GStreamer Investigation

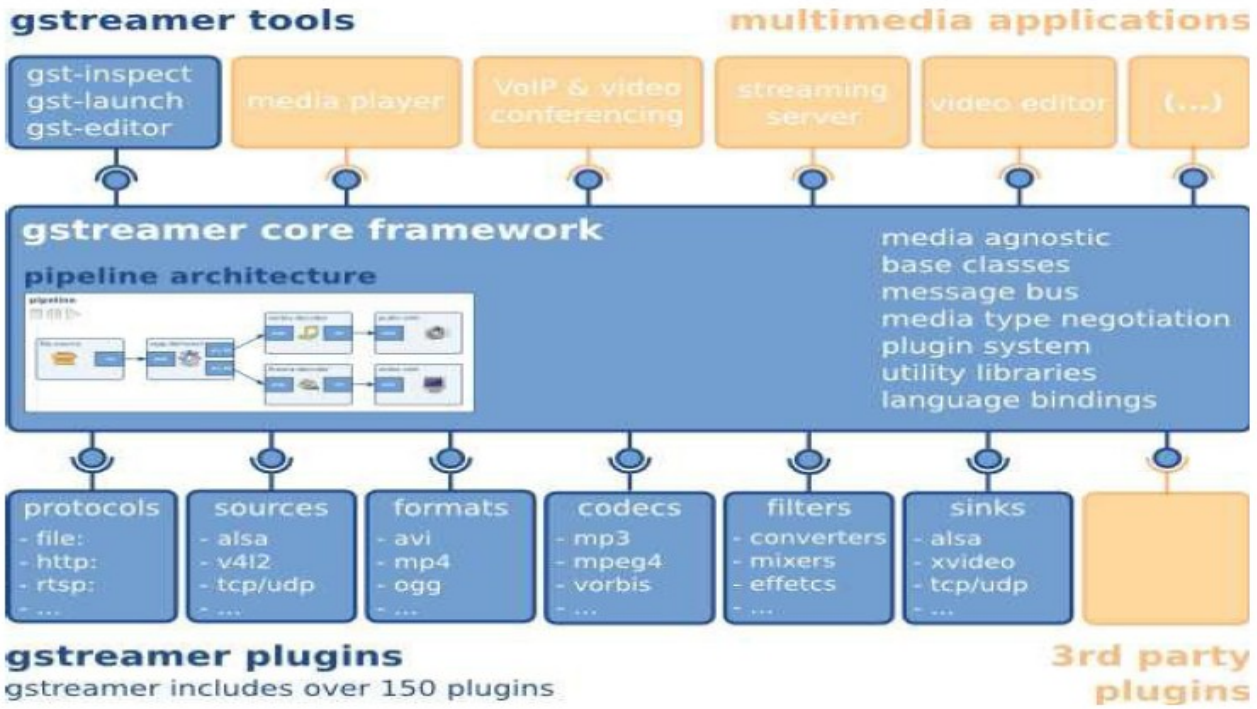
Renesas Design Vietnam Co., Ltd.
Anh Hoang

11/12/12 Rev. 0.00

GStreamer overview

GStreamer Overview

- GStreamer is a framework for creating streaming media applications.
- The GStreamer framework is designed to make it easy to write applications that handle audio or video or both.



