i.MX 6 Series Ubuntu Multimedia User's Guide

Document Number: IMX6UBUNTUUG
Rev L3.0.35_4.1.0, 09/2013

freescale™

Contents

Section number **Title** **Page**

Chapter 1
About This Book

Chapter 2
Installing and Building Plug-Ins

Chapter 3
Testing Installation

Chapter 4
Multi-Overlay Support

Chapter 5
Streaming Support

Chapter 1
About This Book

This document describes the package contents and provides instructions for building the libraries that are based on the GStreamer architecture. GStreamer is a powerful, versatile framework for creating streaming media applications.

1.1 Audience

This document is intended for software, hardware, and system engineers who are planning to use multimedia codecs with GStreamer architecture and for anyone who wants to understand more about multimedia codecs. The document assumes that the user has a basic understanding of GStreamer and LTIB architecture.

1.2 Organization

This document consists of the following chapters:

- Chapter 1: Provides an introduction of the document.
- Chapter 2: Identifies the system requirements for installing and building plug-ins, and explains how to build multimedia components using LTIB or install multimedia components on Ubuntu operating system.
- Chapter 3: Explains how to test and use multimedia codecs.
- Chapter 4: Describes how to add multi-overlay support for video playback.
- Chapter 5: Describes how to add streaming support.

1.3 Conventions

This document uses the following conventions:

- Courier New font: This font is used to identify commands, explicit command parameters, code examples, expressions, data types, and directives.
- $ sign: It is used to specify replaceable command parameters.

1.4 References

The following documents were referenced to build this document:

- i.MX 6Dual/6Quad SABRE-SD Linux User's Guide
- i.MX 6 Series Ubuntu Multimedia Release Notes
- GStreamer Command-Line Player Application Specification
- i.MX Advanced Toolkit Standard Version User's Guide

1.5 Definitions, Acronyms, and Abbreviations

Following is a list of abbreviations used in this document:

- FSL: Freescale
- GUI: Graphic user interface
- SoC: System-on-chip
- Codec: Coder/decoder
- HTTP: Hypertext Transfer Protocol
- LTIB: Linux Target Image Builder
- ARM: Advanced RISC Machine
- ASRC: Asynchronous Sample Rate Converter
- APT: Advanced Packaging Tool
- Gst: GStreamer (open source multimedia framework)
- gplay: Freescale command-line player with GStreamer backend

Chapter 2
Installing and Building Plug-Ins

This chapter describes how to build/install Freescale multimedia core libraries and Freescale GStreamer plug-ins. Freescale multimedia core libraries are released only in binary format. Freescale GStreamer plug-ins contain source code.

Freescale provides the following two types of release packages:

- LTIB packages: These packages are for Freescale core libraries and GStreamer plug-ins. They include Freescale multimedia core binary libraries and Freescale GStreamer plug-in source code.
- Debian packages: These packages are for Ubuntu system. Debian binary packages are used to install Freescale core libraries and GStreamer plug-in binaries on an i.MX 6 series board running Ubuntu operating system. Debian source packages are used to build Freescale GStreamer plug-ins on an i.MX 6 series board.

2.1 Building Plug-Ins by Using LTIB

2.1.1 System Requirements for Installing LTIB

- Hardware requirements:
  - i.MX 6Dual/Quad/6Solo/6DualLite/6SoloLite SABRE-SD/SABRE-AI/EVK board
- Software requirements:
  - Board Support Package (BSP):
    - i.MX Linux BSP version L3.0.35_4.1.0 or above
  - GStreamer:
    - gstreamer (version 0.10.35)
    - gstreamer-plugins-base (version 0.10.35)
    - gstreamer-plugins-good (version 0.10.30)

NOTE

Freescale GStreamer plug-ins are dependent on the GStreamer framework, including the GStreamer core, gstreamer-plugins-base, and gstreamer-plugins-good.

2.1.2 Building Plug-Ins

To implement the procedures given in this section, use a computer (x86) running a Linux operating system.

To install LTIB and extract the package files, follow these steps:

1. Install LTIB on your computer.

```
./<ltib_release>/install
```

NOTE

To install LTIB, see i.MX Linux User's Guide of the target platform.

2. Obtain the following packages included in the release:
   - gst-fsl-plugins-$VERSION.tar.gz: It is a GStreamer plug-in source package that contains source code for the multimedia GStreamer-based plug-in for the i.MX application processor.
   - libfslcodec-$VERSION.tar.gz: It is a codec binary package that contains Freescale multimedia core codec libraries for the i.MX application processor.
   - libfslparser-$VERSION.tar.gz: It is a parser binary package that contains Freescale multimedia core parser libraries for the i.MX application processor.
   - libfslvpuwrap-$VERSION.tar.gz: It is a VPU-wrap source package.
   - gst-plugins-gl-[version].tar.gz: It is a GL plug-in source package.
   - fsl-alsa-plugins-[version].tar.gz: It is an ALSA plug-in source package.

NOTE

The libfslvpuwrap, gst-plugins-gl, and fsl-alsa-plugins packages are only available on i.MX 6Dual/6Quad/6DualLite platform. For details, see "Release Contents" section of i.MX 6 Series Ubuntu Multimedia Release Notes.

3. Copy these standard packages to the LPP directory, which, by default, is set to /var/tmp/pkgs (see litb/.ltibrc %ldirs).

NOTE

If you are installing LTIB for the first time, you need to create this directory manually.

To build the package, follow these steps:

1. Select the platform.

NOTE

For details, see "Building the Linux Platform" chapter of *i.MX 6Dual/6Quad SABRE-SD Linux User Guide* and *i.MX 6Solo/6DualLite SABRE-SD Linux User Guide*.

