elecard

VIDEO COMPRESSION GURU

Elecard GStreamer Codec SDK v.2.1

Reference Manual

Notices

Elecard GStreamer Codec SDK v.2.1 Reference Manual

First edition: November 2013.

Date modified: January 17, 2019.

For information, contact Elecard.

Tel: +7 (3822) 488-580.

More information can be found at: https://www.elecard.com

For Technical Support, please contact the Elecard Technical Support Team: tsup@elecard.com

CONTENTS

1 Introduction

1.1 About This Document

1.1.1 Purpose

This document provides an overview of the installation, set up and use of Elecard GStreamer Codec SDK. It includes information about the structure of the SDK, provides a quick overview of the GStreamer fundamentals and features a detailed description of the components and sample applications.

1.1.2 Topics Covered

The following lists the topics covered in this document:

- **Section 1: Introduction** – provides a general overview of the SDK and describes the purpose of the document.
- **Section 2: Getting Started** – describes how to install, uninstall, and run the SDK. This section also provides information on the Elecard GStreamer Codec SDK folder structure.
- **Section 3: Gstreamer Overview** – provides a brief description of the basic concepts of the GStreamer framework, including component model, filters, pipelines, bins, and more.
- **Section 4: Sample Applications** – describes samples included in Elecard GStreamer Codec SDK.

1.1.3 Related Documentation

In order to thoroughly understand the GStreamer® technology, we strongly recommend that you read the following documentation:

- GStreamer Application Development Manual. You can find this document at: http://gstreamer.freedesktop.org/documentation/

1.2 Preface

Elecard GStreamer Codec SDK is a software development kit intended to enable programmers to develop multimedia applications using Elecard components within the GStreamer® technology.

The Elecard GStreamer Codec SDK package includes the following:

1.2.1 Documentation

Elecard GStreamer Codec SDK documentation – consists of the following documents:

- Elecard GStreamer Codec SDK Reference Manual (this document);
- Elecard GStreamer Codec SDK Release Notes (ReleaseNotes.txt);
- Elecard License Agreement (license_EULA.rtf);
- Party Licenses (License folder).

1.2.2 Components

This section provides a quick overview of the GStreamer plugins included into the SDK.

Table 1. Elecard GStreamer Codec SDK Components

Component	Description	File Name
Elecard AVC Video Decoder	GStreamer plugin for decoding ISO/IEC 14496 part 10 AVC / ITU-T Recommendation H.264 video streams.	libeavcdec.so/ eavcdec.dll
Elecard AVC Video Encoder	Software module for video encoding into AVC/H.264 (MPEG-4 Part 10, ISO/IEC 14496-10) streams.	libeavcenc.so/ eavcenc.dll
Elecard MPEG-2 Video Decoder	GStreamer plugin intended for software-only decoding MPEG-2 video (ISO/IEC 13818-2) and MPEG-1 video (ISO/IEC 11172-2) streams.	libem2vd.so/ em2vd.dll
Elecard MPEG-2 Video Encoder	GStreamer plugin intended for software-only encoding into MPEG-2 video (ISO/IEC 13818-2) and MPEG-1 video compatible (ISO/IEC 11172-2) streams.	libem2venc.so/ em2venc.dll
Elecard HEVC Video Decoder	GStreamer plugin for decoding ISO/IEC 23008-2 MPEG-H Part 2 / ITU-T H.265 video streams.	libehevcdec.so/ ehevcdec.dll
Elecard AAC Audio Decoder	GStreamer plugin for the software-only decoding of AAC and HE-AAC audio streams.	libeaacdec.so/ eaacdec.dll
Elecard AAC Audio Encoder	GStreamer plugin for audio encoding into AAC streams.	libeaacenc.so/ eaacenc.dll
Elecard MPEG Audio Decoder	GStreamer plugin for the software-only decoding of MPEG-1, MPEG-2, MPEG-2.5 and LPCM audio streams.	libempegaudiodec.so/ empegaudiodec.dll
Elecard MPEG Audio Encoder	GStreamer plugin for audio encoding into MPEG streams.	libempegaudioenc.so/ empegaudioenc.dll
Elecard MP4 Demultiplexer	GStreamer plugin that provides demultiplexing of ISO/IEC 14496-14 file format (MP4) and 3GPP2 System streams into a MPEG-4, H.263, AVC/H.264 video streams and AAC, AMR, MPEG-1/2 Audio Layer 3 audio streams.	libemp4dmx.so/ emp4dmx.dll
Elecard MP4 Multiplexer	GStreamer plugin intended for the generation of MPEG-4 (Intermedia Format (MP4)) System Streams.	libemp4mux.so/ emp4mux.dll
Elecard MPEG Demultiplexer	GStreamer plugin for splitting of MPEG-1 System Streams (ISO/IEC 11172-1), MPEG-2 Program and Transport Streams (ISO/IEC 13818-1) into video and audio streams.	libempgdmx.so/ empgdmx.dll
Elecard MPEG Multiplexer	GStreamer plugin that provides the MPEG-2 Transport Stream (TS) or MPEG-2 Program Stream (PS) generation.	libempegmux.so/ empegmux.dll
Elecard MPEG Push Demultiplexer	GStreamer plugin for the software-only splitting of MPEG-1 System Streams, MPEG-2 Program Streams and MPEG-2 Transport Streams into video and audio streams.	libempgpdmx.so/ empgpdmx.dll
Elecard MKV Demultiplexer	GStreamer plugin designed for software-only splitting of Matroska Multimedia Container and WebM Container into video and audio streams. The filter is based on/contains/uses libmatroska, libebml.	libemkvdmx.so/ emkvdmx.dll, libebml_dmx.so/ ebml_dmx.dll, libmatroska_dmx.so/ matroska_dmx.dll

