

Mpeg4 Player (exo1Mpeg4)

NDK 5.7/MPTK 2.4

Application Note, 12 December 2007

Overview

This example application demonstrates the use of the TriMedia MPEG-4 components in order to build an MPEG-4 application. Exo1Mpeg4 is the key demonstration program. This note refers you to others as appropriate.

1 Description

The application builds the following TSSA chain:

```
tmMP4Read/tmNetworkRead --> |--> tmVdecMpeg4/tmVdech264 --> tmVrendGfxVo
                                |
                                |--> tmAdecAac4/tmAdecCelP4 --> tmArendAo
```

Another application that uses the MPEG4 components is the Media Box, or Mbox application.

1.1 exMboxVideoPlay

This uses the mpeg4Player component to read streams with the tmMP4Read component and play them. Does not support network reader. Does support a better implementation of trick modes than exo1Mpeg4.

Exo1Mpeg4 is an example of a "single instance" usage of the MPEG4 components.

`prod\mptk\apps\exo1Mpeg4Mi` and `sd\avs\mpeg4\tst\tstMpeg4Mi` are examples of programs that use the MPEG4 components in a multi-instance mode.

- `TstMpeg4Mi` – connects one of the decoders to the audio and video renderers while the other instance decodes to file.
- `Exo1Mpeg4Mi` – uses an experimental version of the video decoder to display four decodes at once.

2 Instructions

Exo1Mpeg4 can read files from files or from an RTSP server such as Apple's Darwin server. You specify which file to decode using the `-url` command on the startup command line.

When the URL is specified as `file://`, then the MP4 reader is invoked. The MP4 reader can read MPEG4 files and QuickTime™ files. If the file name begins with `DISK:/`, then the program will look for an IDE drive attached to the TriMedia.

When the URL is specified as `rtsp://`, then the network reader is invoked to request the file from an RTSP server.

You can also specify the URL as `http://`. In this case, the MP4 reader is invoked and the HTTP protocol is used to retrieve the file. Unfortunately, this path has some bugs. It repeatedly seeks to the start of the file and so requires much more bandwidth than is reasonable. If this mode is required for your application, contact TCSHelp.

Some examples of working command lines are:

- To play a stream from the host PC's file system:
 - url file://localhost/c:/myfolder/myfile.mp4 OR
 - url file:///c:/myfolder/myfile.mp4

Note the standard URL shortcut to replace localhost. If you use such a URL in a JTAG hosted system, the code will attempt to read the PC's file system over JTAG and it will be VERY slow.

- To play a stream from the IDE hard disk in a stand alone system:
 - url file://localhost/DISK:/myfolder/myfile.mp4 OR
 - url file:///DISK:/myfolder/myfile.mp4
- To play a stream from rtsp server:
 - url rtsp://rtspServerAddress/myfile.mp4
- To play a stream from http server (with the limitations noted):
 - url http://httpServerAddress:PortNumber/myfile.mp4
- In multi-instance demonstrations, the command syntax is for playing a multi-instance stream:
 - url file:///<Mp4\Avi\Mov File PATH> -urlMult file:///<Mp4\Avi\Mov File Path> -fwriteA <Audio Path> -fwriteV <Video Path>

It is often useful to specify the video out mode on the command line. If not specified, the application attempts to choose the appropriate video out mode which may be PAL or NTSC depending on the stream. If your monitor accepts only one of these, specify it.

-videoOutMode ntsc

3 Stream Requirements

For more information on the capabilities of the components used here, please see the specific IP brief.

exoMpeg4 can play streams that fall into these categories:

Container formats

- .mp4 file format
- QuickTime™ file format
- Hinted MP4 files from an RTSP server

Only one video track and one audio track are supported.

Video formats

- MPEG-4 SP or ASP up to L5. Tested maximum bit rate is 4 Mbps.
- H.264 video, baseline or main profile at bit rates up to 3Mbps.