GStreamer Architecture

GStreamer Element

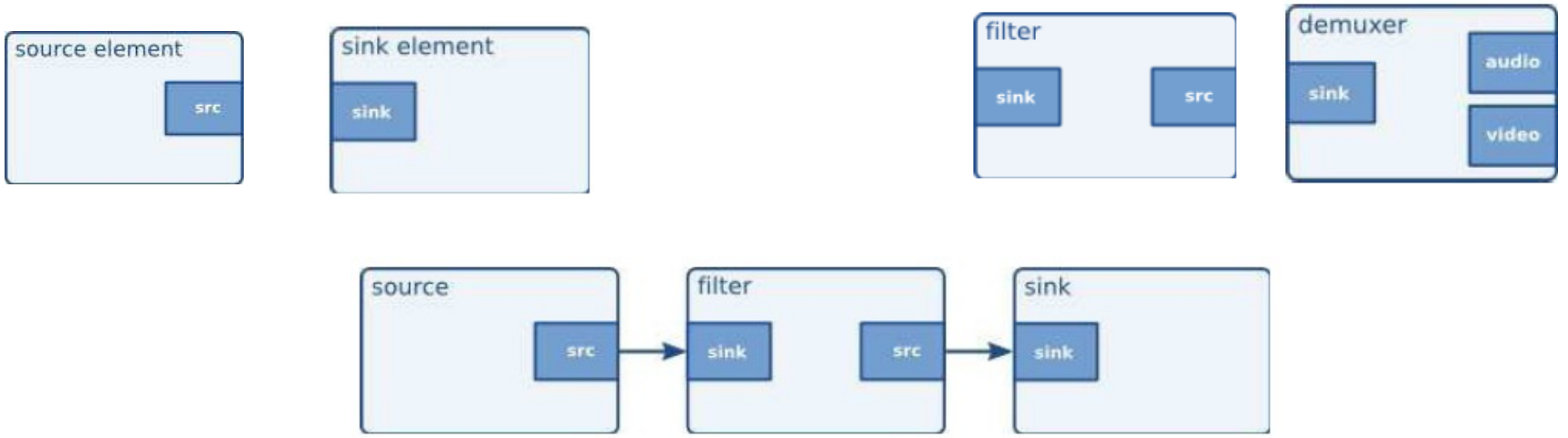
- An element is the most important class of objects in GStreamer
- An element has one specific function, which can be the reading of data from a file, decoding of this data or outputting this data to sound card (or anything else).
- With element, we might put something in, the element does something with it and something else comes out at the other side.
- Type of elements:
  - Source element
  - Filter-like element
  - Sink element
  - Linking element

GStreamer Element

- Source element: generate data for using by a pipeline.

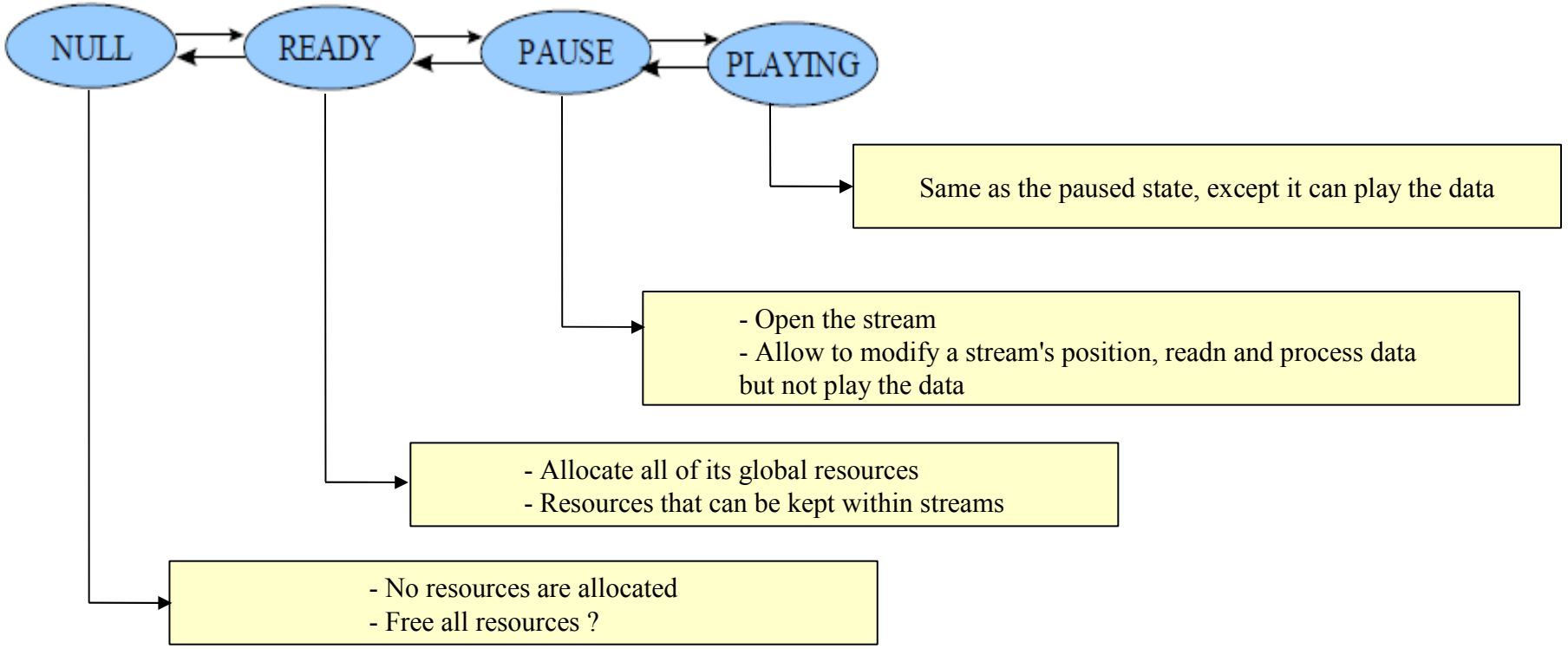
Ex: reading from disk or from a sound card and give output.