Elecard MXF Demultiplexer	GStreamer plugin designed for software-only splitting of Material Exchange Format (MXF) into video, audio and ancillary data streams. The filter is based on/contains/uses libmxf.	libemxfdmx.so/ emxfdmx.dll
Elecard Network Source	GStreamer plugin for receiving media data from the network. It receives the RTP and UDP packets and feeds the filter graph with stream data contained in these packets.	libenwsrc.so/ enwsrc.dll
Elecard Network Sink	GStreamer plugin for broadcasting media data to the network. It is capable of sending RTP, and UDP packets and supports the announcement of its data session via SAP (SDP) packets.	libenwsink.so/ enwsink.dll
Elecard HLS Source	GStreamer plugin for receiving HLS streams.	libehlssrc.so/ ehlssrc.dll
Elecard HLS Sink	GStreamer plugin intended for content preparation via Apple HTTP Live Streaming (HLS) protocol.	libehlssink.so/ ehlssink.dll
Elecard RTSP Source	GStreamer plugin for receiving media data from the network. It is capable of receiving and sending RTSP commands, receiving RTP, UDP, TCP packets.	libertspsrc.so/ ertspsrc.dll
Elecard TimeMarker	GStreamer plugin intended for stream analysis. It defines the type of data, extracts time stamps from the stream and sets these time stamps on the output media samples.	libetimemarker.so/ etimemarker.dll
Elecard Deinterlacer	GStreamer plugin intended for conversion of interlaced video into progressive scan format.	libedif.so/ edif.dll

1.2.3 Sample Applications

The Elecard GStreamer Codec SDK samples are simple applications that demonstrate use of the Elecard components. The samples are written in C++. The following table provides a brief overview of each sample. For further details, see the Sample Applications Section.

Table 2. Elecard GStreamer Codec SDK Sample Applications

SDK samples	Sample Name	Description	File Name
Network	**NWPlayer**	Console sample application that plays media streams from a network and allows saving of the received media data to a data storage. This sample illustrates use of the Elecard Network Source and Elecard MPEG Push Demultiplexer plugins.	nwplayer
	NWServer	Console sample application that demonstrates broadcasting MPEG TS media data to a network. It is capable of sending RTP and UDP packets.	nwserver
	NWServerTranscoder	Console sample application that demonstrates transcoding, multiplexing and broadcasting media data to the network.	nwservertranscoder
Decoders	**SimplePlayer**	Console sample application that demonstrates use of the Elecard video and audio decoders for playback of media files.	simpleplayer

	PlaybinSample	Console sample application that demonstrates use of the Elecard components inside playbin element.	playbinsample

1.3 Requirements

Elecard GStreamer Codec SDK has the following hardware and software requirements:

1.3.1 Hardware Requirements

Minimum hardware requirements:

- SSE-enhanced CPU (Intel® Pentium III, Celeron, AMD® Athlon, Opteron, etc.)
- 1 GB RAM
- Any VGA card

1.3.2 Software Requirements

- CentOS 7.4 x86_64, GStreamer 1.0 (version 1.10.4)
- Windows x64, GStreamer 1.0 (version 1.12.4)

Note: *Installation of the ffmpeg plugins is recommended.*

1.4 Technical Support

For technical support contact Elecard Technical Support Team: tsup@elecard.com.

2 Getting Started

2.1 Introduction

The following section details the procedures for installing Elecard GStreamer Codec SDK. In addition, it provides the description of the SDK folder structure.

2.2 Installing Elecard GStreamer Codec SDK

If Elecard GStreamer Codec SDK of an earlier version is installed on your computer, remove it before installing a later version.

The SDK installation process varies depending on the operating system.

