# i.MX Linux Multimedia Framework

## User's Guide

**Document Number: 924-76335**
**Rev. 3.0.5**
**12.2012**

freescale™
semiconductor

### *How to Reach Us:*

**Home Page**:
www.freescale.com

**Web Support**:
http://www.freescale.com/support

**USA/Europe or Locations Not Listed**:
Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

**Europe, Middle East, and Africa**:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

**Japan**:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064, Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

**Asia/Pacific**:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 010 5879 8000
support.asia@freescale.com

***For Literature Requests Only:***
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

# Contents

## 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.

## Audience

This document is intended for software, hardware, and system engineers who are planning to use the Multimedia codecs with Gstreamer architecture and for anyone who wants to understand more about the Multimedia codecs.  A basic understanding of Gstreamer and LTIB architecture is required.

## Organization

This document contains the following chapters.

| | |
|---|---|
| Chapter 1 | Identifies the BSP requirements, and explains how to build the multimedia components from LTIB or install multimedia components on Ubuntu OS |
| Chapter 2 | Explains how to test and use the multimedia codecs. |

## Conventions

This document uses the following conventions:

| | |
|---|---|
| `Courier` | Is used to identify commands, explicit command parameters, code examples, expressions, data types, and directives. |
| *Italic* | Is used for emphasis, to identify new terms. For replaceable command parameters it will start with *$*. |

## References

The following documents were referenced to build this document.

1. i.MX Linux User’s Guide

2. i.MX Linux Multimedia Framework Release Notes

3. i.MX Advanced ToolKit Standard User’s Guide

## Definitions, Acronyms, and Abbreviations

The following list defines the abbreviations used in this document.

| | |
|---|---|
| FSL | **F**ree**s**ca**l**e |
| Codec | **co**der-**dec**oder |
| LTIB | **L**inux **T**arget **I**mage **B**uilder |
| ARM | **A**dvanced **R**ISC **M**achine |
| ASRC | **A**synchronous **S**ample **R**ate **C**onverter |
| APT | **A**dvanced **P**ackaging **T**ool |
| Gst | Gstreamer (open source multimedia framework) |
| gplay | Freescale command line player with Gstreamer backend |

# Chapter 1
# Installing and Building the Plugins

This chapter describes how to build/install Freescale multimedia core libraries and Freescale Gstreamer plugins.

Freescale multimedia core libraries are released in binary only. Freescale Gstreamer plugins include source code.

Freescale provides two types of release packages; LTIB packages are for Freescale core libraries and Gstreamer plugins, while Debian packages are for Ubuntu system.

The LTIB packages contain Freescale multimedia core binary libraries and Freescale Gstreamer plugins source code. (Refer to section 1.1)

Debian binary packages are used to install Freescale core libraries and Gstreamer plugins binaries into an i.MX series board running Ubuntu OS. Debian source packages are used to build Freescale Gstreamer plugins on an i.MX series board. (Refer to section 1.2)

## 1.1 Building the Plugins with LTIB

### 1.1.1 BSP Requirements

Requirements:

- i.MX series board
- Compliant i.MX series Linux BSP 12.12 (1.0.0) or above.
- Gstreamer
  - Gstreamer (version>=0.10.35)
  - Gstreamer-plugins-base (version>=0.10.35)
  - Gstreamer-plugins-good (version>=0.10.30)

**NOTE**

The Freescale Gstreamer plugins are dependent on the Gstreamer framework including the Gstreamer Core, Gst-Plugins-base, and Gst-Plugins-good.

### 1.1.2 Building the Plugins with LTIB

Following LTIB related procedures are running on a PC(x86) with a Linux OS.

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

1. Install LTIB on PC.

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

   This command installs LTIB to your directory.

   For instructions, see the *i.MX Linux User's Guide* for the target platform.
2. Obtain the following packages included in the release

   There are two standard packages for building the Freescale multimedia Linux codecs.