- Filter-like element: have both input and output data. Type of this element include filters, convertors, demuxers, muxers and codecs.
- Sink element: end points in a media pipeline. It just receives data.
- Linking element: by linking a source element with zero or more filter-like elements and finally a sink element, you set up a media pipeline. This creates simple chain of elements for processing data from source to sink.
- Below are images for elements:



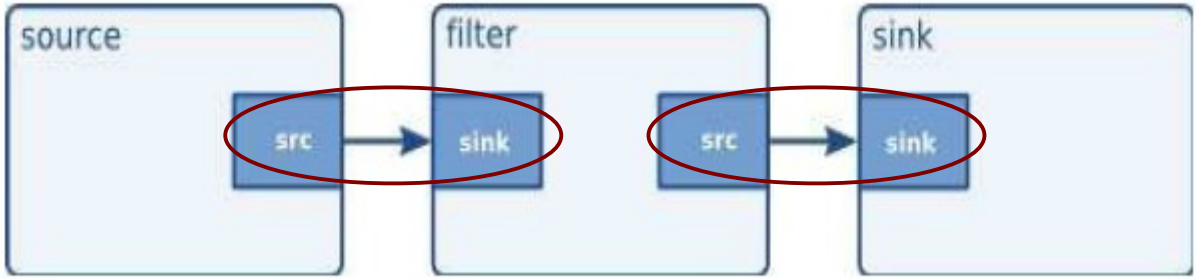
GStreamer Element

- After being created, an element will not actually perform any actions yet. You need to change elements state to make it do something.
- There are 4 states:



