSubtitleFile: %d\n", ret);
        exit(0);
    }
    else
    {
        printf("resourceId=%d\n", resourceId);
    }
        ////释放subtitleXmlFile内存
        if(NULL !=subtitleBuffer)
        {
            delete subtitleBuffer;
            subtitleBuffer=NULL;
        }
    //// XLQ:载入字体文件
#if 1
#ifdef WIN32
```

```
    FILE *fontFile = fopen("D:\\dieying3_xyz_sub\\a00ccb11-62d4-46b1-bd05-c14b86bea9d7\\simhei-C.ttf", "rb");
#else
   // FILE *fontFile = fopen("/home/zhang/sdk_32bit/movie/dieying3_xyz_sub/a00ccb11-62d4-46b1-bd05-c14b86bea9d7/simhei-C.ttf", "rb");
        FILE *fontFile = fopen(FONT_FILE, "rb");

#endif
   if (fontFile==NULL)
    {
                printf("open fontFile failed\n");
        fputs ("File error",stderr);
        exit (1);
    }
        //fread(data, 1, 54904, fontFile);
 //   dataSize = 54904;
 //   fclose(fontFile);

    //ret = subtitleDec.sendOverlayElement("simhei-C.ttf", data, dataSize, resourceId);

        unsigned long fontLength;
        unsigned char *fontBuffer=NULL;
        ///计算字体文件大小
        ret=fseek(fontFile,0L,SEEK_END);
        printf("ret:%d\n",ret);
        fontLength=ftell(fontFile);
        printf("fontLength:%lu\n",fontLength);
        fontBuffer=new unsigned char[fontLength];

        if(fontBuffer==NULL)
        {
                fclose(fontFile);
                printf("new fontFile memory failed\n");
                return -1;
        }
        ///读入字体文件到buffer
        //printf("读入字体文件到buffer\n");
        fseek(fontFile,0L,SEEK_SET);
        //fread(fontBuffer,1,fontLength,fontFile);
        fread(fontBuffer,fontLength,1,fontFile);
        fclose(fontFile);
        printf("Read subtitle File Success,Length=%lu\n",fontLength);

        ret = subtitleDec.sendOverlayElement("simhei-C.ttf", fontBuffer, fontLength, resourceId);

        if(NULL !=fontBuffer)
        {
                delete fontBuffer;
                fontBuffer=NULL;
        }

    if (MM_IS_ERROR(ret))
    {
        printf("failed to sendOverlayElement: %d\n", ret);
        exit(0);
    }
```

.ttf and .png are all overlay elements,they will be downloaded by sendOverlayEl

```
#else

    OverlayElementDataBuffer element;
    subtitleDec.getDataBuffer(element);
    if (element)
    {
        element.setElementName("simhei-C.ttf", resourceId);
        FILE *fontfile = fopen("D:\\dieying3_xyz_sub\\a00ccb11-62d4-46b1-bd05-c14b86bea9d7\\simhei-C.ttf","rb");
        if (fontfile)
        {
            ret = element.send(fread(element.getBufferAddress(),1,element.getFreeSize(),fontfile));
            if (MM_IS_ERROR(ret))
            {
                printf("failed to element.send: %d\n", ret);
                exit(0);
            }
            fclose(fontfile);
        }
        else
        {
            printf("could not open font: %s\n","D:\\dieying3_xyz_sub\\a00ccb11-62d4-46b1-bd05-c14b86bea9d7\\simhei-C.ttf");
        }
    }

#endif

    ret = subtitleDec.enableSubtitles(0, SubtitleDecoder::Render_Soft_Shadows);
    //ret = subtitleDec.enableSubtitles(1000000, SubtitleDecoder::Render_Soft_Shadows);
    //ret = subtitleDec.enableSubtitles(-1000000, SubtitleDecoder::Render_Soft_Shadows);
    //ret = subtitleDec.enableSubtitles(-3800000, SubtitleDecoder::Render_Soft_Shadows);
    if (MM_IS_ERROR(ret))
    {
        printf("failed to enableSubtitles: %d\n", ret);
        exit(0);
    }

        printf("20 picture preload before run!\n");
    for (int i = 0; i < 20; i++)
    {
        unsigned int m_uPlaintextOffset;
        unsigned int m_uSourceLength;

        cmxfParserModule.GetVideoFrameData(i, videoDataBuffer, videoDataLength, m_uPlaintextOffset, m_uSourceLength);

                ret = TransferVideo_PT(j2kdec, videoDataBuffer, videoDataLength);

        if (ret)
        {
            exit(0);
        }

        cmxfParserModule.GetAudioFrameData(i, audioDataBuffer, audioDataLength, m_uPlaintextOffset, m_uSourceLength);

        ret = TransferAudio_PT(pcmDec, audioDataBuffer, audioDataLength);
        if (ret)
```

Another way to download subtitle data.

```
        {
            exit(0);
        }

    }
        printf("Star play!\n");
    if (ret == MMRC_Ok)
    {
        playctrl.run();

        for (int i = 0; i < videoFrameSum; i++)
        {
            unsigned int m_uPlaintextOffset;
            unsigned int m_uSourceLength;

            cmxfParserModule.GetVideoFrameData(i, videoDataBuffer, videoDataLength, m_uPlaintextOffset, m_uSourceLength);

                        ret = TransferVideo_PT(j2kdec, videoDataBuffer, videoDataLength);
                        if (ret)
                        {
                                printf("Play video failed!\n");
                                exit(0);
                        }

            cmxfParserModule.GetAudioFrameData(i, audioDataBuffer, audioDataLength, m_uPlaintextOffset, m_uSourceLength);
            ret = TransferAudio_PT(pcmDec, audioDataBuffer, audioDataLength);
            if (ret)
            {
                                printf("Play audio failed!\n");
                exit(0);
            }
        }

                j2kdec.setEndOfStream();
        pcmDec.setEndOfStream();

    }
        if (ret != MMRC_Ok)
    {
        //printf("picture transfer failed (%s)\n",filename);
        printf("picture transfer failed\n");
    }
        playctrl.waitForEndOfStream();

    printf("End of stream reached\n");

         if(NULL != audioDataBuffer)
    {
        delete[] audioDataBuffer;
        audioDataBuffer = NULL;
        audioDataLength = 0;
    }

    if(NULL != videoDataBuffer)
    {
```

```
        delete[] videoDataBuffer;
        videoDataBuffer = NULL;
        videoDataLength = 0;
    }
        return 0;
}
```