2. Select Package List --> Freescale Multimedia Plugins/Codecs, as shown in the below figure.

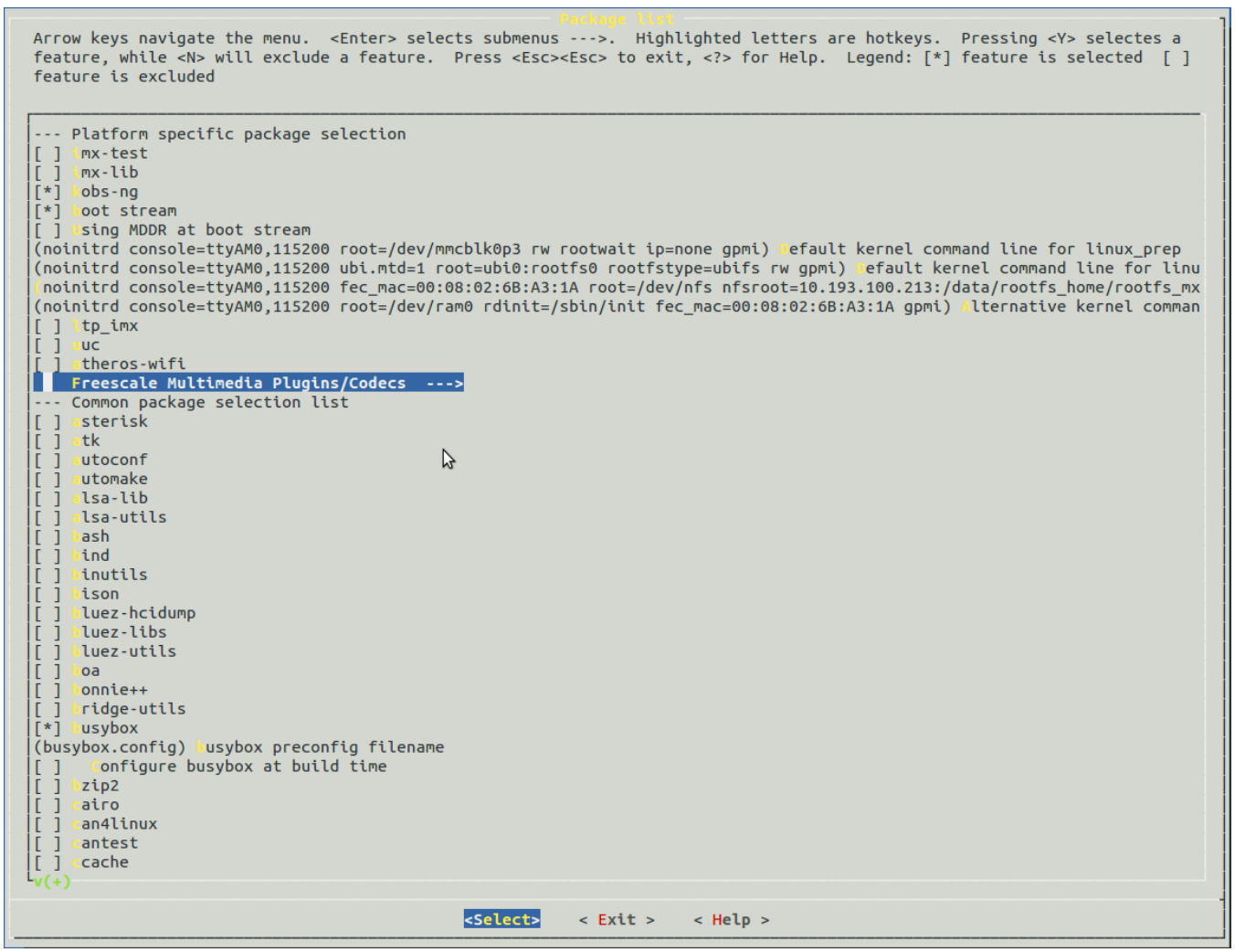

Figure 2-1. LTIB Package Selection Menu

3. Select libfslcodec, libfslparser, libfslvpuwrap, gst-fsl-plugins, gst-plugins-gl, and fsl-alsa-plugins, as shown in the below figure.