Audio formats

- MPEG 4 AAC LC
- MPEG4 CELP

This is not an exhaustive list of the formats supported by Nexperia. It is a list of the formats supported by the exolMpeg4 application.

4 User Options

Note: When the application is running, you can only give commands for trick modes, when you have enabled this with the command line argument `-trickmode`.

The application accepts the following command line arguments:

-url <url>

With this command, the location of the resource is set. You must specify the location of a resource in order to use the sample application. The syntax is compliant with RFC 1738.

[-clientserversetup]

Applicable only in the case of a playback via RTSP or HTTP.

Providing this command will give the user the possibility to specify the following client/server parameters:

- For playback via RTSP:
 - connection timeout in milliseconds
 - pre-load time in milliseconds
 - reading timeout in milliseconds
- For playback via HTTP 1.1:
 - proxy address & port
 - connection timeout in milliseconds

Note: This is an important note regarding proxy servers:

In case the streaming server is behind a proxy, please be sure to provide the correct proxy IP address and port. When you provide the wrong proxy address or port, you will most likely not be notified by exolMpeg4. At this time only error returned by proxy- or http-servers are returned correctly to the client (file not found, access not authorized,...). Errors that occur because a proxy- or http server could not be found, are not returned correctly. This will be solved in a new release of the `tmHTTPIODriver`.

In case the streaming server is not behind a proxy, please don't set the proxy server or provide '0' as proxy server. This way the proxy server will not be set.

[-rtpbuffersetup]

Applicable only in the case of a playback via RTSP.

Providing this command will give the user the possibility to specify following rtp buffering parameters:

- For playback via RTSP:

- buffer size in milliseconds
- lower buffer threshold for monitoring low watermark in %
- upper buffer threshold for monitoring high watermark in %
- discontinuity timeout for monitoring timestamps by means of:
 - maximum processing path:
Specifies the delay (expressed in milliseconds) to be added to the CTS and DTS of all outgoing TSSA packet for TSSA chain delay compensation with respect to the local TSA clock.
 - discontinuity time out:
Specifies the threshold (expressed in milliseconds) at which two consecutive CTS values are considered discontinuous.

[-networkstats]

Applicable only in the case of a playback via RTSP.

Providing this comment will make sure that the network reader will progress following information:

- Audio/Video buffer load
- Audio/Video Time Interval
- Audio/Video AUs received
- Last Audio/Video CTS
- Audio/Video RTP packets lost

[-trickmode]

Enables trick modes as implemented in the MP4Reader and the network reader.

For file playback, NXP recommends a trick mode implementation modeled on the Mbox application, and not what is demonstrated here.

Usage of following trick mode commands

PAUSE:	This command is used to pause the stream which is playing from a local disk or from HTTP/RTSP server.
RESUME	This command is used to bring back the stream in normal condition after execution of PAUSE, FASTFORWARD, FASTREWIND, SINGLESTEP, I-ONLY and IP-ONLY commands.
RESTART:	This command is used to play the stream again from the beginning after executing stop command.
STOP:	This command is used to stop the stream
JUMP:	This command is used to jump to the desire location in the stream while playback is going on.
GETTIME:	This command is used to get the current playback time of the stream.
SINGLESTEP:	This command is used to display the stream frame by frame. User has to feed this command to see each and every frame.
FASTFORWARD:	This command is used to fast forward the stream according to the entered speed. Enter the value of 1->X , 2-> 2X, 3->4X, 4->8X, 5->16X, 6->32X
FASTREWIND:	This command is used to fast rewind the stream according to the entered speed. enter the value of 1->X , 2-> 2X, 3->4X, 4->8X, 5->16X, 6->32X
I-ONLY FRAME:	This command is used to render only I Frame.
IP-ONLY FRAME:	This command is used to render only IP Frame.
QUIT:	This command is used to stop the stream asynchronously and exit from the player.

