A Media Processing Implementation for ISDTV Middleware with Optional Hardware Acceleration Support

Jean F. G. Quadro, Tiago H. Trojahn, Juliano L. Gońvalves, Luciano V. Agostini, Leomar S. da Rosa Junior
{jeanquadro, thtrojahn, juliano, agostini, leomarjr}@inf.ufpel.edu.br

Group of Architectures and Integrated Circuits – GACI
Technology Development Center - CDTec
Federal University of Pelotas – UFPel
Pelotas – Brazil

Abstract

The media decoding is one of the fundamental components required in a Digital TV standard, mainly because High Definition uncompressed videos cannot be transmitted due to bandwidth constraints. Currently, one of the most advanced Digital TV standards is the Brazilian International Standard for Digital Television (ISDTV), featuring a return channel, high-definition content and a middleware named Ginga. This paper describes a media decoder, with optional hardware acceleration support, compatible with Ginga. Results present a performace analyze of the Media Processing with and without hardware acceleration support when reproducing video stream with different resolutions.

1. Introduction

The video stream, normally broadcasted through the air or using a cable, in the Digital TV standards is a coded version of the original video created directly by the producers. The coding process is used to reduce the video stream size: a single frame using 3-bytes for coloring in the 1920x1080 resolution uses about 5.93 Mega Bytes (MB) of size. In a video with 30 frames per second (FPS), it is required a constant transmission rate of about 178 MB per second, much more than an average internet connection or even a local network can provide nowadays.

In addition, some of the Digital TV standards provide support to multiple audio and subtitle streams and, in some cases, even applications to be used in a single stream, known as Transport Stream (TS), requiring even more bandwidth of the transmission channel. For this reason, coding standards are required to provide high-definition and high-quality content to TV viewers at reasonable costs.

After coded, the data needs to be decoded to be effectively used. In terms of Digital TV, the decoded video and audio stream can be showed right to the viewer or manipulated by other application, increasing the complexity of the decoding procedure.

This work presents a video decoding application, named Media Processing, able to handle both audio, video and subtitle streams using an open-source library. The Media Processing was developed to be compatible with the Ginga, the International System for Digital Television (ISDTV) middleware and its audio, video and subtitle specifications.

The ISDTV is an emergent Digital TV standard developed in Brazil which presents several peculiar aspects that innovates and improves the quality of Digital TV services. For example, the Brazilian Digital TV system adopts the H.264/AVC [1] as video coding standard, differently from the established American, European and Japanese Digital TV standards. Also, the Ginga middleware provides support to interactivity through a return channel, delivering more possibilities to Digital TV applications like bank accounts access, shopping, and mass advertising.

This work is organized as follows: section 2 presents the Media Processing and its main feature. Section 3 presents the performance analysis of the Media Processing. Finally, section 4 presents conclusions and future works.

2. The Media Processing

The libVLC [2] library at version 1.1.5 was used to implements the Media Processing module. The Media Processing was developed to be able to provide all the basic features expected from a video decoder, including the methods:

- Play – The most used and the very fundamental feature of all media decoders. It is responsible to start all the decoding operation and provide the decoded stream to the viewer or to another application.
- Pause – Responsible to pause the operation without deallocating the resources to continue the decoding.
- Stop – The finalization method, used to deallocate all resources and shutdown the decoding process.

However, the ISDTV standard requires more features of the Media Processing than these basic methods. For example, the Media Processing must provide data to other modules and, sometimes, directly to applications...
installed by the viewers. This characteristic makes the Media Processing much more complex and consequently prone to eventual failures. Thus, the Media Processing implemented in this paper presents a wide exception control in all methods provided to others modules or applications in order to offer a smooth an easy utilization by applications programmers. Exception like malformed inputs streams and mistakes in calling methods are trivially controlled. Some more complex exception handling, like race conditions and allocation problems are performed directly by libVLC.

Besides the main methods of reproduction, auxiliary methods were implemented to applications and viewer usage. These methods are divided in two large groups, getter and setter methods. The details of each group are explained below:

- **Getters methods:** A large number of getters of various video, audio and subtitle information like video resolution and audio volume. These methods are important to other modules or applications that may require details of the media being reproduced.
- **Setter methods:** A set of setters methods that can change the reproduction behavior, like video brightness, audio volume and subtitle stream. These methods are fundamental to viewers to control the reproduction, reducing the volume for example. Applications can also use these methods.

A part from that, the Media Processing presents a large set of implemented features, listed below:

- **Video Reproduction:** The Media Processing support a wide range of video coding standards, including the H.264/AVC used in the ISDTV.
- **Audio Reproduction:** In terms of audio, the Media Processing provides full support to the mp3 and the AAC audio coded streams.
- **Subtitles:** The SubRip (SRT) and the Advanced Substation Alpha (ASS) are supported. Multiple subtitle streams in the same media stream are supported too.
- **Demux:** The Media Processing has a built-in Demux which can recognize the ISDTV Transport Stream and other input containers like Audio Video Interlace (AVI) and the Matroska Video (MKV).
- **Screenshot:** The Media Processing provides methods permitting a viewer to take screenshot and save it in a lossless JPEG format [3].
- **Optimizations:** The Media Processing can use all the optimizations present on the libVLC library like SSE and SSE2. A video card accelerated decoding can be used when the computer have a compatible video card [4], aiming to reduce the decoding stress to processor.
- **Input:** The input stream can be a file containing the streams which needs to be demuxed, a basic media stream already demuxed by another application or module, or an interface to an input device. A remotely located media can be accessed through a connection using the HTTP, FTP or RTP protocols.
- **Output:** The current Media Processing implementation features a built-in X.Org and a FrameBuffer video output. ALSA and DirectX WaveOut are the audio outputs currently implemented.

