**INTERNATIONAL ORGANISATION FOR STANDARDISATION**
**ORGANISATION INTERNATIONALE DE NORMALISATION**
**ISO/IEC JTC1/SC29/WG11**
**CODING OF MOVING PICTURES AND AUDIO**

**ISO/IEC JTC1/SC29/WG11 MPEG2012/M29230**
**April 2013, Incheon (KR)**

**Source** **Telecom ParisTech**
**Status** **For consideration at the 104nd MPEG meeting**
**Title** **GPAC Updates on ISOBMF**
**Author** Jean Le Feuvre, Cyril Concolato

# 1 Introduction

This contribution introduces some of the latest developments in GPAC [1], especially on the content production side. Comments and feedback are always warmly welcome !

# 2 HEVC Support

In the context of the 4Ever [2] project, support for HEVC has been added to GPAC according to the latest version of the latest HEVC specs and associated transport standards. The playback support is powered by the open-source OpenHEVC decoder [3]. The following are supported:

***Multiplexing HEVC bitstreams into ISOBMF***
All profiles should be supported, interlace is not supported yet. The support is based on the study text of HEVC file format.

| Operation | Command line example | Notes |
|---|---|---|
| HEVC Import | `MP4Box -add file.hvc -new file.mp4` | Adds file.hvc (Annex B format) to the given file.<br><br>The default import format uses "hvc1" storage. |
| | `MP4Box -add file.bin:FMT=HEVC -new file.mp4` | FMT is used to indicate the format is HEVC, and can be omitted if the file extension is hvc, hevc or 265 |
| | `MP4Box -add file.hvc:fps=50 -new file.mp4` | FPS is by default 25, and should be specified most of the time as VUI timing is not yet parsed. |
| HEVC | `MP4Box -info file.mp4` | Gives info on the file or on |

| | | |
|---|---|---|
| File inspection | `MP4Box -info ID file.mp4` | the track. |
| HEVC Export | `MP4Box -raw <trackId> file.mp4` | Exports an HEVC file to annex B format. |

***HEVC DASH***

All DASH operations from GPAC (client and MP4Box) are supported on HEVC, including bitstream switching modes using hev1. For more information on DASH and GPAC please refer to GPAC web site [1].

***MPEG-2 TS support***

All MPEG-2 TS operations from GPAC (client and MP42TS) are supported on HEVC. MP42TS can be used to generate TS files usable for DASH or for injection in modulation chains; it can also be used to send the TS over an UDP or RTP stream in unicast or multicast mode.

# 3 SVC Support

In the context of the OptiSaT2 [4] project, extended support for SVC has been added to GPAC. The playback support is powered by the open-source OpenSVC decoder [5]. The following are supported:

***Splitting SVC layers into ISOBMF***

SVC can now be imported in ISOBMF files with one track per layer, using Extractors to base layers. The syntax used is as follows:

| Operation | Command line example | Notes |
|---|---|---|
| SVC Import | `MP4Box -add file.avc:svcmode=split -new file.mp4` | Adds file.avc (Annex B format) to the given file. Each SVC layer is stored in a given track.<br>The same command can be used with an ISOBMF input to split an already imported file. |
| | `MP4Box -add file.avc:svcmode=merged -new file.mp4` | Adds file.avc (Annex B format or ISOBMF) to the given file. All SVC layers are stored in a single track. |
| | `MP4Box -add file.avc:svcmode=splitbase -new file.mp4` | Adds file.avc (Annex B format or ISOBMF) to the given file. All SVC layers are |

| | | |
|---|---|---|
| | | stored in a single track but the AVC base layer is stored in a different track. |

***MPEG-2 TS Encapsulation***

SVC can now be imported in MPEG-2 TS one PID per layer, using ISOBMF files where each SVC tracks are in different layers. No specific option is required for the MP42TS utility.

***RTP streaming and hinting Encapsulation***

SVC files can now be hinted or streamed over RTP using MP4Box, with one RTP stream PID per layer, using ISOBMF source files where each SVC tracks are in different layers. No specific option is required for the MP4Boxutility.

***Improved SVC Playback***

Thanks to the OpenSVC decoder, all these SVC transport modes are now supported for playback in GPAC. Switching between layers is achieved using ctrl+h (for high) and ctrl+l (low). SVC Switching now implies shutting down the associated network streams (stops multicast socket or PID filtering).

***SVC Future Work***

We plan to introduce SVC in DASH support in both MP4Box and the player for the next MPEG meeting.