[-videoOutMode <valid mode>]

Format: <width>x<height>x<framerate><scantype>

Examples:

ntsc

pal

720x480x60p

720x576x25i (PAL) (default)

[-videoOutScalingMode <asIs|fullScreen>]

- asIs: render as is, no scaling (default)
- fullScreen: full screen, not respecting aspect ratio

[-enableGracefulDegradation]

Video decoder will skip frames when they arrive too late at the decoder.

Frames cannot arrive on time when the load is high.

[-operatingMode <tmolVrendGfxVo_OperatingModeFieldBased\ tmolVrendGfxVo_OperatingModeFrameBased>]

The operating mode for layer 0. The operating mode denotes whether the operations are:

1. scaling
2. filtering, and
3. picture repeat in case of underruns, are done field or frame based.

For more details, please refer to the user manual for tmVrendGfxVo.

The default operating mode is 'tmolVrendGfxVo_OperatingModeFieldBased' for interlaced video output modes and 'tmolVrendGfxVo_OperatingModeFrameBased' for progressive video output modes.

This command line option is only relevant in case the film detection mode is 'filmDetectionModeNone'.

[-filmDetectorMode < filmDetectionModeNone \ilmDetectionModeTrustFlags>]

filmDetectorMode denotes which method is used to determine the scan type of the stream. The scan type of the stream is very important when determining the best operating mode, deinterlacing mode and eddi level. Possible values for the film detection mode are:

- filmDetectionModeNone
No intelligence or assumptions. The operating mode, deinterlacing mode and eddi level are exactly as specified in the command line options.
- filmDetectionModeTrustFlags
The flags in the stream that denote the scan type of the stream are being trusted and are used to set the operating mode, deinterlacing mode and eddi level:

	Interlaced video out	Progressive video out
Interlaced stream	tmolVrendGfxVo_OperatingModeFieldBased tmolVrendGfxVo_DeinterlaceModeNone tmolVrendGfxVo_Eddi0ff	tmolVrendGfxVo_OperatingModeFrameBased <deinterlacing mode as specified in the command line> <eddi level as specified in the command line>
Progressive stream	tmolVrendGfxVo_OperatingModeFrameBased tmolVrendGfxVo_DeinterlaceModeNone tmolVrendGfxVo_Eddi0ff	tmolVrendGfxVo_OperatingModeFrameBased tmolVrendGfxVo_DeinterlaceModeNone tmolVrendGfxVo_Eddi0ff

- filmDetectionModeDetector
The film detector, included in the video renderer, is being used to determine the scan type of the stream. This mode is not implemented for exolMpeg4, because we assume that 1) the scan type flag in MPEG-4 streams can be trusted and 2) the encoder did a decent job (combined the proper fields into a frame).

The default film detection mode is filmDetectionModeTrustFlags.

5 Expected Output

If your stream contains video, then the video should appear centered on the screen, on a blue background. Audio, when present in the stream, should be heard.

The application prints a list with the components that are in use, together with their versions, in the console. Some information about the stream is also shown, like the bit rate and the size of the video.

The expected output of exolMpeg4 application is displayed in the following table.

TriMedia MPEG-4 example application

Copyright (c) 2005 by Koninklijke Philips Electronics N.V. exolMpeg4 2.4.44937

System Info Reports:

CPU is pnx1500/TM3260, rev 1 running at 270.00MHz with memory at 166.50MHz.
Board is identified as MDS LCP1500 (ID 0x51136):
DRAM_LO: 0xec000000. DRAM_HI: 0xf0000000. C_LIMIT: 0xeffc0000.
Application built in TRACE mode.