To install the SDK on CentOS, run the following command:

```
$ sudo rpm -i ecodecsdk-xx.rpm
```

To install the SDK on Windows OS:

1. Run Elecard GStreamer Codec SDK setup.
2. The *Elecard GStreamer Codec SDK setup* window will appear. Follow the onscreen instructions of the installation wizard.
3. After installation, edit the GST_PLUGIN_PATH environment variable by adding the following path to the components: *(Elecard GStreamer Codec SDK root)\lib\gstreamer-1.0* and the PATH environment variable by adding the following path to the components: *(Elecard GStreamer Codec SDK root)\lib*.

2.3 Uninstalling Elecard GStreamer Codec SDK

The SDK uninstallation process varies depending on the operating system.

To uninstall the SDK on CentOS, run the following command:

```
$ sudo rpm -e ecodecsdk
```

To uninstall the SDK on Windows OS:

1. Click Start → Programs → Elecard → Elecard GStreamer Codec SDK → Uninstall;
2. Follow the onscreen instructions to complete removal of the SDK.

2.4 Components Activation

Most of the encoder and decoder (video and audio) filters from the SDK have a copy-protection mechanism: without activation these filters operate in an evaluation mode (e.g. overlay logo on the video). After activation filters operate in a demo mode (e.g. still overlay logo on the video, but without restrictions imposed on an expired mode). When the development is finished, the OEM pack with the components used in the product must be ordered (licensed) from Elecard.

2.5 Running Elecard GStreamer Codec SDK Sample Applications

If the SDK is installed on Linux OS, the SDK sample applications are located in the

/usr/share/elecard/codec-sdk/bin folder. The SDK sample applications can be started from any folder.

If the SDK is installed on Windows OS, the SDK sample applications are located in the *(ProgramFiles root)\Elecard\Elecard GStreamer Codec SDK\bin* folder.

2.6 Describing SDK Folder Structure

After installing Elecard GStreamer Codec SDK, its folder will appear in the destination folder specified during installation. On Linux OS, usually it is the *\usr\share\elecard\codec-sdk* folder. On Windows OS, usually it is the *(ProgramFiles root)\Elecard\Elecard GStreamer Codec SDK* folder.

The SDK folder contains:

1. **bin** – contains the executable binaries for the SDK samples;
2. **doc** – includes all SDK-related documentation including Elecard license and third party licenses;
3. **/usr/share/elecard/lib/x86_64-linux-gnu/gstreamer-1.0** – contains the 64-bit Elecard GStreamer components on Linux OS;

   (Elecard GStreamer Codec SDK root)\lib\gstreamer-1.0 – contains the 64-bit Elecard GStreamer components on Windows OS;
4. **src/samples/** – contains the source codes of the Elecard GStreamer Codec SDK samples.

3 GStreamer Overview

3.1 Introduction

The following section provides an overview of the GStreamer technology.

This section uses information from the GStreamer Application Development Manual. You can find this document at: http://gstreamer.freedesktop.org/documentation/.

3.2 GStreamer 1.0 Installation

To install GStreamer 1.0 on CentOS, run the following command:

```
$ sudo yum install gstreamer1.0*
```

To install GStreamer 1.0 on Windows OS:

Go to the official GStreamer website https://gstreamer.freedesktop.org.

Find and download the MSI installer of the version specified in Software Requirements (see 1.3.2).

Run the MSI installer and follow the onscreen instructions.

Add GST_PLUGIN_PATH into the list of environment variables. When adding, specify the variable name and the path to plugins. The example of the path is below:

```
GST_PLUGIN_PATH="C:\gstreamer\1.0\x86_64\lib"
```

Add a path to the GStreamer binary directory to the PATH environment variable. The example of the path is below:

```
"C:\gstreamer\1.0\x86_64\bin"
```

3.3 GStreamer Main Terms and Definitions

GStreamer is a framework for creating streaming media applications. The fundamental design comes from the video pipeline at Oregon Graduate Institute, as well as some ideas from DirectShow. The framework is based on plugins that will provide the various codec and other functionality. The plugins can be linked and arranged in a pipeline. This pipeline defines the flow of the data.

The GStreamer core function is to provide a framework for plugins, data flow and media type handling/negotiation. It also provides an API to write applications using the various plugins.

GStreamer plugins could be classified into:

- protocols handling (file, http, rtsp...);
- sources: for audio and video (involves protocol plugins);
- formats: parsers, formaters, muxers, demuxers, metadata, subtitles;
- codecs: coders and decoders;
- filters: converters, mixers, effects, etc.;
- sinks: for audio and video (involves protocol plugins).

GStreamer is packaged into:

- gstreamer: the core package;

- gst-plugins-base: an essential exemplary set of elements;
- gst-plugins-good: a set of good-quality plug-ins under LGPL;
- gst-plugins-ugly: a set of good-quality plug-ins that might pose distribution problems;
- gst-plugins-bad: a set of plug-ins that need more quality;
- gst-libav: a set of plug-ins that wrap libav for decoding and encoding;
- a few other packages.