GStreamer Pads

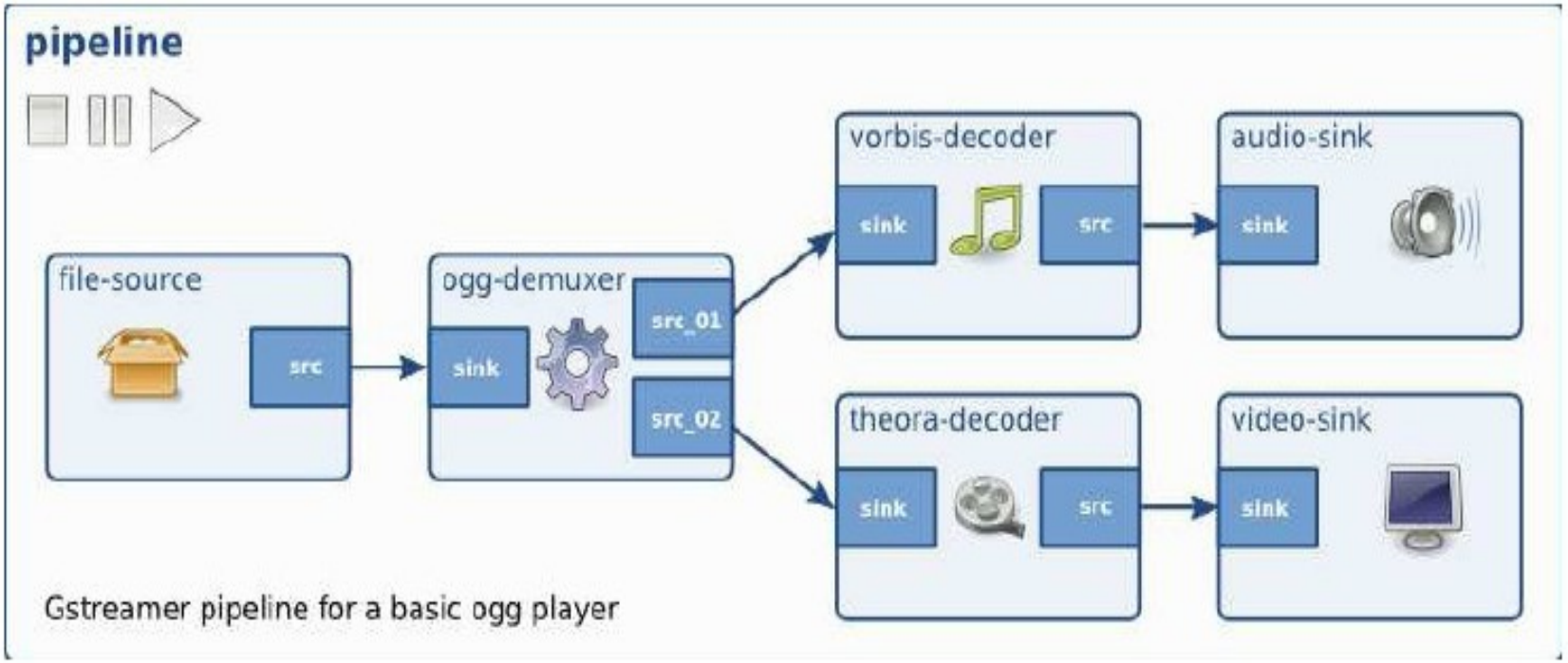
- Pad: element's input and output to connect other elements. They are used to negotiate links and data flow between elements in GStreamer.
- Source element and sink element just have one pad.
- Filter-like elements have two or more pads.
- Demuxer has 1 sink pad and several source pads, and a muxer has several sink pads and 1 source pad. Evidently, an element's source pad connects to a downstream (succeeding) element's sink pad.



- Capabilities of pad: describe the type of data streamed between 2 pads. Type of data is media type/information of stream is played back.