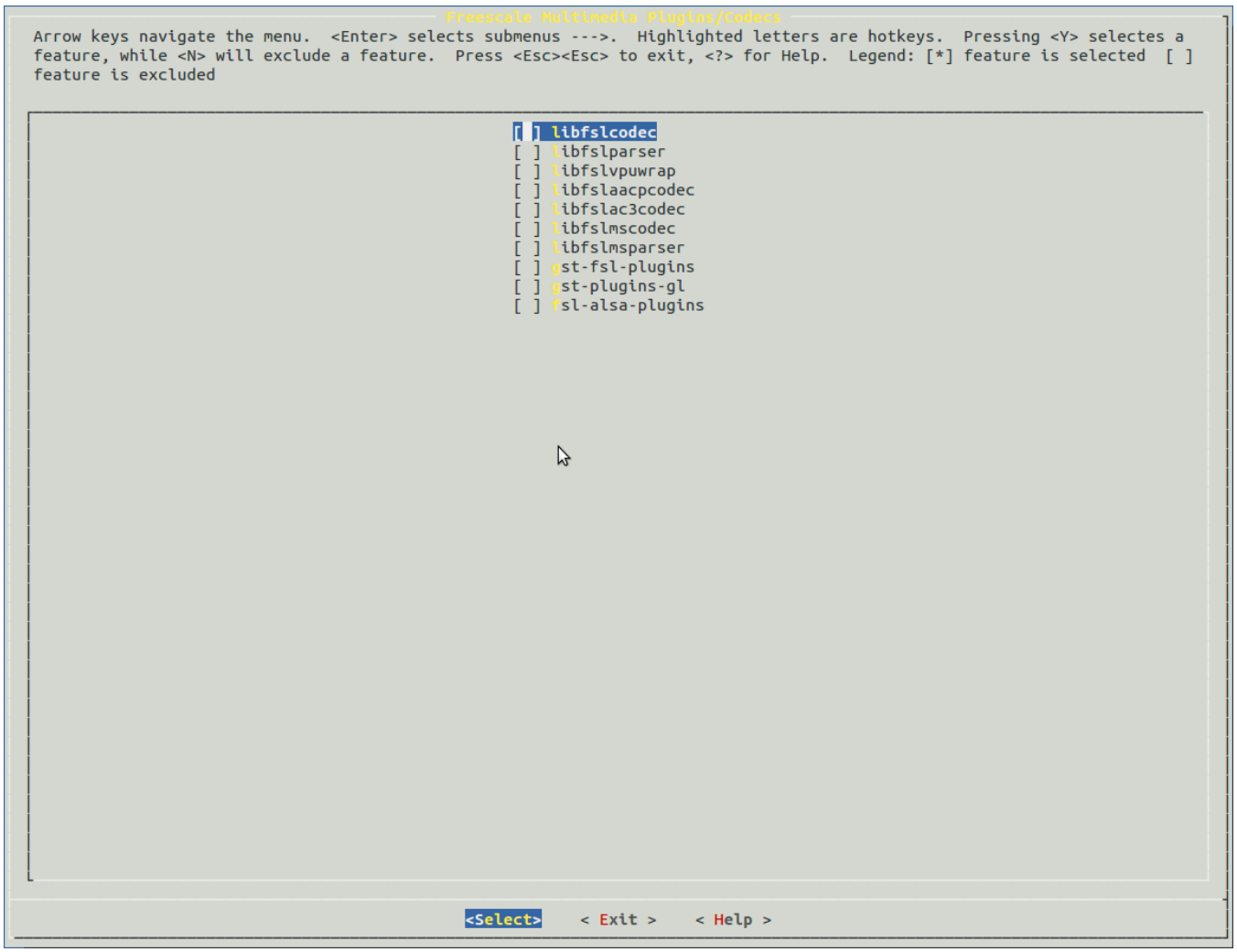

Figure 2-2. Selecting Plug-Ins

4. Select gstreamer-plugins-good package, as shown in the below figure.

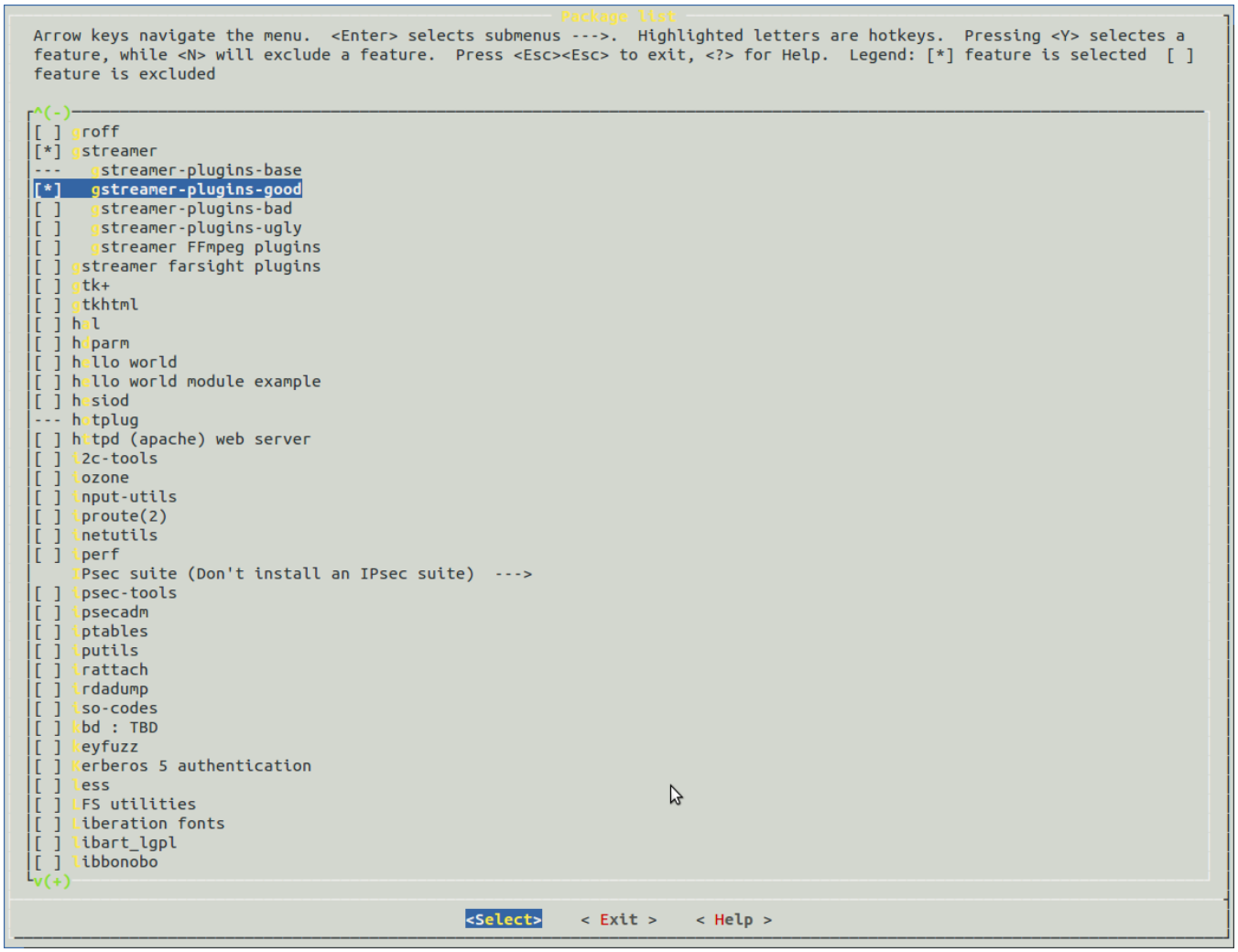

Figure 2-3. Selecting gstreamer-plugins-good Package

5. Follow the LTIB compilation instructions.

After a successful build, uImage and rootfs will be created. The rootfs is located in LTIB directory. It includes Freescale multimedia Linux codecs and Freescale GStreamer plug-ins. The uImage is located in rootfs/boot in LTIB build directory.

NOTE

For detailed information on a platform, see i.MX Linux User's Guide for that platform.

2.2 Installing/Building Plug-Ins on Ubuntu

2.2.1 System Requirements for Ubuntu

- i.MX 6 series board runs Ubuntu operating system (11.10 Oneiric).

- imx-lib Debian package (version>=4.1.0).
- firmware-imx Debian package (version>=4.1.0).
- GStreamer:
  - gstreamer (version=0.10.35 on Ubuntu 11.10)
  - gstreamer-plugins-good (version=0.10.30 on Ubuntu 11.10)

2.2.2 Installing/Building Plug-Ins

This section describes how to build and generate Debian packages on the i.MX 6 series board natively.

Following steps illustrate how to install Freescale multimedia plug-ins on Ubuntu and also how to build a Debian package from the source.

1. Prepare an i.MX 6 series board running Ubuntu operating system (11.10 Oneiric).
2. Install BSP support libraries package, which is released with the BSP. You need to make the .deb files available to the board or a USB key. Copying the files to the board in some other way is also fine.

```
sudo dpkg -i imx-lib-$VERSION-$RELEASE.deb
sudo dpkg -i kernel_$VERSION-imx_$RELEASE_armel.deb
```

NOTE

It is desired to install all other BSP Debian packages as well.

3. Obtain the Debian binary packages given in the below two tables from the multimedia release package. You can copy them to a USB key that will be connected to the board. The packages listed in the first table are repackaged to avoid any toolchain compatibility issue.

Table 2-1. BSP-Related Debian Packages

Package Name	Content	License
gpu-viv	gpu-viv-ubt0_[version]_armel.deb	Standard

Table 2-2. Multimedia-Related Debian Packages

Package Name	Content	License
FSL plugins	gstreamer0.10-plugins-fsl_$VERSION_armel.deb gstreamer0.10-plugins-fsl-dbg_ $VERSION_armel.deb	Standard

Table continues on the next page...