3.4 GStreamer Basic Concepts

3.4.1 Elements

An *element* is the most important class of objects in GStreamer. You will usually create a chain of elements linked together and let data flow through this chain of elements. By chaining together several such elements, you create a *pipeline* that can do a specific task, for example media playback or capture. GStreamer ships with a large collection of elements by default, but if needed, you can also write new elements.

3.4.2 Pads

Pads are element's input (named sink) and output (named src), where you can connect other elements. They are used to negotiate links and data flow between elements in GStreamer. Pads have specific data handling capabilities: a pad can restrict the type of data that flows through it. Links are only allowed between two pads when the allowed data types of the two pads are compatible. Data types are negotiated between pads using a process called caps negotiation. Data types are described as a *GstCaps*.

3.4.3 Containers

Elements operate inside *containers*. Container controls a message sending from one element to another, manages element states. There are two container types:

- Bin;
- Pipeline.

A *bin* is a container for a collection of elements. Since bins are subclasses of elements themselves, you can mostly control a bin as if it were an element, thereby abstracting away a lot of complexity for your application. Usually sin is used for creation group of elements that must perform some action. For instance, **decodebin** is a stream decoding element that automatically selects the stream processing elements depending on data type, and lightens the work of program builder.

A *pipeline* is a top-level bin. It provides a bus for the application and manages the synchronization for its children. As you set it to PAUSED or PLAYING state, data flow will start and media processing will take place. Once started, pipelines will run in a separate thread until you stop them or the end of the data stream is reached.

3.4.4 GStreamer Utility Applications

The gstreamerXX-tools package contains utilities **gst-launch-XX** and **gst-inspect-XX** (where XX – the GStreamer version number). For GStreamer 1.0 the utility names are **gst-launch-1.0** and **gst-inspect-1.0** respectively.

gst-launch is a simple script-like command line application that can be used to test pipelines.

gst-inspect can be used to inspect all properties, signals, dynamic parameters and the object hierarchy of an element.

For the utility detailed descriptions, see the *21* and *23.4* sections of the *GStreamer Application Development Manual* document.

3.4.5 Getting Information About Installed Elements Using gst-inspect

To query the information about an element with the **gst-inspect** utility, it is needed to specify the element name as a parameter.

A simple command line that queries the *filesrc* element information looks like:

```
$ gst-inspect-1.0 filesrc
```

To get an Elecard component name, remove “lib” from the component filename. For instance, the Elecard MPEG Multiplexer filename is *libempegmux.so*. The component name is *empegmux* respectively:

```
$ gst-inspect-1.0 empegmux
```

3.4.6 Building Pipeline Using gst-launch

A simple command line that starts playback of the *audio.mp3* file looks like:

```
$ gst-launch-1.0 filesrc location=audio.mp3 ! mad ! audioresample ! autoaudiosink
```

A simple command line for transcoding video_only.mpg to AVC and multiplexing to MPEG TS using Elecard AVC Video Encoder and Elecard MPEG Multiplexer components looks like:

```
$ gst-launch-1.0 filesrc location=video_only.mpg ! decodebin ! eavcenc ! empegmux ! filesink location=dump.mpg
```

4 Sample Applications

4.1 Introduction

This section describes the sample applications included in Elecard GStreamer Codec SDK. If the SDK is installed on Linux OS, they are available from the SDK folder */usr/share/elecard/codec-sdk/src/samples*. If the SDK is installed on Windows OS, they are available from the SDK folder *(ProgramFiles root)\Elecard\Elecard GStreamer Codec SDK*.

Sample applications are CMake-based C++ applications. You need to install the GStreamer libraries (see 3.2), Make and CMake **before** building samples included in Elecard GStreamer Codec SDK.

4.1.1 Building Sample Applications on Linux OS

To build the sample applications with the KDevelop IDE, it is necessary to perform the following actions:

1. Open the CMake file */usr/share/elecard/codec-sdk/src/samples/CMakeLists.txt* in the KDevelop IDE.
2. In the opened dialog box **Configure a build directory for...**, check or specify values in the **CMake Binary** and **Build Directory** fields, and click **OK**.
3. You see *codec-sdk* project in the **Projects** tool window. The **Projects** tool window has two parts: the top half (titled **Projects)** lists all of your projects and lets you expand the underlying directory trees. The bottom half (titled **Project Selection**) lists a subset of those projects that will be built if you choose the menu command **Project → Build selection** or press **F8**.
4. Choose the menu command **Project → Build selection** or press **F8**.