   Standard packages:
   - gst-fsl-plugins-$VERSION.tar.gz is **gstreamer plugin source package** that contains source code for the multimedia Gstreamer-based plugin for the i.MX application processor.
   - libfslcodec -$VERSION.tar.gz is **codec binary package** that contains the Freescale multimedia core codec libraries for the i.MX application processor.
   - libfslparser -$VERSION.tar.gz is **parser binary package** that contains the Freescale multimedia core parser libraries for the i.MX application processor.
   - libfslvpuwrap -$VERSION.tar.gz is **vpu-wrap source package**.
   - gst-plugins-gl-[version].tar.gz is **GL plugin source package.**
   - fsl-alsa-plugins-[version].tar.gz is **ALSA plugin source package**
3. Copy these standard packages to the LPP directory, which by default is set to /var/tmp/pkgs (please see litb/.ltibrc %ldirs).

**NOTE**

For the first LTIB installation create this directory manually.

To build the package, please follow these steps:

1. Select the platform

   Please refer to corresponding chapter in <i.MX [board_name] Linux User Guide.pdf> on "Building the Linux Platform".

2. Select **Package List > Freescale Multimedia Plugins/Codecs**.

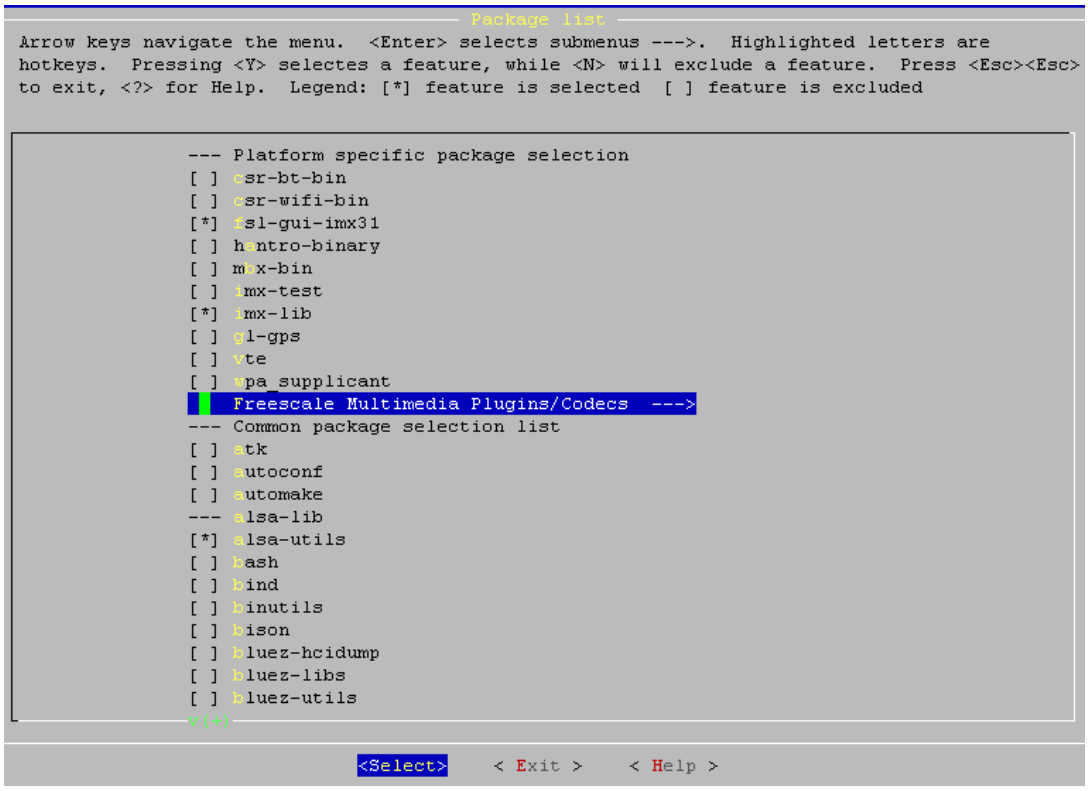


Figure 1 LTIB Package Selection Menu

3. Select libfslcodec, libfslparser, libfslvpuwrap and gst-fsl-plugins, gst-plugins-gl, gst-alsa-plugins (Figure 2). ()

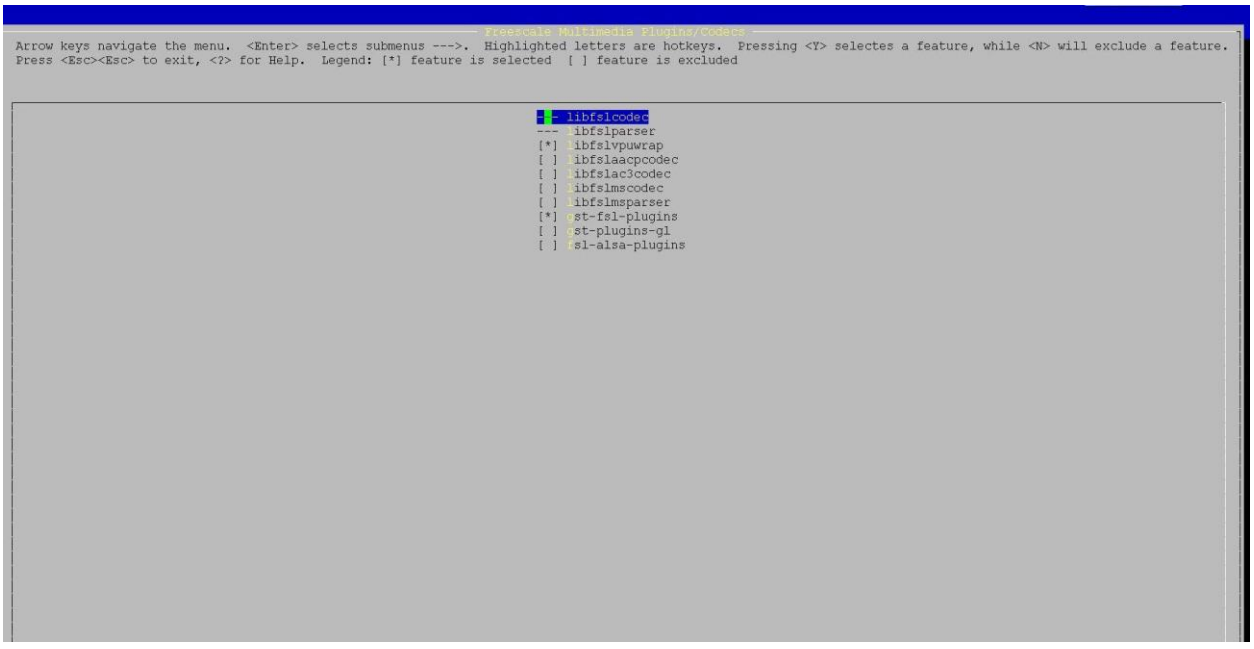


Figure 2 Selecting the Plugins

4. Select **gstreamer-plugins-good** package (Figure 3)

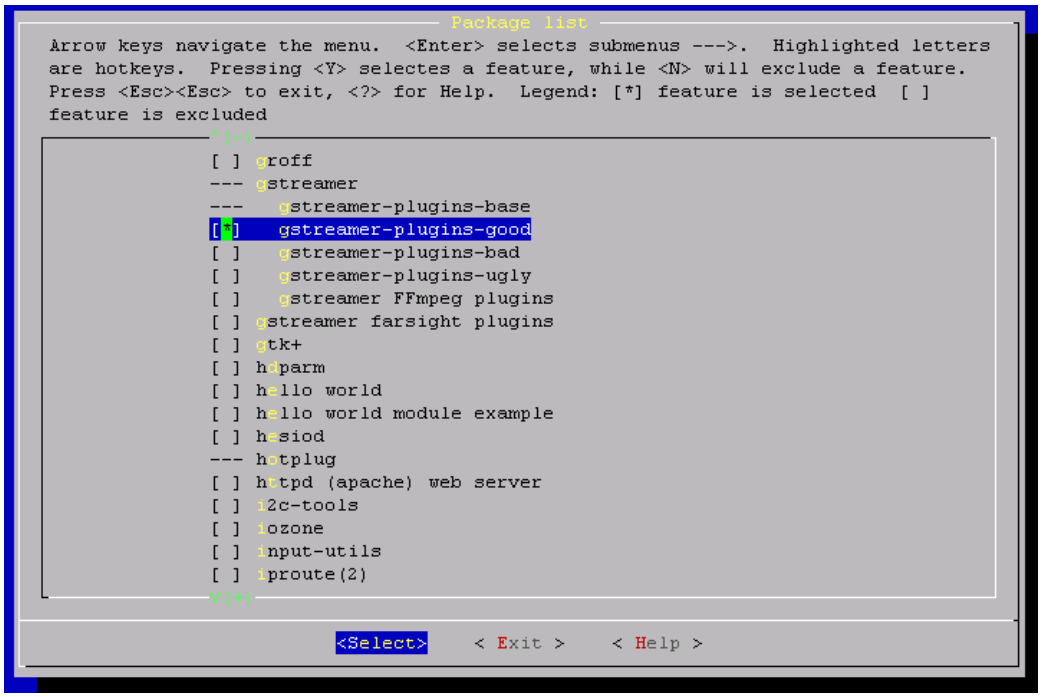


**Figure 3 Selecting gstreamer-plugins-good package**

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 the Freescale multimedia Linux codecs and Freescale Gstreamer plugins. The uImage is located in rootfs/boot in LTIB build directory.

For each platform's detail information, please refer to *i.MX Linux User's Guide*.

**NOTE**

If FSL Multimedia plugin "gst-plugins-gl" is selected, the BSP package "gpu-viv-bin-mx6q" need to be selected also.

# Installing/Building the Plugins on Ubuntu

## 1.1.3 BSP Requirements

Requirements:

- i.MX51/ i.MX53/ i.MX6 board runs Ubuntu OS (11.10 Oneiric).
- imx-lib Debian package (version>=12.12(1.0.0)).
- firmware-imx Debian package (version>=12.12(1.0.0)).
- Gstreamer
  - Gstreamer (version=0.10.35 on Ubuntu 11.10)
  - Gstreamer-plugins-good (version=0.10.30 on Ubuntu 11.10)

## 1.1.4 Installing/Building the Plugins

This chapter describes how to build and generate debian packages on the i.MX series board natively.

Following steps illustrate how to install Freescale Multimedia Plugins on Ubuntu and also how to build a Debian package from the source.

1. Prepare an i.MX6 board running Ubuntu OS(11.10 Oneiric)
2. Install BSP support libraries package, the package is released with BSP. (you need to have the .deb file available to the board, a USB-Key or copying the file to the board in some other way should be fine)

   **sudo dpkg -i** imx-lib-$VERSION-$RELEASE.deb

   **sudo dpkg -i** kernel_$VERSION-imx_$RELEASE_armel.deb

   It is desired to install all other BSP Debian packages.
3. Obtain the following Debian binary packages, which are included in the Debian release. (i.e. you can copy them to a USB-Key that will be connected to the board)

Some BSP related packages needed (repackage for avoid some toolchain compatibility issue)

| Package name | Content | License |
|---|---|---|
| imx-lib | imx-lib-test-ubt_[version]_armel.deb<br>imx-lib-ubt0_[version]_armel.deb | |
| gpu-viv | gpu-viv-ubt0_[version]_armel.deb | |