Table 2-2. Multimedia-Related Debian Packages (continued)

Package Name	Content	License
	gstreamer0.10-plugins-fsl-doc_$VERSION_all.deb	
FSL command media player	gstreamer0.10-plugins-fsl-tools_ $VERSION_armel.deb gstreamer0.10-plugins-fsl-tools-dbg_ $VERSION_armel.deb	Standard
FSL multiple audio core codecs	libfslaudiocodec1_$VERSION_armel.deb libfslaudiocodec-dev_$VERSION_armel.deb libfslaudiocodec-doc_$VERSION_all.deb libfslaudiocodec-test_$VERSION_armel.deb	Standard
FSL parser core libraries	libfslparser3_$VERSION_armel.deb libfslparser-dev_$VERSION_armel.deb libfslparser-doc_$VERSION_all.deb	Standard
FSL video and image core codecs	libfslvideocodec1_$VERSION_armel.deb libfslvideocodec-dev_$VERSION_armel.deb libfslvideocodec-doc_$VERSION_all.deb libfslvideocodec-test_$VERSION_armel.deb	Standard
FSL VPU wrapper library	libfslvpuwrap3_$VERSION_armel.deb libfslvpuwrap3-dbg_$VERSION_armel.deb libfslvpuwrap-dev_$VERSION_armel.deb libfslvpuwrap-doc_$VERSION_all.deb	Standard
FSL GST libraries	libgstreamer-plugins-fsl0.10-0_ $VERSION_armel.deb libgstreamer-plugins-fsl0.10-dbg_ $VERSION_armel.deb libgstreamer-plugins-fsl0.10-dev_ $VERSION_armel.deb	Standard
FSL GL plugin	libgstreamer-plugins-gl0.10_$VERSIONarmel.deb libgstreamer-plugins-gl0.10-dbg- $VERSION_armel.deb libgstreamer-plugins-gl0.10-dev_ $VERSION_armel.deb	Standard
FSL ALSA plugin	fsl-alsa-plugins_$VERSION_armel.deb fsl-alsa-plugins-dbg_$VERSION_armel.deb fsl-alsa-plugins-dev_$VERSION_armel.deb	Standard
AAC plus core decoder	libfslaacpaudiocodec1_$VERSION_armel.deb libfslaacpaudiocodec-dev_$VERSION_armel.deb	License restricted

Table continues on the next page...

Table 2-2. Multimedia-Related Debian Packages (continued)

Package Name	Content	License
	libfslaacpaudiocodec-doc_$VERSION_all.deb libfslaacpaudiocodec-test_$VERSION_armel.deb	
AC3 core decoder	libfslac3audiocodec1_$VERSION_armel.deb libfslac3audiocodec-dev_$VERSION_armel.deb libfslac3audiocodec-doc_$VERSION_all.deb libfslac3audiocodec-test_$VERSION_armel.deb	License restricted
Microsoft audio core codecs	libfslmsaudiocodec1_$VERSION_armel.deb libfslmsaudiocodec-dev_$VERSION_armel.deb libfslmsaudiocodec-doc_$VERSION_all.deb libfslmsaudiocodec-test_$VERSION_armel.deb	License restricted
Microsoft parser	libfslmsparser3_$VERSION_armel.deb libfslmsparser-dev_$VERSION_armel.deb libfslmsparser-doc_$VERSION_all.deb	License restricted
Microsoft video core codecs	libfslmsvideocodec1_$VERSION_armel.deb libfslmsvideocodec-dev_$VERSION_armel.deb libfslmsvideocodec-doc_$VERSION_all.deb libfslmsvideocodec-test_$VERSION_armel.deb	License restricted

Use the dpkg command to install these packages:

```
sudo dpkg -i *.deb
```

4. To build the codec and plug-in deb packages from source, install the below tools and dependencies:

```
sudo apt-get update
sudo apt-get install aptitude
sudo aptitude install dpkg-dev
sudo aptitude install devscripts
sudo aptitude install dh-autoreconf
```

For fundamental packages:

```
sudo aptitude install ntp cdbs dh-autoreconf gnupg-agent keychain pbuilder quilt
```

For GStreamer plug-ins:

```
sudo apt-get install libgstreamer0.10-dev libgstreamer-plugins-base0.10-dev gstreamer-
tools
```

For gst-plugins-gl:

```
sudo apt-get install mesa-common-dev libgtk2.0-dev libjpeg-dev libpng12-dev
```

For fsl-alsa-plugins:

```
sudo apt-get install libasound2-dev
```

It is required to install the dev Debian packages from the multimedia release package to solve the build dependency.

5. Create a directory, and copy all the source Debian packages given in the below table to this directory.

Table 2-3. Source Debian Packages

Package Name	Content	License
FSL plugins	gst-fsl-plugins_$VERSION.debian.tar.gz gst-fsl-plugins_$VERSION.dsc gst-fsl-plugins_$VERSION.orig.tar.gz	Standard
FSL core codecs	libfslcodec_$VERSION.debian.tar.gz libfslcodec_$VERSION.dsc libfslcodec_$VERSION.orig.tar.gz	Standard
FSL core parsers	libfslparser_$VERSION.debian.tar.gz libfslparser_$VERSION.dsc libfslparser_$VERSION.orig.tar.gz	Standard
FSL VPU wrapper	libfslvpuwrap_$VERSION.debian.tar.gz libfslvpuwrap_$VERSION.dsc libfslvpuwrap_$VERSION.orig.tar.gz	Standard
FSL GL plugin	gst-plugins-gl_$VERSION.debian.tar.gz gst-plugins-gl_$VERSION.dsc gst-plugins_gl_$VERSION.orig.tar.gz	Standard
FSL ALSA plugin	sl-alsa-plugins_$VERSION.debian.tar.gz fsl-alsa-plugins_$VERSION.dsc fsl-alsa-plugins_$VERSION.orig.tar.gz	Standard
AAC plus core codec	libfslaacpcodec_$VERSION.debian.tar.gz libfslaacpcodec_$VERSION.dsc libfslaacpcodec_$VERSION.orig.tar.gz	License restricted
AC3 core codec	libfslac3codec_$VERSION.debian.tar.gz libfslac3codec_$VERSION.dsc libfslac3codec_$VERSION.orig.tar.gz	License restricted
Microsoft core codecs	libfslmscodec_$VERSION.debian.tar.gz libfslmscodec_$VERSION.dsc	License restricted

Table continues on the next page...

Table 2-3. Source Debian Packages (continued)

Package Name	Content	License
	libfslmscodec_$VERSION.orig.tar.gz	
Microsoft core parsers	libfslmsparser_$VERSION.debian.tar.gz libfslmsparser_$VERSION.dsc libfslmsparser_$VERSION.orig.tar.gz	License restricted

6. Run the following command to build each of the above packages:

```
dpkg-source -x <file.dsc>
cd <newly_generated_directory>
debuild -i -uc -us
```

The Debian binaries will be created at the above directory. You can install them by using the following command:

```
dpkg -i <package.deb>
```

Chapter 3
Testing Installation

This chapter explains how to check and test the multimedia codecs (audio decoder, audio encoder, video decoder, and video encoder). It also explains how to enable the post-process filter to the pipeline to be created in the GStreamer architecture.