Command line arguments:

url: rtsp://130.145.68.40/MPEG4/SelfEncoded/Hinted/LONG/moulin_rouge.Hinted.mp4

Components in use:

NetworkRead version 4.4.43616
VdecMpeg4 version 4.4.44268
VrendGfxVo version 1.11.40454
AdecAac4 version 2.12.35695
AdecCelp4 version 1.14.35695
ArendAO version 2.0.0

MPEG-4 network reader:

object type: video
average bitrate: 500.000000 kps
maximum bit rate: 9216.000000 kbps
media duration: 7668588 ms

MPEG-4 network reader:

object type: audio
format: AAC
average bitrate: 16.000000 kps
maximum bit rate: 66.782000 kbps
media duration: 7668588 ms

AAC 4 audio decoder:

sample rate: 44100.000000 Hz

MPEG-4 video decoder:

profile and level: Simple Profile/Level 3
video object layer width: 352
video object layer height: 288
video scan type: progressive
frame rate: variable

videoOut autoMode:

Determined [pal] as best match for current stream

Remark:

menu for trick modes not enabled by default
use command line argument -trickmode to activate menu

The expected output of tstMpeg4Mi application is displayed in the following table.

```
TriMedia MPEG-4 example application [compiled Oct 18 2007 22:43:10]

Copyright (c) 2005 by Koninklijke Philips Electronics N.V.  exolMpeg4 2.6.47769

-----
System Info Reports:
  CPU is pnx1700/TM5250, rev 0 running at 450.00MHz with memory at 199.80MHz.
  Board is identified as MDS LCP1700 (ID 0x171136):
  DRAM_LO: 0xd8000000.  DRAM_HI: 0xdc000000.  C_LIMIT: 0xdbfc0000.
  APERT1_LO: 0x00000000.  APERT1_HI: 0xfc000000.
  PCSW: 0x20001e02
  Application built in ASSERT mode.

Command line arguments:
  url: file:///
      D:\Test\Rocky_av8374kbps_v8230kbps_NTSC_720x480_a144kbps_48khz_AAC_stereo.mp4
  urlMult: file:///D:\Test\robots-hd-18mb-23p.mp4
  fwriteA: D:\Test\1700\test8_a.raw
  fwriteV: D:\Test\1700\test8_v.raw
no local hard drive present
  Waiting for Network to come up ...

Components in use:
  MP4Read version 4.5.43238
  VdecMpeg4 version 5.5.56914
  VdecH264 version 5.0.60198
  VrendGfxVo version 3.3.59243
  AdecAac4 version 2.13.35695
  AdecCelp4 version 1.14.35695
  ArendA0 version 2.0.0
  MP4Read_Mult version 4.5.43238
  VdecMpeg4_Mult version 5.5.56914
  AdecAac4_Mult version 2.13.35695

MPEG-4 file reader:
  object type: video
  format: MPEG4
  average bit rate: 8229.584000 kbps
  maximum bit rate: 8229.584000 kbps
  media duration: 331464 ms
MPEG-4 file reader:
  object type: audio
  format: AAC
  average bit rate: 143.881000 kbps
  maximum bit rate: 143.881000 kbps
  media duration: 331498 ms
AAC 4 audio decoder:
  sample rate: 48000.000000 Hz

MPEG-4 file reader:
```

```
new position: 0.000000 s
MPEG-4 video decoder:
  profile and level: Advanced Simple Profile/Level 1
  video object layer width: 720
  video object layer height: 480
  video scan type: progressive
  frame rate: 25.000000 Hz (25/1)
videoOut autoMode:
  Determined [pal] as best match for current stream

MPEG-4 file reader:
  object type: video
  format: MPEG4
  average bit rate: 3228.704000 kbps
  maximum bit rate: 5172.072000 kbps
  media duration: 142767 ms

MPEG-4 file reader:
  object type: audio
  format: AAC
AAC 4 audio decoder:
  sample rate: 44100.000000 Hz

MPEG-4 video decoder :
Remark:
  menu for trick modes not enabled by default
  use command line argument -trickmode to activate menu
  profile and level: Advanced Simple Profile/Level 5
Press q to exit, or m to check memory
  video object layer width: 720
  video object layer height: 416
  video scan type: progressive
  frame rate: 23.976030 Hz (24000/1001)
```