| Package name | Content | License |
|---|---|---|
| FSL plugins | gstreamer0.10-plugins-fsl_$VERSION_armel.deb<br>gstreamer0.10-plugins-fsl-dbg_$VERSION_armel.deb<br>gstreamer0.10-plugins-fsl-doc_$VERSION_all.deb | Standard |
| FSL command media player | gstreamer0.10-plugins-fsl-tools_$VERSION_armel.deb<br>gstreamer0.10-plugins-fsl-tools-dbg_$VERSION_armel.deb | Standard |
| FSL multiple audio core codecs | libfslaudiocodec1_$VERSION_armel.deb<br>libfslaudiocodec-dev_$VERSION_armel.deb<br>libfslaudiocodec-doc_$VERSION_all.deb<br>libfslaudiocodec-test_$VERSION_armel.deb | Standard |
| FSL parser core libraries | libfslparser3_$VERSION_armel.deb<br>libfslparser-dev_$VERSION_armel.deb<br>libfslparser-doc_$VERSION_all.deb | Standard |
| FSL video and image core codecs | libfslvideocodec1_$VERSION_armel.deb<br>libfslvideocodec-dev_$VERSION_armel.deb<br>libfslvideocodec-doc_$VERSION_all.deb<br>libfslvideocodec-test_$VERSION_armel.deb | Standard |
| FSL VPU wrapper library | libfslvpuwrap3_$VERSION_armel.deb<br>libfslvpuwrap3-dbg_$VERSION_armel.deb<br>libfslvpuwrap-dev_$VERSION_armel.deb<br>libfslvpuwrap-doc_$VERSION_all.deb | Standard |
| FSL GST libraries | libgstreamer-plugins-fsl0.10-0_$VERSION_armel.deb<br>libgstreamer-plugins-fsl0.10-dbg_$VERSION_armel.deb<br>libgstreamer-plugins-fsl0.10-dev_$VERSION_armel.deb | Standard |
| FSL GL plugin | libgstreamer-plugins-gl0.10_$VERSIONarmel.deb<br>libgstreamer-plugins-gl0.10-dbg-$VERSION_armel.deb<br>libgstreamer-plugins-gl0.10-dev_$VERSION_armel.deb | Standard |
| FSL ALSA plugin | fsl-alsa-plugins_$VERSION_armel.deb<br>fsl-alsa-plugins-dbg_$VERSION_armel.deb<br>fsl-alsa-plugins-dev_$VERSION_armel.deb | Standard |
| AAC plus core decoder | libfslaacpaudiocodec1_$VERSION_armel.deb<br>libfslaacpaudiocodec-dev_$VERSION_armel.deb<br>libfslaacpaudiocodec-doc_$VERSION_all.deb<br>libfslaacpaudiocodec-test_$VERSION_armel.deb | License restricted |
| AC3 core decoder | libfslac3audiocodec1_$VERSION_armel.deb<br>libfslac3audiocodec-dev_$VERSION_armel.deb<br>libfslac3audiocodec-doc_$VERSION_all.deb<br>libfslac3audiocodec-test_$VERSION_armel.deb | License restricted |
| Microsoft audio core codecs | libfslmsaudiocodec1_$VERSION_armel.deb<br>libfslmsaudiocodec-dev_$VERSION_armel.deb<br>libfslmsaudiocodec-doc_$VERSION_all.deb | License restricted |

| | | |
|---|---|---|
| | libfslmsaudiocodec-test_$VERSION_armel.deb | |
| Microsoft parser | libfslmsparser3_$VERSION_armel.deb<br>libfslmsparser-dev_$VERSION_armel.deb<br>libfslmsparser-doc_$VERSION_all.deb | License restricted |
| Microsoft video core codecs | libfslmsvideocodec1_$VERSION_armel.deb<br>libfslmsvideocodec-dev_$VERSION_armel.deb<br>libfslmsvideocodec-doc_$VERSION_all.deb<br>libfslmsvideocodec-test_$VERSION_armel.deb | License restricted |

Use dpkg command to install them:

**sudo dpkg -i** *.deb

4. To build the codec and plugin 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 plugins:*

**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 in MM release to solve the build dependency.

5. Create a directory, and copy all the source Debian packages to this directory.

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

6. For each of the above packages, run below command to build:

```
dpkg-source -x <file.dsc>

cd <newly_generated_directory>

debuild -i -uc -us
```

The debian binaries will be created at the upper directory, and you can install them by "dpkg -i <package.deb>".

# Chapter 2
# Testing the 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 that is being created in the Gstreamer architecture.

**NOTE**

Each platform provides a certain set of codecs. Please refer to the Release Notes to determine which codecs are included in the BSP.

## 2.1 Testing multimedia environment setting

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

### 2.1.1 Audio output Setting

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

```
pactl list sinks
```

A list of available audio sinks will be displayed:

```
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 accordingly as the sink number in list showed 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, Please use "`aplay -l`" to check the audio device. And use "`gst-launch playbin2 uri=<stream> audio-sink="alsasink device=plughw:$audio_device_number"`" to output to the specific device.

### 2.1.2 Audio input

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

```
pactl list sources
```

A list of available audio sources will be displayed:

```
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 accordingly as the source number in list

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

Note: if not need record and playback at the same time, there is no need to set to monitor mode.