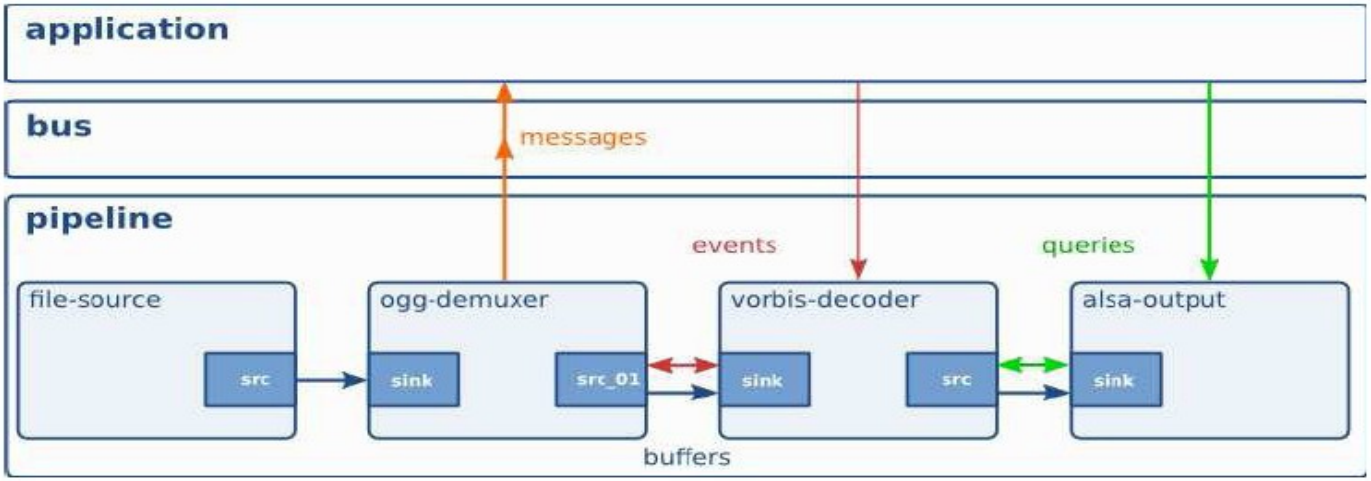
GStreamer Pipeline

- A pipeline is a central concept in GStreamer. It is a (usually simple) directed graph connecting elements for encoding/decoding.
- When pipeline is set to PAUSED or PLAYING state, data flow will start and media processing will take place.



Communication

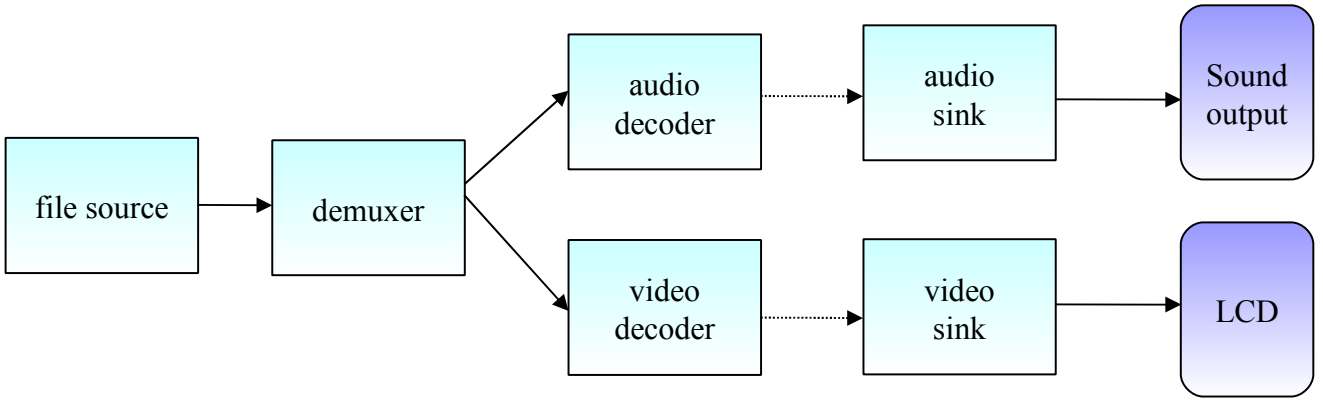
- GStreamer provides mechanism for communication b/t pipeline and applicaton and data exchange in pipeline
  - Buffer: pass streaming data between elements in the pipeline by downstream.
  - Message: posted by elements on pipeline's message bus. Messages are used to transmit information such as errors, tags, state changes, buffering state, redirects etc. from elements to the application
  - Queries: allow application request information from pipeline. Queries are always answered synchronously.
  - Event: sent between elements or from the application to elements. Events can travel upstream and downstreamcontain. It is used to control information.