6 Background information

<p>videoOutMode autoMode (frame rate conversion)</p>	<p>The application supports the so called auto detection of video standard. What actually is meant with this feature is that exolMpeg4 tries to use the output which best matches with the properties of the stream. In the current implementation it will look to the frame rate of the stream. If e.g. the frame rate of the stream is 25 Hz, it will use PAL output.</p> <p>Note that a lot of streams don't have an indication of the frame rate (the application will report "frame rate: variable". In that case it is not possible to determine automatically a suitable video out mode. With MPEG-4 it is in general not possible to determine the frame rate by looking at the number of lines of a frame. It may often give an indication of the frame rate, but you can't rely on this information as MPEG-4 has no restriction on the picture size.</p> <p>However the example application will use the number of lines to determine the video mode if it is not able to determine a match using the frame rate indication.</p> <p>WARNING:</p> <p>If you are building your own application, keep in mind that you can only use the frame rate indication to determine a best match. The number of lines may indicate the frame rate that is used, but in general this is not true for MPEG-4.</p>
<p>frame drop / frame repeated counters</p>	<p>The application reports the number of frames that are repeated or dropped. These numbers may indicate if the application is running real time. But be careful with interpreting these numbers! If the application is doing frame rate conversion, it will drop frames, e.g. to render a stream with 30 fps on a PAL TV. In that case, it is advised to check with a tool like TimeDoctor to investigate if there are load problems or that the application on regularly base is dropping frames to perform frame rate conversion.</p> <p>Also when the application reports that the frame rate is variable it is not so trivial to interpret these numbers.</p> <p>Due to audio/video synchronization, it is possible that frames are repeated or dropped at the start of the sequence in order to get in sync.</p> <p>WARNING:</p> <p>To determine if the application is not running real-time, you can only rely on the information reported by dedicated tools like TimeDoctor!</p>

ExolMpeg4 Application States

Once all the components are started, the exolMpeg4 application can go in the following states if the user provides the `trickmode` parameter as a command line argument.

Table 1-1: trickmode parameters

State	Description
PAUSE	Stream playback is paused.
RESUME	Resumes playback state in which stream can be brought back to play normally from these states: PAUSE, FASTFORWARD, FASTREWIND and SINGLESTEP.
JUMP	Application jumps to the provided location in the stream.
GETTIME	Obtains the current time of the stream.
START	Stream can be started again from the beginning location after STOP state.
STOP	Stream is forced to stop.
SINGLESTEP	Application is playing video frame-by-frame.
FASTFORWARD	Media file is fast forwarded to display next I-frame.
FASTREWIND	Media file is fast rewind to display previous I-frame.
I_ONLY	Only I frame is displayed.
IP_ONLY	Only IP frames are displayed.
RESTART	Stream is started again after reaching to EOS.

7 Frequently Asked Questions

- ***I cannot convince the application to open a file***

The MPEG-4 application and components supports the URL naming convention as defined in RFC 1738. For local files, you have to use the following format: "file://localhost/<path + filename>". You can also use the format "file:///<path + filename>".

For example, to open the file `myfile.mp4` in the folder `d:\mp4` on your local pc, you use the following:

```
-url file://localhost/d:/mp4/myfile.mp4" or "-url file:///d:/mp4/myfile.mp4
```

The syntax used in the past i.e. "file://<path + filename>", is not compliant with RFC 1738 and the application will not find the resource.

If your stream or folder name contains spaces, you must pass the spaces to the application encoded as `%20`, as required by RFC 1738. For example, to open "my stream" you use "my%20stream".

- ***No audio***

Check if the stream contains an audio track. The application prints an overview of the objects it found in the stream in the console.

- ***Video looks distorted***

The application does not take into account the pixel aspect ratio of the video stream and the pixel aspect ratio of the display device. Typical this effect is seen with half D1 movies. If you scale the image to fullscreen, the quality of the video decreases.