Use "pactl" command to see the current status of audio I/O path setting.

```
pactl stat
```

A list will be displayed like following:

```
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
```

### 2.1.3 Video setting

According to different video output (HDMI / LVDS  etc.) , please refer to BSP User Guide for how to set boot cmd in uboot.

### 2.1.4 ASRC  Setting

To use ASRC plugin:

- disable pulseaudio on UBUNTU (or use LTIB built rootfs)
- please modify /etc/asound.conf  provided by BSP package as following:

  Change "defaults.pcm.rate_converter" from " linear"  to "asrcrate"

Limitation:

- The ASRC hardware can at most support 3 instances and 10 channels simultanously. If you want to make use of asrc plugins, make sure you won't create more than 3 asrc plugin instances, and total number of converting channels is less than or equal to 10. For example, you can run 3 asrc converters with channels 2, 2, 6, while you can't run 2 asrc converters with channels 6, 6.
- ASRC hardware can only support some fixed sample rates, don't make use it if you don't know which rates are in your cases.

  Input: 8000 16000 22050 32000 44100 48000 64000 88200 96000 176400 192000

  Output: 32000 44100 48000 64000 88200 96000 176400 192000

## 2.2 Testing the Codecs with Gstreamer

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

### 2.2.1 gst-inspect Tool

The **gst-inspect** tool can provide information about an available Gstreamer plugin, a particular plugin, or a particular element.

To view the list of installed freescale multimedia codec plugins, type the following command in a shell:

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

A list similar to the following is displayed.

```
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 maybe different depend on the target platform

For example:

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

The first “vpu.imx” is plugin name, the second “vpudec” is element name, “VPU-based video decoder” is long name of element.

Use following gst-inspect command to view the detail information of an element.

```
gst-inspect $ELEMENT_NAME
```

For example,

```
gst-inspect vpudec
```

to display detail information of element vpudec

All these plugins can be classified into audio decoder/encoder, video decoder/encoder, demuxer, etc.

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

## 2.2.2 gst-launch Tool

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

### 2.2.2.1 Playback with playbin2

Freescale recommends using Gstreamer playbin2 plugins to playback audio or/and video. Playbin 2 is self-constructed pipeline elements which will auto connect all necessary elements to decode a media file/resource, including source, parser, decoder and sink, etc. The command is

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

The *$URI* is Universal Resource Identifier. For a local file, URI start with file://, for example, to play a local file test.avi locate in /media directory, please use

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

**NOTE**

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

#### 2.2.2.2 Audio playback

Use the beep unified audio decoder plugin to test MP3 / AAC / WMA / Vorbis / AC3 audio playback

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

For example

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

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

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

**NOTE**

For this test, the Gstreamer Good Plugin package must be installed.Due to hardware and opensource element limitation, for some combine configurations of specific channels and samplerate, the sound may not be heard.

#### 2.2.2.3 Video only playback

To create a video-only pipeline with the gst-launch tool, use these commands:

```
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 this command:

```
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
```

### 2.2.2.4 AV 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 the Freescale i.MX SoC with embedded VPU.

The **max-size-time** in Queue element should be set because the playback could be not smoothly with default value one second.

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

```
On SoC without VPU, like MX508, MX6SL:
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, like MX6DL, MX6DQ:
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 the Freescale i.MX SoC with embedded VPU.

### 2.2.2.5 Audio Encoder Record

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

**MP3 Encoder Record**

Encoding from 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

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

To verify that 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 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

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

To verify that 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
```

#### 2.2.2.6 VPU based Video Encoder Record

**NOTE**

The VPU encoder is currently available only for some of the Freescale i.MX SoC with embedded VPU.

Camera must be enabled before running video record. For the camera driver install, please refer 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 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** plugin depends on the resolution of the input image. For example:

```
blocksize = inputwidth * inputheight * 1.5
```

The **codec type** property of the **$video_encoder_plugin** plugin control the target encode codec type. It could be 0(MPEG4), 5(H263), 6(H264) or 12(MJPG).