NOTE

Each platform provides a certain set of codecs. See i.MX 6 Series Ubuntu Multimedia Release Notes to determine which codecs are included in the BSP release package.

3.1 Testing Multimedia Environment Setting

If the test rootfs is Ubuntu, some audio/video input/output may need to be configured before testing. In below description, it is assumed that pulseaudio is installed.

3.1.1 Audio Output Setting

Use "pactl" command to list all available audio sinks:

```
pactl list sinks
```

A list of available audio sinks will be displayed as shown below:

```
Sink #0
        State: SUSPENDED
        Name: alsa_output.platform-soc-audio.1.analog-stereo
        Description: sgtl5000-audio Analog Stereo

          ...
           ...
Sink #1
        State: SUSPENDED
        Name: alsa_output.platform-soc-audio.4.analog-stereo
        Description: imx-hdmi-soc Analog Stereo
          ...
          ...
```

Use "pacmd" command to set the default audio sink according to the sink number given above:

```
pacmd set-default-sink $sink-number   (e.g. $sink-number could be 0 or 1 in above list)
```

After setting the default sink, use below command to verify the audio path:

```
gst-launch audiotestsrc ! pulsesink
```

NOTE

The pulseaudio is only available for Ubuntu rootfs. For LTIB environment HDMI output, use "aplay -l" to check the audio device. Use "gst-launch playbin2 uri=<stream> audio-sink="alsasink device=plughw:$audio_device_number"" to output to the specific device.

3.1.2 Audio Input Setting

Use "pactl" command to list all available audio sources:

```
pactl list sources
```

A list of available audio sources will be displayed as shown below:

```
Source #0
        State: SUSPENDED
        Name: alsa_output.platform-soc-audio.1.analog-stereo.monitor
        Description: Monitor of sgtl5000-audio Analog Stereo
          ...
          ...
Source #1
        State: SUSPENDED
        Name: alsa_input.platform-soc-audio.1.analog-stereo
        Description: sgtl5000-audio Analog Stereo                                    ...
          ...
          ...
```

Use "pacmd" command to set default audio source according to the source number given above:

```
pacmd set-default-source $source-number (e.g. $source-number could be 0 or 1 in above list)
```

NOTE

If record and playback are not needed at the same time, there is no need to set to monitor mode.

Use "pactl" command to see the current status of audio input/output path setting:

```
pactl stat
```

A list will be displayed as given below:

```
Currently in use: 1 blocks containing 64.0 KiB bytes total.
Allocated during whole lifetime: 2931 blocks containing 5.3 MiB bytes total.
```

```
Sample cache size: 0 B
Server String: /home/linaro/.pulse/82aa6303a555980d8320686e000e1e89-runtime/native
Library Protocol Version: 24
Server Protocol Version: 24
Is Local: yes
Client Index: 11
Tile Size: 65496
User Name: linaro
Host Name: linaro-ubuntu-desktop
Server Name: pulseaudio
Server Version: 1.0
Default Sample Specification: s16le 2ch 44100Hz
Default Channel Map: front-left,front-right
Default Sink: alsa_output.platform-soc-audio.1.analog-stereo
Default Source: alsa_input.platform-soc-audio.1.analog-stereo
Cookie: e9a5:dcd9
```

3.1.3 Video Setting

See i.MX Linux User's Guide to know how to set boot cmd in U-Boot for different video output (HDMI, LVDS, and so on).

3.2 Testing Codecs with GStreamer

GStreamer provides two useful applications for testing multimedia codecs: gst-inspect and gst-launch.

3.2.1 gst-inspect Tool

The gst-inspect tool can provide information about an available GStreamer plug-in, a particular plug-in, or a particular element.

To view the list of installed Freescale multimedia codec plug-ins, use the following command:

```
gst-inspect | grep imx
```

A list is displayed as shown below:

```
wmadec.imx:  mfw_wma10decoder: wma audio decoder
aiur.imx:  aiurdemux: aiur universal demuxer
v4lsink.imx:  mfw_v4lsink: v4l2 video sink
amrdec.imx:  mfw_amrdecoder: amr audio decoder
beep.imx:  beepdec: beep audio decoder
wmvdec.imx:  mfw_wmvdecoder: wmv video decoder
aacpdec.imx:  mfw_aacplusdecoder: aac plus audio decoder
mp3enc.imx:  mfw_mp3encoder: mp3 audio encoder
v4lsrc.imx:  mfw_v4lsrc: v4l2 based src
mp3dec.imx:  mfw_mp3decoder: mp3 audio decoder
isink.imx:  mfw_isink: IPU-based video sink
adownmix.imx:  mfw_downmixer: audio down mixer
vorbisdec.imx:  mfw_vorbisdecoder: vorbis audio decoder
```

```
wma8enc.imx:  mfw_wma8encoder: wma8 audio encoder
aacdec.imx:  mfw_aacdecoder: aac audio decoder
audiopeq.imx:  mfw_audio_pp: audio post equalizer
vpu.imx:  vpudec: VPU-based video decoder
```

The elements contained in this list may be different depending on the target platform.

For example:

```
vpu.imx:  vpudec: VPU-based video decoder
```

Here, "vpu.imx" is plug-in name, "vpudec" is element name, and "VPU-based video decoder" is long name of the element.

Use the following gst-inspect command to view the detailed information of an element:

```
gst-inspect $ELEMENT_NAME
```

For example, use the below command to display the detailed information of the vpudec element:

```
gst-inspect vpudec
```

All these plug-ins can be classified into audio decoder/encoder, video decoder/encoder, demuxer, and so on (as shown in the below table).

Table 3-1. Freescale Multimedia Codec Plug-Ins

audio_decoder_plugin	General codecs: • beepdec: Beep-unified audio decoder plug-in • mfw_amrdecoder: amr audio decoder plug-in • mfw_aacdecoder: aac audio decoder plug-in • mfw_mp3decoder: mp3 audio decoder plug-in • mfw_vorbisdecoder: vorbis audio decoder plug-in License limited codecs: • mfw_wma10decoder: wma audio decoder plug-in • mfw_aacplusdecoder: aac plus audio decoder plug-in • mfw_ac3decoder: ac3 audio decoder plug-in
audio_encoder_plugin	mfw_mp3encoder: mp3 audio encoder plug-in mfw_wma8encoder: wma8 audio encoder plug-in
video_decoder_plugin	vpudec: VPU-based video decoder plug-in mfw_wmvdecoder: Software wmv789 video decoder plug-in
video_encoder_plugin	vpuenc: VPU-based video encoder plug-in
demuxer_plugin	aiurdemux: aiur universal demuxer plug-in supporting General demuxer: AVI, MKV, MP4, MPEG2, OGG, FLV, WebM License limited demuxer: ASF
video_sink_plugin	mfw_v4lsink: v4l2 video sink plug-in mfw_isink: IPU device based video sink plug-in
camera_src_plugin	mfw_v4lsrc: v4l2 based embedded camera src plug-in

NOTE

The supported plug-ins may vary in different platforms. See i.MX 6 Series Ubuntu Multimedia Release Notes for details.