Data flow of Pipeline for decoding

Data flow in Pipeline for decoding

- Basically, data flow of a pipeline for decoding as following

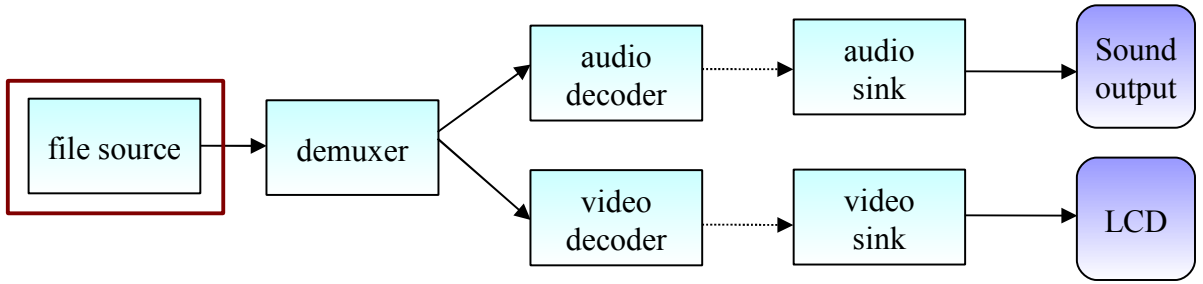


With:

- file source: input data
- demuxer: extract input file into audio part and video part
- audio decoder: decode for audio part
- audio sink: receive the output from audio decoder and send data to Sound output
- video decoder: decode for video part
- video sink: receive the output from video decoder and send data for displaying on LCD
- After receive output from decoder, zero or more elements used for converting to suitable data format then give output to audio/video sink

Data flow in Pipeline for decoding

- File source is considered as media container.

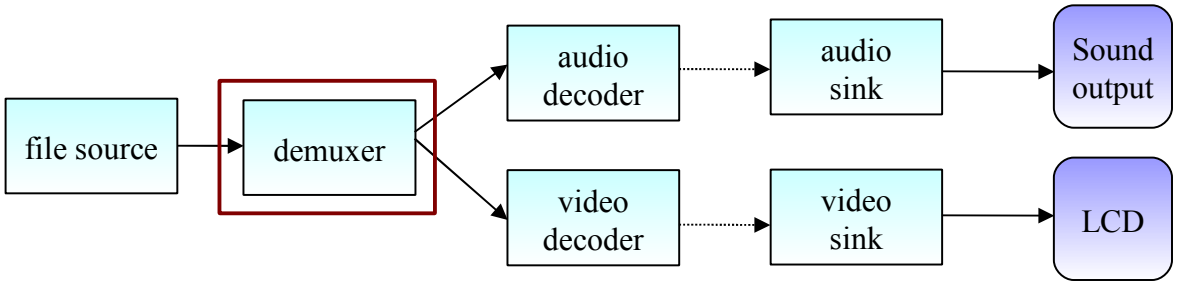


Example for some containers:

No.	Container	Video format supported	Audio format supported
1	MP4 (.mp4)	H.264/MPEG-4 AVC, MPEG-1, MPEG-2, WMV	AAC, MP3, AC-3, WMA
2	QuickTime (.mov, .qt)	H.264/MPEG-4 AVC, MPEG-1, MPEG-2, WMV	AAC, MP3, AC-3, WMA
3	AVI (.avi)	MPEG-1, MPEG-2, WMV	AAC, MP3, WMA
4	3GP (.3gp)	H.263, H.264/MPEG-4 AVC	AAC
5	OGG (.ogg)	Theora	MP3, Vorbis