Please refer to 'Element Properties' of 'codec' prompted by command 'gst-inspect <vpuenc_plugin>'

The different camera need set different capture mode for special resolution, please refer BSP document for different camera setting, one example for Recording:

```
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** plugin control the camera capture frame rate.

The **codec** property of the **$video_encoder_plugin** plugin control the target encode codec type. The detail value please refer the property by gst-inspect command.

### 2.2.2.7 SPDIF Transmit and Receive Converter

The SPDIF supports both transmit and receive feature with PCM or Non-PCM data. With Non-PCM data, the **mfw_spdifrx** and **mfw_spdiftx** plugin convert data between the IEC958 format and raw data. In this version, only support AC3 data format.

To verify the SPDIF receive is correct, use the **mfw_spdifrx**:

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

**NOTES**

The SPDIF feature is applied in i.MX35 platform. For more information, see the *i.MX Linux User's Guide* for target platform.

To verify the SPDIF transmit is correct, use the **mfw_spdiftx**:

```
$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 i.MX35 3-Stack board, **snd-spdif** module will be built in **rootfs**. For more information, see the *i.MX Linux User's Guide*.

The "plughw" parameter depends on target system.

#### 2.2.2.8 Audio Post-Process

To verify the Parametric EQ is correct, use the **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 the Downmixing is correct, use the **mfw_downmixer**:

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

#### 2.2.2.9 Dual display

From i.MX6 on, mfw_v4lsink can support 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.MX6DQ 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 uboot command line should be set correctly for dual display. For details, please check the BSP user guide.

#### 2.2.2.10 Transcoding

The command line example as following:

```
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
```

### 2.2.3 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 etc.

The command line is

```
$gplay $MEDIA_FILE
```

For detail information of gplay tool, please refer to

*Gstreamer_Command-line_Player_Application_Specification.pdf*.

### 2.2.4 Totem Player

Totem is the official movie player of the GNOME desktop environment based on GStreamer, it's a graphic UI based player running on Linux desktop system. Totem is default installed on i.MX51/i.MX53/i.MX6 running Ubuntu OS or Gnome mobile OS.

For detail information of using totem, please refer to totem help.

The command line is

```
$totem $MEDIA_FILE
```

To use totem in serial terminal , following environment need to be set

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

## 2.3 Testing the Core Codec Libraries

Some core codec libraries have no corresponding Gstreamer plugins, such as the **image** and some **audio encoders**. To view the list of Gstreamer plugins, see the *i.MX Multimedia Framework Linux Release Notes.*

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

## 2.4 Debug exception in multimedia plugin

In the GDB debug mode, some multimedia plugins might generate exceptions on their system check initialization but are safe to continue since the exceptions are handled directly by the multimedia components. This might disturb a debug environment with processing these exceptions. The following step specifies how to configure the debugger so that these exceptions are handled automatically without user input needed.

```
$ handle SIGBUS nostop
```

Add this command to .gdbinit script as the default setting to debug the multimedia plugins.

# Appendix A: Multi-overlay support

**mfw_isink** plugin for Gstreamer is a IPU lib based sink element which provides multi-overlay support of video playback. It means several video playback can run the same time on the same display device or different one, eg. DVI and/or TV and DVI and/or WVGA* , each video can setting the display windows size and position. Currently, mfw_isink plugin is only available on i.MX51, i.MX53 and i.MX6X platform.

**NOTE**

- For MX5

  default setting for mfw_isink is DVI and TV, if you want use DVI and WVGA please replace vssconfig with vssconfig.div_wvga under /usr/share directory.