- **Artifacts in image after jumping (NetworkRead)**
 The graceful degradation just skips those frames that arrives too late at the decoder. As it does not take the frame type into account, it will just continue with decoding a random frame type after a frame was skipped. This can introduce artifacts, For example, when a P-frame is decoded after the previous I-frame was skipped.
- **Artifacts in image when graceful degradation is enabled**
 Refer to the answer in the above question.
- **Application reports "frame rate: variable", but I know that the frame rate is fixed**
 It may be possible that a stream is encoded with the characteristic of variable frame rate, but that each frame is encoded with the same time difference w.r.t. to the previous frame. In that case you actually have a fixed frame rate stream, but your encoder was not so intelligent to mark the stream as a fixed frame rate one.
- **Possibly a short period of silence after jump with MP4Read**
 Example: when one wants to jump to 80 seconds and the nearest previous random access point (RAP) is located at 77 seconds, video will start playing at 77 seconds while audio will join at 80 seconds. There will thus be silence during this 3 seconds.
- **Application does not seems to do anything after receiving the IP-address with targettcp**
 The ethernet driver supports auto negotiation, but there are problems reported when connecting the board directly to a 10 MB hub. In this case you only get an IP-address, but the application will not continue.
- **Strange behavior of application when multiple boards running exolMpeg4 are connected to the same network via targettcp.**
 When using targettcp, the ethTcp component is used which contains a hard coded MAC address. If you have the sources of ethTcp, you can change the MAC address at line 524 in the file `\net\comps\tmEthTcp.c` in order to avoid conflicts.
- **The application does not start with streaming after I have set the proxy address and proxy port.**
 In case the streaming server is behind a proxy, please be sure to provide the correct proxy IP address and port. When you provide the wrong proxy address or port, you will most likely not be notified by exolMpeg4. At this time only error returned by proxy- or http-servers are returned correctly to the client (file not found, access not authorized,...). Errors that occur because a proxy- or httpserver could not be found, are not returned correctly. This will be solved in a new release of the tmHTTPIODriver.

In case the streaming server is not behind a proxy, please don't set the proxy server or provide '0' as proxy server. This way the proxy server will not be set.
- **The application does not start with streaming when I use RPC sockets and I want to instant replay a file.**
 Instant replay of an URL generates a socket connection failure. This is a known issue logged as DS#4837.
- **When streaming from a RTSP server, we see rubbish or we start streaming from the middle of the file.**
 This means that the UDP packets of the previous connection are still being processed. This can be the case if we reset the target before an end of stream. The server may need a minute or 2 before it realises there is no client connected anymore.

- ***I see artifacts after a jump when playing from a server.***
The application sends the jump command just to the server. The server itself will do the actual jump. It is possible that the server itself does not jump to a key frame. If that is the case, you see artifacts in the image.
- ***I notice artifacts and freezes both in audio and video after a pause/resume when streaming from a Helix server.***
When you do a pause and some time later a resume, the server will not send out the data at normal speed, but a lot faster as it tries to catch up the time lost during the pause. This will lead to buffer overflows in the application. Data will be lost and you see artifacts during playback.
- ***The application does not jump to the position I specify.***
The application jumps to the nearest key frame located before the specified time. If the time you specified does not contain a key frame, the application will jump to a different position.
Note: When playing via RTSP the behavior depends on the jump implementation on the server itself.

- ***I see frequent interruptions in audio and video playback.***
 - It may be possible that the decoder is not able to decode the stream real-time. Try again with the option `-enableGracefulDegradation` enabled.
 - The interleaving of audio and video is not optimal. Check if your file is formatted in a progressive download friendly format. This is important when playing from a HTTP server.
 - Build the application with the `posix` diversity. With that diversity, read request are faster because with `posix` diversity there is more efficient file access due to less buffering (avoids memory copies).