To build the sample applications using console, run the following commands sequentially:

```
$ mkdir /usr/share/elecard/codec-sdk/src/samples/build

$ cd /usr/share/elecard/codec-sdk/src/samples/build

$ cmake ../

$ make
```

4.1.2 Building Sample Applications on Windows OS

To build the sample applications with the Microsoft Visual Studio IDE, it is necessary to create a solution using CMake. You may use cmake-gui.exe or call cmake.exe from a command prompt.

To create a solution using **cmake-gui.exe**, perform the following actions:

1. Launch **cmake-gui.exe**.
2. In the **Where is the source code** field set the following path → *(Elecard GStreamer Codec SDK root)\src\samples*.
3. In the **Where to build the binaries** field set the path for configuration files to be created, for example: *(Elecard GStreamer Codec SDK root)\src\samples\build.*
4. Click **Configure** button.

5. Select the installed version of Visual Studio for creating a solution file. Choose **Win64** version and click **Finish**.
6. Wait for the configuration stage completion and then click **Generate**.
7. Go to the directory you created in step **3**.
8. Open **elecard_codec_sdk.sln**.
9. In the **Solution Explorer** tab right-click on **ALL_BUILD** and choose **Build** or press **F7**.

To create a solution using **cmake.exe**, it is necessary to perform the following actions in the command prompt:

1. Go into the *(Elecard GStreamer Codec SDK root)\src\samples* directory:

   ```
   cd (Elecard GStreamer Codec SDK root)\src\samples
   ```

2. Create a **Build** directory and move into it:

   ```
   mkdir build
   ```

   ```
   cd build
   ```

3. To create a Visual Studio 2013 (version 12) solution using v120 toolset, run the following command:

   ```
   cmake -G "Visual Studio 12 Win64" -T v120 ..
   ```

If you need to create a solution for the other Visual Studio version or using another toolset, change the command.

4. Complete the steps **8** and **9** described above or run the following command:

   ```
   cmake --build . --config Debug --clean-first
   ```

4.2 Elecard GStreamer Codec SDK Sample Applications

The following describes the Elecard GStreamer Codec SDK samples, including:

- NWPlayer;
- NWServer;
- NWServerTranscoder;
- SimplePlayer;
- PlaybinSample.

4.2.1 NWPlayer

4.2.1.1 Description

NWPlayer is a console sample application that plays media streams from the network and allows saving of the received media data to a data storage. The program saves the whole stream, but decodes only one video and one audio streams that are detected first.

NWPlayer demonstrates work of the following components:

- Elecard Network Source;
- Elecard MPEG Push Demultiplexer.

4.2.1.2 Running NWPlayer

To run NWPlayer, use the following command:

```
nwplayer [options...]
```

The following options are available:

Help Options	
-h, --help	Shows help options.
--help-all	Shows all available prompts.
--help-gst	Shows GStreamer parameters.
Application Options	
-u, --url=<udp | rtp>://multicast_group:port	Specifies input URL.
-i, --interface=xxx.xxx.xxx.xxx	Specifies network interface that is used for data receiving. The option is useful, if several network adapters are installed in the system. Optional. Default value for Linux OS: IP interface of 'eth0' or 'enp0s3' network. Default value for Windows OS: IP interface found first.
-s, --server=xxx.xxx.xxx.xxx	Specifies address of server that is used for data input. Optional. Default value: "0.0.0.0".
-d, --dump=FILE	Specifies a dump file location. Optional. Default value: NULL (not dump).
-v, --version	Shows the program version. The program is closed after the version displaying.

Examples:

```
nwplayer --help
nwplayer --version
nwplayer --url=udp://234.5.5.5:10202
nwplayer --url=udp://234.5.5.5:10202 --dump=dump.mpg
```

When the program is run, information about created components, component connections, and changes of pipeline states is displayed.

To quit the program, use the CTRL+C key combination.

4.2.1.3 Viewing Pipeline in DOT File

NWPlayer allows you to view a pipeline in a .dot file which is created automatically after running the

application. The Graphviz package must be installed on your computer to use the function.

To install the Graphviz package on CentOS, use the following command:

```
sudo yum install graphviz
```

To install the Graphviz package on Windows, download installer from site, run it and follow the onscreen instructions.

To enable the function for application on Linux systems, use the following command:

```
GST_DEBUG_DUMP_DOT_DIR=<dot file directory> nwplayer [options...]
```

To enable the function for application on Windows, add GST_DEBUG_DUMP_DOT_DIR to the list of environment variables, specify path to DOT file directory and run the application as usual.

The specified directory must exist.

To view a pipeline in the created .dot file, use the following command:

```
display <dot-file>
```

To convert the .dot file into an image, use the following command:

```
dot -Tpng pipeline.dot > pipeline.png
```

4.2.1.4 Path

Source (Elecard GStreamer Codec SDK root)\src\samples\network\nwplayer

Binaries (Elecard GStreamer Codec SDK root)\bin\nwplayer

4.2.1.5 Features

The NWPlayer application supports the following features:

- Playback of MPEG-2 TS streams from a network;
- Saving received media data to a disk;
- Decoding of AVC, HEVC and MPEG-2 video formats using Elecard components;
- Decoding of MPEG and AAC audio formats using Elecard components.