3.2.2 gst-launch Tool

The gst-launch tool builds and runs the basic GStreamer pipeline without trick mode support.

3.2.2.1 Playback with playbin2

Freescale recommends using GStreamer playbin2 plug-ins to play back audio or/and video. Playbin2 is self-constructed pipeline element, which auto connects all necessary elements to decode a media file/resource, including source, parser, decoder, sink, and so on. It uses the below command:

```
gst-launch playbin2 uri=$URI
```

The $URI is Universal Resource Identifier. For a local file, URI starts with file:///. For example, to play a local file, test.avi, located in /media directory, use:

```
gst-launch playbin2 uri=file:///media/test.avi
```

NOTE

To make playbin2 compatible with Freescale multimedia GStreamer plug-ins, a Freescale optimized gstreamer-plugins-base package needs to be installed. This package is released with LTIB package or provided with multimedia Debian packages.

3.2.2.2 Audio Playback

Use the beep-unified audio decoder plug-in to test MP3/AAC/WMA/Vorbis/AC3 audio playback as follows:

```
gst-launch filesrc location=[clip_name] typefind=true ! $audio_decoder_plugin ! alsasink
```

For example:

```
gst-launch filesrc location=test.mp3 typefind=true ! [id3dmeux !]  beepdec ! audioconvert !
'audio/x-raw-int, channels=2' ! alsasink
```

[] indicates that the element inside it is optional.

Some mp3 may have id3 headers, which require you to have id3demux to parse them.

To test the WAV audio playback, use the following command:

```
gst-launch filesrc location=test.wav ! wavparse ! alsasink
```

NOTE

Before performing this test, you need to install the gstreamer-plugins-good package. Due to hardware and open source element limitation, for some combined configurations of specific channels and sample rates, the sound may not be heard.

3.2.2.3 Video only Playback

To create a video-only pipeline with the gst-launch tool, use the following command:

```
gst-launch filesrc location= test.video typefind=true ! $demuxer_plugin ! queue max-size-
time=0
! $video_decoder_plugin ! $video_sink_plugin
```

For example, for an ASF (WMV only) file playback, use these commands:

```
gst-launch filesrc location=test.asf typefind=true ! aiurdemux ! queue max-size-time=0 !
mfw_wmvdecoder ! mfw_v4lsink
gst-launch filesrc location=test.asf typefind=true ! aiurdemux ! queue max-size-time=0 !
vpudec
! mfw_v4lsink
```

3.2.2.4 Audio/Video File Playback

To create an audio/video combined pipeline with the gst-launch tool, use these commands:

```
gst-launch filesrc location=test_file typefind=true ! $demuxer_plugin name=demux demux. !
queue max-size-buffers=0 max-size-time=0 ! $video_decoder_plugin ! $video_sink_plugin
demux. !
queue max-size-buffers=0 max-size-time=0 ! $audio_decoder_plugin ! audioconvert !
'audio/x-raw-int, channels=2' ! alsasink
```

In VPU mode, change video_decoder_plugin to vpudec. The VPU mode is only used for Freescale i.MX SoCs with embedded VPU.

The max-size-time in Queue element should be set because the playback could be not smooth with default value, 1 second.

For example, for AVI (H264+MP3) video playback:

On SoC without VPU, such as i.MX 6SoloLite:

```
gst-launch filesrc location=test.avi typefind=true ! aiurdemux name=demux demux. ! queue
max-size-buffers=0 max-size-time=0 ! mfw_h264decoder ! mfw_v4lsink demux. ! queue
max-size-buffers=0 max-size-time=0 ! beepdec ! audioconvert ! 'audio/x-raw-int,
channels=2' !
alsasink
```

On SoC with VPU, such as i.MX 6Solo/6DualLite and i.MX 6Dual/6Quad:

```
gst-launch filesrc location=test.avi typefind=true ! aiurdemux name=demux demux. ! queue
max-size-buffers=0 max-size-time=0 ! vpudec ! mfw_v4lsink demux. ! queue max-size-buffers=0
max-size-time=0 ! beepdec ! audioconvert ! 'audio/x-raw-int, channels=2' ! alsasink
```

NOTE

The VPU decoder is currently available only for Freescale i.MX SoC with embedded VPU.

3.2.2.5 Audio Encoder Record

This release provides two audio encoders: MP3 and WMA8. Both may be enabled.

MP3 Encoder Record:

Encoding from a file:

```
$gst-launch filesrc location=test.wav ! wavparse ! mfw_mp3encoder ! filesink
location=output.mp3
```

Recording:

```
$gst-launch alsasrc num-buffers=$NUMBER blocksize=$SIZE ! mfw_mp3encoder ! filesink
location=output.mp3
```

The time duration of recording equals $NUMBER*$SIZE*8/ (samplerate*channel*bitwidth).

For example, to record 60 seconds of stereo channel sample with 44.1K sample rate and 16bit width, use the following command:

```
$gst-launch alsasrc num-buffers=240 blocksize=44100 ! mfw_mp3encoder ! filesink
location=output.mp3
```

To verify if the MP3 output is correct, use the beepdec:

```
$gst-launch filesrc location=output.mp3 typefind=true ! beepdec ! audioconvert !
'audio/x-raw-int, channels=2' ! alsasink
```

WMA8 Encoder Record:

Encoding from a file:

```
$gst-launch filesrc location=test.wav ! wavparse ! mfw_wma8encoder ! filesink
location=output.wma
```

Recording:

```
$gst-launch alsasrc num-buffers=$NUMBER blocksize=$SIZE ! mfw_wma8encoder ! filesink
location=output.wma
```

The time duration of recording equals $NUMBER*$SIZE*8/ (samplerate*channel*bitwidth).

For example, to record 60 seconds of stereo channel sample with 44.1K sample rate and 16bit width, use the following command:

```
$gst-launch alsasrc num-buffers=240 blocksize=44100 ! mfw_wma8encoder ! filesink
location=output.wma
```

To verify if the WMA output is correct, use the beepdec:

```
$gst-launch filesrc location=output.wma typefind=true ! aiurdemux ! queue max-size-time=0 !
beepdec ! audioconvert ! 'audio/x-raw-int, channels=2' ! alsasink
```

3.2.2.6 VPU-Based Video Encoder Record

NOTE

The VPU encoder is currently available only for some Freescale i.MX SoCs with embedded VPU.