Fig. 1 shows the complete procedure to reproduce an input stream in a Digital TV device.

![Fig. 1- Basic audio, video and subtitles decoding procedure.](image)

The process described in Fig. 1 presents the Input, where the stream is recognized as valid input. The validated input goes to Demux, where the input stream is demuxed into audio, video and subtitles streams. Each elemental stream goes to the corresponding decoder and, after the decoding process, the output shows the resulting video and subtitle on-screen and the audio on TV speakers, for example.

An alternative Media Processing for the ISDTV is presented in [5], but it does not present audio decoding, video card hardware acceleration and some other features.

### 3. Performance Analysis

The Media Processing implementation was evaluated to verify the performance when reproducing input video and audio streams during nine minutes. This experiment was repeated three times in order to deliver more accurate results. The performance was measured in terms of processor usage and memory cost. The samples were generated with the procps application in an Ubuntu 10.04 operating system with recommended installation settings.

As benchmarks, the videos, were used the Big Buck Bunny [6], the Elephants Dream [7] and the Sintel [8] animations for input streams in three different resolutions: 848x480 (480p), 1280x720 (720p) and 1920x1080
(1080p). The progressive scan was utilized, dispensing the default libVLC De-Interlace filter. Video streams were coded using the x264 [9] in version 1649, and audio streams were coded with the Nero AAC Codec [10] at version 1.5.4.0.

All videos were coded using 10.000 Kbps (10 Mbps) for Average Bitrate (ABR) in High Profile at 3.2 level. It was used three reference frames, CABAC and Trellis activated and a 4:2:0 YCbCr sub sampling. The video frame rate was converted to 30 FPS to be ISDTV compatible.

The audio was coded in Constant Bitrate (CBR) mode with 192 Kbps using the Low Complexity (known as AAC-LC) in version 4. It was used 48 kHz for output frequency.

An Intel Core i5 760 with 2.8 GHz and 4 GB of RAM was the basic platform for the experiment. Three video cards configurations were used:

- The first configuration was named Computer A, without any dedicated video card and using only the software-based decoding of Media Processing.
- The second, named Computer B, using a XFX GeForce 9500GT with 512 MB of DDR2 RAM memory and the Media Processing hardware accelerated decoding activated.
- The last one, named Computer C, using a Zotac GeForce GTX 470 with 1280 MB of DDR5 RAM memory and the Media Processing hardware accelerated decoding activated.

3.1. Results

The processor usage, in percentage, and memory cost, in MegaBytes, results for the performance tests are presented in Fig. 2 and Fig. 3 respectively.

![Fig. 2 - Processor usage (%) of Media Processing.](image)

![Fig. 3 - Memory cost (MB) of Media Processing.](image)

The results show a processor usage increment when increasing the video resolution, which can be explained with the increase of the pixels which needs to be processed. The Computer A and Computer B presented an equivalent performance in terms of processor usage. However, the Computer C presents a better performance. The processing power of GeForce GTX 470 when compared to GeForce 9500GT suggests that the reduction of the main processor usage is directly related to the increase in video card processing power. Also, the equivalency between the Computer A and Computer B test results suggest that the Computer B video card is not able to reduce the load in the main processor in a satisfactory manner when decoding H.264/AVC videos.

In terms of memory, the implementations using the hardware acceleration in video card presented a larger memory cost. This behavior can be explained by the decoding methodology that libVLC adopts. Basically, the coded frame is sent to the video card and the decoded frame is kept in memory while not displayed. The
memory cost increases even more because several frames are kept in the memory to prevent synchronization problems. As with processor usage tests, the memory cost has risen as the increase in video resolution.

4. Conclusions and Future Works

This paper presented a decoding module, the Media Processing, for the Ginga ISDTV middleware. A performance test was performed to evaluate the component when using some of the available methods.

The ISDTV plays an important role in South America, where the standard was adopted in other countries like Argentina, Chile and Venezuela. Besides, Africa countries like South Africa and Democratic Republic of Congo are currently testing the ISDTV for possible use as their Digital TV standard. The Media Processing presented in this paper can be used in all these countries to achieve video, audio and subtitle decoding features.

Our Media Processing performance tests demonstrate that better results can be achieved in terms of processor usage when using a video card with high processing power with hardware acceleration activated. When using a video card with lower processing power, the advantage becomes negligible.

In terms of memory, the Media Processing hardware acceleration increases more than two times the memory needs when compared with the full software solution. The Computer B, without a significant better processor usage when compared with Computer A and with two times the memory cost, becomes the worst solution analyzed.

If the device presents a compatible video card and enough memory, the best solution consists in to activate the Media Processing video card hardware acceleration. On the other hand, if memory is limited, the full software decoding is better, but with a significant increase in processor usage.

Finally, it is important to mention that these results also point to the fact that an Application Specific Integrated Circuit (ASIC) dedicated to decode High Definition videos for the Brazilian Digital TV standard could be investigated. Powerful CPU, GPU and video cards are not feasible to be embedded in affordable Digital TV equipments.

For future works it is intended to do a wider performance tests using some other optimizations based in the processor architecture. It is also intended to use the Media Processing in a portable device to analyze its behavior when running streams with different video resolutions. Finally, another version of Media Processing is being implemented using the NVIDIA CUDA framework to achieve even better performance with a computer equipped with a dedicated video card.

5. References