4.2.2 NWServer

4.2.2.1 Description

NWServer is a sample application that demonstrates broadcasting MPEG TS media data to the network. It is capable of sending RTP, UDP packets. This sample illustrates use of the Elecard TimeMarker and Elecard Network Sink plugins.

4.2.2.2 Running NWServer

To run **NWServer**, use the following command:

```
nwserver [options...]
```

The following options are available:

Help Options	
-h, --help	Shows help options.
--help-all	Shows all available prompts.

--help-gst	Shows GStreamer parameters.
Application Options	
-f, --file=FILE	Specifies location of the file to broadcast.
-u, --url=<udp | rtp>://multicast_group:port	Specifies output URL.
-i, --interface=xxx.xxx.xxx.xxx	Specifies network interface that is used for data broadcasting. The option is useful, if several network adapters are installed in the system. Optional. Default value for Linux OS: IP interface of 'eth0' or 'enp0s3' network. Default value for Windows OS: IP interface found first.
-l, --loop	Looped playback. Optional.
-v, --version	Shows the program version. The program is closed after the version displaying.

Examples:

```
nwserver --help

nwserver --version

nwserver --file=Film.mpg --url=udp://234.5.5.5:10202
```

When the program is run, information about created components, program settings, and changes of pipeline states is displayed.

To quit the program, use the CTRL+C key combination.

4.2.2.3 Viewing Pipeline in DOT File

NWServer allows you to view a pipeline in a .dot file which is created automatically after running the application. The Graphviz package must be installed on your computer to use the function.

To install the Graphviz package on CentOS, use the following command:

```
sudo yum install graphviz
```

To install the Graphviz package on Windows, download installer from site, run it and follow the onscreen instructions.

To enable the function for application on Linux systems, use the following command:

```
GST_DEBUG_DUMP_DOT_DIR=<dot file directory> nwserver [options...]
```

To enable the function for application on Windows, add GST_DEBUG_DUMP_DOT_DIR to the list of environment variables, specify path to DOT file directory and run the application as usual.

The specified directory must exist.

To view a pipeline in the created .dot file, use the following command:

```
display <dot-file>
```

To convert the .dot file into an image, use the following command:

```
dot -Tpng pipeline.dot > pipeline.png
```

4.2.2.4 Path

Source (Elecard GStreamer Codec SDK root)\src\samples\network\nwserver

Binaries (Elecard GStreamer Codec SDK root)\bin\nwserver

4.2.2.5 Features

The NWServer sample application supports the following features:

- Broadcasting MPEG TS media streams without transcoding;
- Broadcasting data with use of specified network interface;
- Demonstrating work of the Elecard TimeMarker component.

4.2.3 NWServerTranscoder

4.2.3.1 Description

NWServerTranscoder is a sample application that demonstrates transcoding, multiplexing and broadcasting media data to the network. It is capable of sending RTP, UDP packets. This sample illustrates use of Elecard AVC Video Encoder and Elecard MPEG-2 Video Encoder, Elecard AAC Audio Encoder and Elecard MPEG Audio Encoder, Elecard MPEG Multiplexer and Elecard Network Sink plugins.

The NWServerTranscoder sample application supports the following output formats:

- Multiplexing formats – MPEG-2 TS;
- Video formats – AVC, MPEG-2;
- Audio formats – AAC, MPEG.

4.2.3.2 Running NWServerTranscoder

To run NWServerTranscoder, use the following command:

```
nwservertranscoder [options...]
```

The following options are available:

Help Options	
-h, --help	Shows help options.
--help-all	Shows all available prompts.
--help-gst	Shows GStreamer parameters.
Application Options	
-f, --file=*FILE*	Specifies location of the file to broadcast.
-u, --url=<udp | rtp>://multicast_group:port	Specifies output URL.
-i, --interface=xxx.xxx.xxx.xxx	Specifies network interface that is used for data

	broadcasting. The option is useful, if several network adapters are installed in the system. Optional. Default value for Linux OS: IP interface of 'eth0' or 'enp0s3' network. Default value for Windows OS: IP interface found first.
`-l, --loop`	Looped playback. Optional.
`--vcodec=`*`ENCODER`*	Specifies the name of the Elecard video encoder component. Optional. Available values: eavcenc (default), em2venc.
`--vbitrate-mode=`*`MODE`*	Specifies video bitrate mode. Optional. Available values: vbr, cbr (default).
`--vbitrate-avg=`*`VALUE`*	Specifies average video bitrate in kbit/sec. Optional.
`--vbitrate-max=`*`VALUE`*	Specifies the maximum video bitrate for Variable Bitrate Mode in kbit/sec. Optional.
`--acodec=`*`ENCODER`*	Specifies the name of the Elecard audio encoder component. Optional. Available values: eaacenc (default), empegaudioenc.
`--aencbitrate=`*`VALUE`*	Specifies the audio bitrate in kbit/sec. Optional.
`-v, --version`	Shows the program version. The program is closed after the version displaying. Optional.