Camera must be enabled before running video record (to install the camera driver, see the appropriate BSP document). For example:

For 5640:

```
$modprobe ov5640_camera_mipi
$modprobe mxc_v4l2_capture
```

For 5642:

```
modprobe ov5642_camera
modprobe mxc_v4l2_capture
```

Encoding from a file:

```
gst-launch filesrc location=test.yuv blocksize= $BLOCK_SIZE ! 'video/x-raw-yuv,
format=(fourcc)I420, width=$WIDTH, height=$HEIGHT, framerate=(fraction)30/1' !
$video_encoder_plugin codec=0 ! matroskamux ! filesink location=output.mkv sync=false
```

NOTE

The input file support I420 format YUV files.

The blocksize property of the filesrc plug-in depends on the resolution of the input image. For example:

blocksize = inputwidth * inputheight * 1.5

The codec type property of the $video_encoder_plugin plug-in controls the target encode codec type. It could be 0 (MPEG4), 5 (H263), 6 (H264), or 12 (MJPG).

See 'Element Properties' of 'codec' prompted by 'gst-inspect <vpuenc_plugin>' command.

The different cameras need to set different capture mode for special resolution (see the BSP document for camera settings), one example of recording is given below:

```
gst-launch mfw_v4lsrc fps-n=15 capture-mode=X ! queue ! $video_encoder_plugin codec=0 !
matroskamux ! filesink location=output.mkv sync=false
```

NOTE

The fps-n property of the mfw_v4lsrc plug-in controls the camera capture frame rate.

The codec property of the $video_encoder_plugin plug-in controls the target encode codec type. Use the gst-inspect command to get more details about the codec property.

3.2.2.7 SPDIF Transmit and Receive Converter

The SPDIF supports both transmit and receive features with PCM or Non-PCM data. With Non-PCM data, the mfw_spdifrx and mfw_spdiftx plug-ins convert data between the IEC958 format and raw data. In this version, only AC3 data format is supported.

To verify if the SPDIF receive is correct, use the mfw_spdifrx plug-in:

```
$gst-launch alsasrc device="plughw:1,0" ! mfw_spdifrx ! filesink location= test.bits
```

NOTE

For more information, see i.MX 6 Linux User's Guide.

To verify if the SPDIF transmit is correct, use the mfw_spdiftx plug-in:

```
$gst-launch filesrc location= test.bits  ! mfw_spdiftx ! alsasink device="plughw:1,0"
```

NOTE

Insert the snd-spdif.ko kernel module with the insmod command. For more information, see i.MX 6 Linux User's Guide.

The "plughw" parameter depends on the system used.

3.2.2.8 Audio Post-Process

To verify if the parametric equalizer is correct, use mfw_audio_pp:

```
$gst-launch filesrc location=test.mp3 typefind=true ! beepdec ! mfw_audio_pp enable=1
eqmode=2
! alsasink
gst-launch playbin2 uri=file://test.mp3 audio-sink="mfw_audio_pp enable=true eqmode=2 !
alsasink"
```

NOTE

The eqmode value 2 means the "bass booster" scene.

To verify if the downmixing is correct, use mfw_downmixer:

```
$gst-launch filesrc location= test.mp3 typefind=true ! beepdec ! mfw_downmixer ochannels=2 !
alsasink
```

3.2.2.9 Dual Display

In i.MX 6 series, mfw_v4lsink supports dual display.

```
gst-launch playbin2 uri=file:///<filename> video-sink="mfw_v4lsink device=<VIDEO_DEVICE1>"
audio-sink="pulsesink device=<AUDIO_DEVICE1>" &
gst-launch playbin2 uri=file:///<filename> video-sink="mfw_v4lsink device=<VIDEO_DEVICE2>"
audio-sink="pulsesink device=<AUDIO_DEVICE2>"
```

Example on i.MX 6Dual/6Quad SABRE-SD board:

```
gst-launch playbin2 uri=file:///<filename> video-sink="mfw_v4lsink device=/dev/video17"
audio-sink="pulsesink device=alsa_output.platform-soc-audio.4.analog-stereo" &
gst-launch playbin2 uri=file:///<filename> video-sink="mfw_v4lsink device=/dev/video19"
audio-sink="pulsesink device=alsa_output.platform-soc-audio.5.analog-stereo"
```

NOTE

The U-Boot command-line should be set correctly for dual display. For details, see i.MX Linux User's Guide.

3.2.2.10 Transcoding

A command-line example for transcoding is given below:

```
gst-launch filesrc location=<MEDIA_FILE> typefind=true ! aiurdemux ! vpudec ! mfw_ipucsc !
'video/x-raw-yuv, format=(fourcc)NV12, width=1280, height=720' ! vpuenc ! matroskamux !
filesink location=720p.mkv
```

3.2.3 gplay Player

gplay is a command-line based player. It is based on GStreamer playbin2 element and provides full functions of playback, including trick mode, video display setting, and so on.

The command-line is:

```
$gplay $MEDIA_FILE
```

For more information on gplay tool, see the GStreamer Command-Line Player Application Specification document included in the multimedia release package.

3.2.4 Totem Player

Totem is the official movie player of the GNOME desktop environment based on GStreamer, it's a GUI-based player running on Linux desktop system. Totem is by default installed on i.MX 6 series running Ubuntu operating system or Gnome mobile operating system.

For more information on the Totem player, see Totem help.

The command-line is:

```
$totem $MEDIA_FILE
```

To use the Totem player in serial terminal, the following environment need to be set:

```
$export DISPLAY=:0
$totem $MEDIA_FILE
```

3.3 Testing Core Codec Libraries

Some core codec libraries have no corresponding GStreamer plug-ins, such as image decoder/encoder and some audio encoders. To view the list of GStreamer plug-ins, see i.MX 6 Series Ubuntu Multimedia Release Notes.

To test those core codec libraries, use Freescale proprietary test applications included in the codec/parser binary package.

3.4 Debug Exception in Multimedia Plug-In

In the GDB debug mode, some multimedia plug-ins might generate exceptions on their system check initialization but they are safe to continue because the exceptions are handled directly by the multimedia components. Processing these exceptions might disturb the debug environment. To avoid this, the debugger can be configured such that it handles these exceptions automatically without asking the user to provide input. This can be done by using the below command:

```
$ handle SIGBUS nostop
```

This command should be added to .gdbinit script as the default setting to debug the multimedia plug-ins.