Data flow in Pipeline for decoding

- Demux is used to extract the file source to video and audio stream

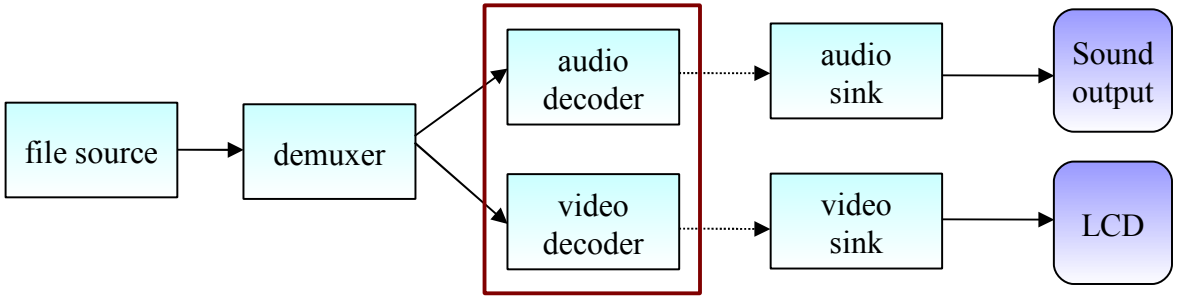


Example for demux elements

No.	Container	Demuxer supported	Note
1	MP4 (.mp4)	qtdemux	demuxer outputs audio/video format supported in container
2	QuickTime (.mov, .qt)	qtdemux	ditto
3	AVI (.avi)	avidemux	ditto
4	3GP (.3gp)	qtdemux	ditto
5	OGG (.ogg)	oggdemux	ditto

Data flow in Pipeline for decoding

- Decoder is used to decode audio/video stream.

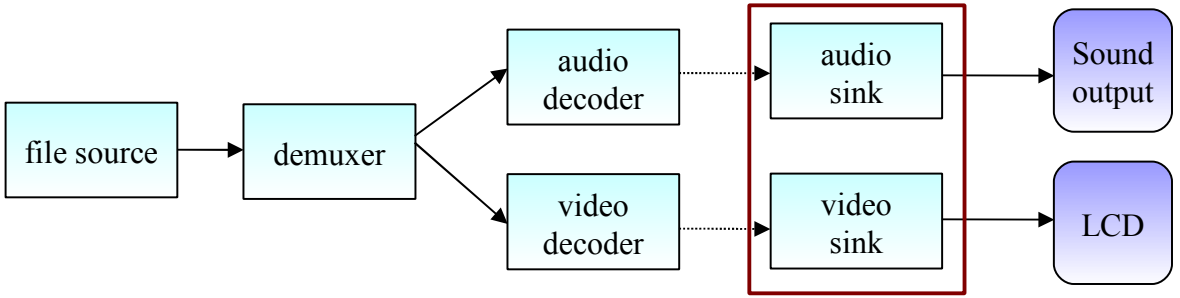


Example for decoder elements

No.	Container	Video decoder supported	Audio decoder supported
1	**MP4 (.mp4)**	**omx_h264dec, ffdec_h264, ffdec_mpeg4, ffdec_mpegvideo, ffdec_mpeg2video, ffdec_wmv3, ffdec_wmv2, ffdec_wmv1**	**omx_aacdec, ffdec_aac, ffdec_ac3, ffdec_mp3**
2	**QuickTime (.mov, .qt)**	**omx_h264dec, ffdec_h264, ffdec_mpeg4, ffdec_mpegvideo, ffdec_mpeg2video, ffdec_wmv3, ffdec_wmv2, ffdec_wmv1**	**omx_aacdec, ffdec_aac, ffdec_ac3, ffdec_mp3**
3	AVI (.avi)	decodebin, decodebin2, ffdec_mpegvideo, ffdec_mpeg2video, ffdec_wmv3, ffdec_wmv2, ffdec_wmv1	decodebin, decodebin2
4	3GP (.3gp)	omx_h264dec, ffdec_h264, ffdec_h263	omx_aacdec, ffdec_aac
5	OGG (.ogg)	theoradec	vorbisdec

Data flow in Pipeline for decoding

Audio/video sink are used for playing audio to Sound output and displaying video on LCD