Examples:

```
nwservertranscoder --help

nwservertranscoder --version

nwservertranscoder --file=Film.mpg --url=udp://234.5.5.5:10202
```

```
nwservertranscoder --file=Film.mpg --url=udp://234.5.5.5:10202 --
vcodec=em2venc --vbitrate-mode=vbr --vbitrate-avg=3000 --vbitrate-max=6000
```

When the program is run, information about created components, program settings, and changes of pipeline states is displayed.

To quit the program, use the CTRL+C key combination.

4.2.3.3 Viewing Pipeline in DOT File

NWServerTranscoder allows you to view a pipeline in a .dot file which is created automatically after running the application. The Graphviz package must be installed on your computer to use the function.

To install the Graphviz package on CentOS, use the following command:

```
sudo yum install graphviz
```

To install the Graphviz package on Windows, download installer from site, run it and follow the onscreen instructions.

To enable the function for application on Linux systems, use the following command:

```
GST_DEBUG_DUMP_DOT_DIR=<dot file directory> nwservertranscoder [options...]
```

To enable the function for application on Windows, add GST_DEBUG_DUMP_DOT_DIR to the list of environment variables, specify path to DOT file directory and run the application as usual.

The specified directory must exist.

To view a pipeline in the created .dot file, use the following command:

```
display <dot-file>
```

To convert the .dot file into an image, use the following command:

```
dot -Tpng pipeline.dot > pipeline.png
```

4.2.3.4 Path

Source (Elecard GStreamer Codec SDK root)\src\samples\network\nwservertranscoder

Binaries (Elecard GStreamer Codec SDK root)\bin\nwservertranscoder

4.2.3.5 Features

The NWServerTranscoder sample application supports the following features:

- Transcoding and multiplexing media streams;
- Broadcasting (multicast or unicast) media streams over the Internet/Intranet;
- Decoding and encoding video using Elecard components;
- Decoding and encoding audio using Elecard components.

4.2.4 SimplePlayer

4.2.4.1 Description

SimplePlayer is a console sample application that plays local media files. The sample decodes only one video and one audio streams that are detected first. For MPEG TS/PS files, decoded streams has the same program number.

SimplePlayer demonstrates work of the following components:

- Elecard AVC Video Decoder;
- Elecard HEVC Video Decoder;
- Elecard MPEG Audio Decoder;

- Elecard AAC Audio Decoder;
- Elecard MPEG Demultiplexer;
- Elecard MP4 Demultiplexer;
- Elecard MKV Demultiplexer;
- Elecard MXF Demultiplexer;
- Elecard Deinterlacer.

4.2.4.2 Running SimplePlayer

To run SimplePlayer, use the following command:

```
simpleplayer [options...]
```

The following options are available:

Help Options	
`-h, --help`	Shows help options.
`--help-all`	Shows all available prompts.
`--help-gst`	Shows GStreamer parameters.
Application Options	
`-f, --file=FILE`	Specifies location of the file to broadcast.
`--deinterlace-mode=MODE`	Deinterlace mode. Optional. Available values: auto (default), all, none.
`--deinterlace-method=METHOD`	Deinterlace method. Optional. Available values: better (default), faster.
`-v, --version`	Shows the program version. The program is closed after the version displaying.

Examples:

```
simpleplayer --help

simpleplayer --version

simpleplayer --file=Video.mpg

simpleplayer --file=Video.mpg --deinterlace-mode=all --deinterlace-method=faster
```

When the program is run, information about created components, component connections, and changes of pipeline states is displayed.

Elecard Deinterlacer component converts interlaced video into progressive one. This component can be configured by using optional parameters: deinterlace-mode and deinterlace-method. If a player has been started without deinterlace-mode, the Elecard Deinterlacer component is not included in a pipeline, and the pipeline is similar to the one following below:

```
filesrc location=videofile.mpg ! some_elecard_demuxer ! some_elecard_decoder ! edif ! videoconvert ! videoscale ! autovideosink
```

Three deinterlace modes are available:

- auto – only interlaced frames are deinterlaced;
- all – every frame is deinterlaced;
- none – no deinterlacing is performed.

The *deinterlace-method* parameter specifies user's preference:

- better – deinterlacing with better output quality is preferred (data processing speed can be low);
- faster – deinterlacing with higher speed is preferred (output quality can be low).