Chapter 4
Multi-Overlay Support

mfw_isink plug-in is a IPU library-based sink element for GStreamer that provides multi-overlay support for video playback. It means, several video playback can run on the same display device or different one, such as DVI and/or TV and DVI and/or WVGA* at the same time, and the size and position of display window for each video can be set separately. Currently, mfw_isink plug-in is only available on i.MX 6 series platform.

NOTE

Modify the vssconfig as follows:

vss device definition

Master=DVI, Slave=TV

add "video=video=mxcdi1fb:YUV444,720P60 video=mxcdi0fb:RGB24,1024x768M-16@60" to kernel startup command-line

master display

[master]

type = framebuffer

format = RGBP

fb_num = 1

main_fb_num = 0

Each mfw_isink supports two configurations for the same input video. It can also construct more GStreamer pipelines with mfw_isink to support different video playback contents.

4.1 How to Use mfw_isink

As a standard GStreamer plug-in, gst-launch tool and GStreamer-based application (such as Totem) can use mfw_isink as a video sink element.

Since mfw_isink needs to access IPU and framebuffer devices. So, you need to either log on as the root user or run the following command in a terminal window to change the corresponding device permission:

```
$chmod 666 /dev/mxc_ipu /dev/fb0 /dev/fb1 /dev/fb2 /sys/class/graphics/fb1/mode
/sys/class/graphics/fb1/pan /sys/class/graphics/fb2/pan
```

By default, mfw_isink uses fb2 as a display framebuffer on LCD. Since fb2 is invisible, by default. So, you need to run the following command in a terminal window to enable mfw_isink local alpha feature when using mfw_isink with gst-launch:

```
$export VSALPHA=1
```

4.1.1 Using mfw_isink with gst-launch

The following command illustrates a complete pipe to play back an avi file by mfw_isink as a video sink element with advanced property settings. It plays back video on LCD and video position starts at (100, 100) with window size 640x480.

```
$gst-launch filesrc location=test.avi typefind=true ! aiurdemux ! vpudec ! mfw_isink
axis-top=100 axis-left=100 disp-width=640 disp-height=480
```

or,

```
$gst-launch playbin2 uri=file:///test.avi video-sink="mfw_isink axis-top=100 axis-left=100
disp-width=640 disp-height=480"
```

mfw_isink also supports other properties for display settings. Use $gst-inspect mfw_isink command to get the detailed list of supported properties.

Some of these properties are described below:

- display: Sets display device (name) for config 0 (for DVI and TV, the setting is "DVI" or "TV;" see vssconfig file under /usr/share for detailed information on output device name)
- axis-top: y position for top-left corner of video window in pixels for config 0
- axis-left: x position for top-left corner of video window in pixels for config 0
- disp-width: Width of display window in pixels for config 0
- disp-height: Height of display window in pixels for config 0
- mode: Display mode for config 0 (for available value, see "mode" section in vssconfig file under /usr/share)

- display-1: Set display device (name) for config 1 (for DVI and TV, the setting is "DVI" or "TV;" see vssconfig file under /usr/share for detailed information on output device name)
- axis-top-1: y position for top-left corner of video window in pixels for config 1
- axis-left-1: x position for top-left corner of video window in pixels for config 1
- disp-width-1: Width of display window in pixels for config 1
- disp-height-1: Height of display window in pixels for config 1
- mode-1: Display mode for config 1 (for available value, see "mode" section in vssconfig file under /usr/share)

Running several gst-launch commands with mfw_isink will show several videos. An example is given below.

The following command plays back one video in one display, PIP:

```
$gst-launch playbin2 uri=file:///1.avi video-sink="mfw_isink display=DVI display-1=DVI
axis-top=100 axis-left=100 disp-width=640 disp-height=480"
```

The following command plays back two videos in two displays:

```
$gst-launch playbin2 uri=file:///1.avi video-sink="mfw_isink display=DVI" playbin2
uri=file:///2.avi video-sink="mfw_isink display=TV"
```

4.1.2 Using mfw_isink with Totem Player

mfw_v4lsink is the default video sink element. To use mfw_isink with Totem player, run the following command:

```
$gstreamer-properties
Then, set Video --> Default Ouptut --> Pipeline to mfw_isink as shown in the below figure.
```

Figure 4-1. gstreamer-properties Tool

NOTE

If the gstreamer-properties tool is not found, install gnome-media package.

Use the following command to launch multiple Totem players:

```
$totem --no-existing-session
```

Then, play back videos in opened Totem players. You can move/resize each Totem window separately.

NOTE

In Ubuntu 11.10 Oneiric rootfs, the Totem player does not have the parameter of --no-existing-session. So, you need to create two different user accounts to open the Totem players separately.

Chapter 5
Streaming Support

5.1 HTTP Support

Freescale multimedia framework supports HTTP protocol-based streaming. For testing with HTTP protocol-based streaming, an HTTP server with test content is required. Freescale recommends you to use a Linux server with Apache2.

Use playbin2 to test:

```
$gst-launch playbin2 uri=http://SERVER/test.avi
```

Use gplay to test:

```
$gplay http://SERVER/test.avi
```

MPEG2TS stream may not play back using the above commands due to a limitation of typefind plug-in in GStreamer 0.10.35. Therefore, use the following command:

```
$gst-launch souphttpsrc location=http://SERVER/test.ts ! 'video/mpegts' ! aiurdemux
name=demux
demux. ! queue max-size-buffers=0 max-size-time=0 ! vpudec ! mfw_v4lsink demux. ! queue
max-size-buffers=0 max-size-time=0 ! mfw_ac3decoder ! alsasink
```

NOTE

In GNOME rootfs built by running the ltib command, HTTP streaming is currently not supported.

5.2 DLNA/UPnP Support

Freescale multimedia framework supports the Totem application running as a DLNA/UPnP client. To add support for DLNA/UPnP, you need to install the totem-plugins-extra package for your Ubuntu system on i.MX 6 series board. This can be done by running the following command:

```
$apt-get install totem-plugins-extra
```

Besides the DLNA/UPnP client, you also need a DLNA/UPnP server with support for media file sharing.

Now, open the Totem player and select Edit --> Plugins… to open the Configure Plugins dialog box. In the Configure Plugins dialog box, select the check box next to "Coherence DLNA/UPnP Client" and click Close.

In the Totem window, select "Coherence DLNA/UPnP Client" from the drop-down list on the right sidebar. This will display a list of available media servers. Choose a media file to play back.

NOTE

Totem in Oneiric: Totem DLNA plug-in support is removed (see https://bugs.launchpad.net/ubuntu/+source/totem/+bug/827382).

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: IMX6UBUNTUUG
Rev. L3.0.35_4.1.0
09/2013