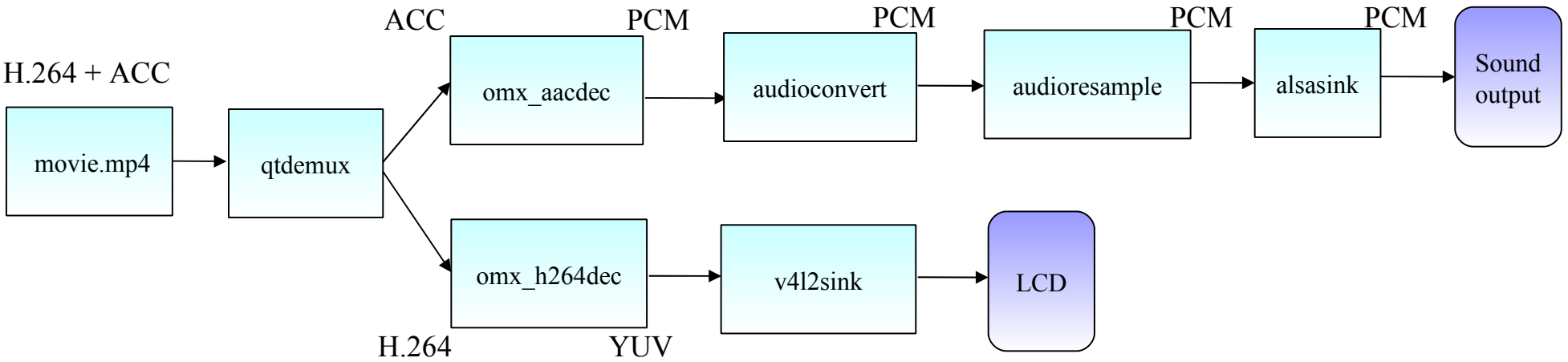


Example for sink elements

No.	Audio sink	Video sink	Other sink
1	autoaudiosink: detect audio automatically	autovideosink: detect video automatically	filesink: store data stream in a file
2	alsasink: output to sound card via ALSA	fbdevsink: support RGB format	udpsink: send data over the network via UDP
3	…	v4l2sink: support YUV format	…

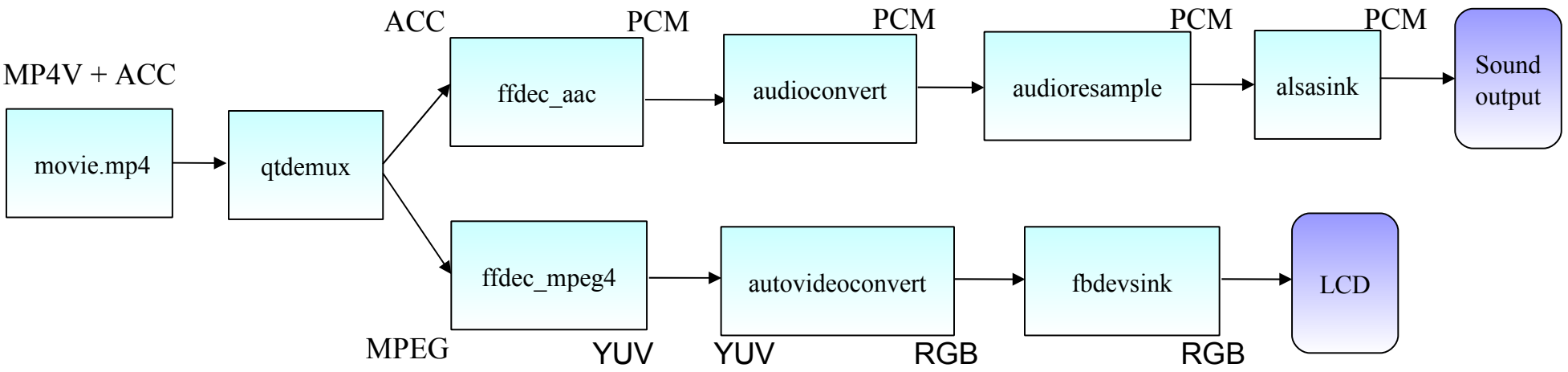
Data flow in Pipeline for decoding - Example

- Following is one example pipeline for decoding one movie file .mp4



Data flow in Pipeline for decoding - Example

- Following is one example pipeline for decoding one movie file .mp4



Renesas Design Vietnam Co., Ltd.