- For MX6
  Modify the vssconfig as following:

```
# vss device definition
# Master=DVI, Slave=TV
# please 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 2 configs for the same input video. It can also construct more gstreamer pipelines with mfw_isink to support different video playback contents.

In multi-overlay case, the video maybe not be smoothly due to performance limitation. Generally, i.MX51/ i.MX53 can support 2-way D1 resolution video playback smoothly.

## A.1 How to use mfw_isink

As a standard gstreamer plugin, gst-launch tool and gstreamer-based application (like totem) can use mfw_isink as video sink element.

Since mfw_isink need access IPU and framebuffer devices, please use as root user or run following command in a terminal window to change 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
```

mfw_isink default use fb2 as display frame buffer on LCD. Since fb2 is default invisible, Please run following command in a terminal window to enable mfw_isink local alpha feature when use mfw_isink with gst-launch

```
$export VSALPHA=1
```

### A.1.1 gst-launch

The following command illustrates a complete pipe to playback an avi file by mfw_isink as a video sink element with advanced property settings. It will playback video on LCD and video position start 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 support other properties for display settings. Please use

```
$gst-inspect mfw_isink
```

to get detail support property list

Following are the detailed information of some of the properties

display: set display device(name) for config 0 (Setting for DVI and TV case is “DVI” or “TV”, please refer vssconfig file under /usr/share for detail output device name).

axis-top: y position for top-left corner of video window in pixel for config 0

axis-left: x position for top-left corner of video window in pixel for config 0

disp-width: width of display window in pixel for config 0

disp-height: height of display window in pixel for config 0

mode: display mode for config 0(available value please refer “mode” section in vssconfig file under /usr/share)

display-1: set display device(name) for config 1 (Setting for DVI and TV case is “DVI” or “TV”, please refer vssconfig file under /usr/share for detail output device name).

axis-top-1: y position for top-left corner of video window in pixel for config 1

axis-left-1: x position for top-left corner of video window in pixel for config 1

disp-width-1: width of display window in pixel for config 1

disp-height-1: height of display window in pixel for config 1

mode-1: display mode for config 1(available value please refer “mode” section in vssconfig file under /usr/share)

Run the several gst-launch command with mfw_isink will show several videos.

Following command illustrate some cases

Playback one video in same 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”
```

Playback 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”
```

### A.1.2 Totem player

mfw_v4lsink is the default video sink element. In order to use mfw_isink with totem player, running the following command and Set “Video->Default Ouptut-> Pipeline to **mfw_isink** (Figure 4)

```
$gstreamer-properties
```

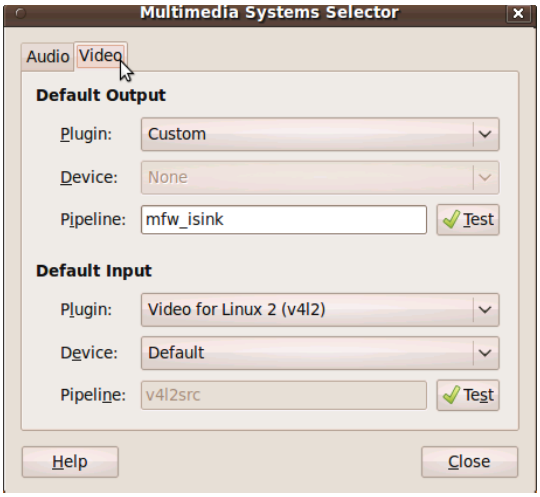


**Figure 4 gstreamer-properties**

**NOTE**

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

Use following command  to launch multi totem players

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

Then playback videos in opened totem players. All the totem window can move/resize separately.

**NOTE**

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

# Appendix B: Streaming support

## B.1 http support

Freescale multimedia framework supports http protocol based streaming.

A http server with test content is required for test with http protocol based streaming, we suggest Apach2 on Linux server.

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 playback by using above commands because limitation of typefind plugin in gstreamer 0.10.28, use 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 ltib built gnome rootfs, http streaming is not supported so far.

## B.2 DLNA/UPnP support

Freescale multimedia framework supports the totem application running as a DLNA/UPnP client. The totem-plugins-extra package need to be installed for your Ubuntu system on i.MX5 or i.MX6 board by following command

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

Also a DLNA/UPnP server with media files sharing is required.

Use menu “Edit”->”Plugins…” to open “Configure Plugins” dialog, enable the “Coherence DLNA/UPnP Client”.

In the sidebar of the totem window, select “Coherence DLNA/UPnP Client”. The media servers will be listed. Choose the media file to playback.

**Note**

Totem in Oneiric: Totem DLNA plugin support is removed (refer to https://bugs.launchpad.net/ubuntu/+source/totem/+bug/827382)