It is possible to change the current position during playback. Values of a new position is set depending on the file duration: from 0 up to 100, where 0 corresponds to the file beginning, 100 corresponds to the file end. If 100 is entered the program quits.

To quit the program, use the CTRL+C key combination.

4.2.4.3 Viewing Pipeline in DOT File

SimplePlayer allows you to view a pipeline in a .dot file which is created automatically after running the application. The Graphviz package must be installed on your computer to use the function.

To install the Graphviz package on CentOS, use the following command:

```
sudo yum install graphviz
```

To install the Graphviz package on Windows, download installer from site, run it and follow the onscreen instructions.

To enable the function for application on Linux systems, use the following command:

```
GST_DEBUG_DUMP_DOT_DIR=<dot file directory> simpleplayer [options...]
```

To enable the function for application on Windows, add GST_DEBUG_DUMP_DOT_DIR to the list of environment variables, specify path to DOT file directory and run the application as usual.

The specified directory must exist.

To view a pipeline in the created .dot file, use the following command:

```
display <dot-file>
```

To convert the .dot file into an image, use the following command:

```
dot -Tpng pipeline.dot > pipeline.png
```

4.2.4.4 Path

Source	(Elecard GStreamer Codec SDK root)\src\samples\decoders\simpleplayer
Binaries	(Elecard GStreamer Codec SDK root)\bin\simpleplayer

4.2.4.5 Features

The SimplePlayer application supports the following features:

- Playback of MPEG-2, MP4, MKV, MXF streams;
- Decoding of AVC, HEVC and MPEG-2 video formats using Elecard components;
- Decoding of MPEG and AAC audio formats using Elecard components;
- Positioning during playback;
- Deinterlacing interlaced video.

4.2.5 PlaybinSample

4.2.5.1 Description

PlaybinSample is a console sample application that demonstrates how to integrate Elecard components into playbin for decoding media streams from the network or files.

The following components can be used by playbin:

- Elecard AVC Video Decoder;
- Elecard HEVC Video Decoder;
- Elecard MPEG Audio Decoder;
- Elecard AAC Audio Decoder;
- Elecard MP4 Demultiplexer;
- Elecard MKV Demultiplexer;
- Elecard MXF Demultiplexer;
- Elecard MPEG Demultiplexer;
- Elecard MPEG Push Demultiplexer;
- Elecard Network Source;
- Elecard RTSP Source;
- Elecard HLS Source.

4.2.5.2 Running PlaybinSample

To run PlaybinSample, use the following command:

```
playbinsample [options...]
```

The following options are available:

Help options	
`-h, --help`	Shows help options.
`--help-all`	Shows all available prompts.
`--help-gst`	Shows GStreamer parameters.
Application Options	
`-u, --uri=URI`	Set the URI that playbin will play.
`-v, --version`	Shows the program version. The program is closed after the version displaying.

Examples:

```
playbinsample --help

playbinsample --version

playbinsample --uri=file:///Video.mpg

playbinsample --uri=udp://235.1.2.3:10200
```

When the program is run, information about created components and changes of pipeline states is displayed.

It is possible to change the current position during playback of files. Values of a new position is set depending on the file duration: from 0 up to 100, where 0 corresponds to the file beginning, 100 corresponds to the file end. If 100 is entered the program quits.

To quit the program, use the CTRL+C key combination.

4.2.5.3 Viewing Pipeline in DOT File

PlaybinSample allows viewing a pipeline in a .dot file which is created automatically after running the application. The Graphviz package must be installed on your computer to use the function.

To install the Graphviz package on CentOS, use the following command:

```
sudo yum install graphviz
```

To install the Graphviz package on Windows, download installer from site, run it and follow the onscreen instructions.

To enable the function for application on Linux systems, use the following command:

```
GST_DEBUG_DUMP_DOT_DIR=<dot file directory> playbinsample [options...]
```

To enable the function for application on Windows, add GST_DEBUG_DUMP_DOT_DIR to the list of environment variables, specify path to DOT file directory and run the application as usual.

The specified directory must exist.

To view a pipeline in the created .dot file, use the following command:

```
display <dot-file>
```

To convert the .dot file into an image, use the following command:

```
dot -Tpng pipeline.dot > pipeline.png
```

4.2.5.4 Path

Source (Elecard GStreamer Codec SDK root)\src\samples\decoders\playbinsample

Binaries (Elecard GStreamer Codec SDK root)\bin\playbinsample

4.2.5.5 Features

The PlaybinSample application supports the following features:

- Receiving data over UDP, RTP, RTSP, HLS;
- Playing back of MPEG-2, MP4, MKV, MXF files;
- Decoding of AVC, HEVC and MPEG-2 video formats;
- Decoding of MPEG and AAC audio formats;
- Positioning during playback (if it possible).