# 4 WebVTT Support

MP4Box now supports basic operations on WebVTT files according to the ISO/IEC 14496-30 DIS and ISO/IEC 14496-12:2012/DAM2. Different files have been tested (regular movie files, chapter files with nested cues, metadata files with XML, and some files with invalid syntax). Here are the basic operations:

| Operation | Command line example | Notes |
|---|---|---|
| WebVTT Import | `MP4Box -add file.vtt file.mp4` | Adds a track to the given file:<br>• Overlapping cues are split into non-overlapping cues and stored in samples<br>• Comments and in-between cues text are ignored (for now) |
| | `MP4Box -add file.vtt:lang=en:layout=800x600x100x100 file.mp4` | Sets the language and track layout information |
| | `MP4Box -add file.vtt:delay=1000 file.mp4` | Adds an edit list |
| | `MP4Box -add file.vtt:dur=20 file.mp4` | Import only a certain duration of the input file |

| | | |
|---|---|---|
| SRT Import as WebVTT track | `MP4Box -add file.srt:FMT=VTT file.mp4` | Forces the import of SRT files to generate a WebVTT track instead of a 3GPP Timed Text track |
| WebVTT Export | `MP4Box -raw <trackId> file.mp4` | Exports a WebVTT file where consecutive cues from different samples and with the same id, settings and payload are merged |
| | `MP4Box -raw <trackId>:vttnomerge file.mp4` | Dump samples as-is w/o merge, and with empty samples |
| | `MP4Box -raws <trackId>:<sampleNumber> file.mp4` | Exports only one VTT sample |
| File Editing | `MP4Box -split-chunk start:end file.mp4` | Split a track into 2 files |
| | `MP4Box -cat file1.mp4 -cat file2.mp4 file3.mp4` | Concatenation of tracks (second header is ignored) |
| DASH | `MP4Box -frag 1000 file.mp4` | |
| | `MP4Box -dash 1000 file.mp4` | Fragments and segments the file and produces an MPD containing:<br>`<Representation id="1" mimeType="video/mp4" codecs="wvtt" width="800" height="600" startWithSAP="1" bandwidth="1939">` |
| RTP | `MP4Box -hint file.mp4` | Not working yet |

The parsing of WebVTT files is meant to be conformant to the parsing algorithm as specified in the W3C specification, with the following refinements for carriage in ISOBMF:

- The WebVTT signature line and the lines gathered in the header loop of the parsing algorithm are both stored in the configuration string of the PlainTextSampleEntry.
- It is not clear from the standard if U+000D CARRIAGE RETURN U+000A LINE FEED (CRLF) pairs and single U+000D CARRIAGE RETURN characters should be replaced by U+000A LINE FEED characters or not before storage. Additionally, in both cases, it is not clear if the CueIDBox, the CueSettingsBox and the CuePayloadBox should contain the trailing LF caracters. In the current implementation, MP4Box:
    - Does not replace U+0000 NULL characters by U+FFFD REPLACEMENT CHARACTERs, NULL characters are not handled;
    - Does not replace CRLF or CR by LF; and
        - Does not store trailing CRLF characters in CueIDBox and CueSettingsBox boxes. The leading space separating the timings from the settings is not stored either in the CueSettings box.

- Does keep and store a trailing LF at end of the signature line and at the end of each header and cue payload lines.

Boxes such as CueSourceIDBox, CueLocalIDBox, CueEndTimeBox, CueStartTimeBox are not supported, as they are not needed.
The CueTimeBox is supported in reading/writing but not generated during parsing nor exploited during export, yet.

# 5 Generic formatting tools for testing new tracks types and sample formats

MP4Box has featured since its early days the ability to import unknown media formats through the NHML XML description. The NHML language has been extended to add custom data for both STSD extension and sample format. This means that it is possible to construct STSD and sample formats for new media types through a simple description. The following example shows how font data stream (cf contribution m29226) have been tested in GPAC.

```
<?xml version="1.0" encoding="UTF-8" ?>
<NHNTStream version="1.0" timeScale="1000" trackID="1" mediaType="fdsm" mediaSubType="fnt1">

<DecoderSpecificInfo>
<BS id="size" bits="32" value="24"/> <!-- box size is 4+4+3+strlen(TriodPostnaja)-->
<BS id="type" fcc="fntC"/>
<BS id="fontFormat" bits="7" value="1"/>
<BS id="storeFont" bits="1" value="0"/>
<BS id="fontName" bits="8" text="TriodPostnaja"/>
<BS id="fontSubsetID" bits="7" value="1"/>
<BS id="reserved" bits="1" value="1"/>
</DecoderSpecificInfo>

<NHNTSample DTS="0" isRAP="yes">
<BS id="fontFormat" bits="7" value="1"/>
<BS id="storeFont" bits="1" value="0"/>
<BS id="fontName" bits="8" text="TriodPostnaja"/>
<BS id="fontSubsetID" bits="7" value="2"/>
<BS id="fontSubsetExtensionFlag" bits="1" value="1"/>
<BS id="fontSpecInfoLength" bits="8" value="0"/>
<BS id="fontData" mediaFile="TriodPostnaja_subsets/TriodPostnaja_CyrillicCaps.ttf" />
</NHNTSample>
…
</NHNTStream>
```

For the current time, box size has to be precomputed, but a future version will include new type to describe boxes.

## References


[1] GPAC, http://gpac.sourceforge.net
[2] 4Ever Project, http://www.4ever-project.com/
[3] OpenHEVC, https://github.com/OpenHEVC/openHEVC
[4] OptiSat2 Project, http://www.optisat2.com/
[5] OpenSVCDecoder Project, http://sourceforge.net/projects/opensvcdecoder/