Onyx OpenMAX AL

This document describes the implementation of Onyx Playback Engine; API provided by the engine and also lists current status of implementation and open issues.

|  |  |  |
| --- | --- | --- |
| Revision | Date | Modifications |
| 1.00 | 04/19/2013 | Initial document |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Contents

[Overview 4](#_Toc366954015)

[ONYX OpenMAX AL Framework 4](#_Toc366954016)

[ASI/UDP Playback Engine 5](#_Toc366954017)

[OpenMAX AL API Wrapper 5](#_Toc366954018)

[Demux 5](#_Toc366954019)

[OMX Video Decode Chain 5](#_Toc366954020)

[OMX Video Audio Decode Chain 5](#_Toc366954021)

[Decode Chain Buffering Scheme 7](#_Toc366954022)

[SDI Capture Engine 8](#_Toc366954023)

[OpenMAX AL API Wrapper 8](#_Toc366954024)

[OMX Video Capture and Capture Decode Chains 8](#_Toc366954025)

[Module Dependency Diagram 9](#_Toc366954026)

[ASI OpenMAX AL API 10](#_Toc366954027)

[List of interfaces supported 10](#_Toc366954028)

[xaEngineItf::CreateMediaPlayer 10](#_Toc366954029)

[Configuration Extensions 11](#_Toc366954030)

[Configuration File 12](#_Toc366954031)

[Display Section 13](#_Toc366954032)

[Decoder Section 13](#_Toc366954033)

[Debug Section 14](#_Toc366954034)

[Open Issues 15](#_Toc366954035)

**References:**

1. OpenMAX Application Layer Application Program Interface Specification:   
   <http://www.khronos.org/registry/omxal/specs/OpenMAX_AL_1_0_1_Specification.pdf>
2. ISO/IEC Specification 13818-1 : Information technology — Generic coding of moving pictures and associated audio information: Systems

# Overview

This section gives the overview of playback/capture application framework for DVM based on OpenMAX AL Specification. It also presents an overview of design/implementation details of ASI, UDP and SDI Playback Engine.

## ONYX OpenMAX AL Framework

# ASI/UDP Playback Engine

\*For UDP ASI Reader/Driver is replaced with udp socket reader. All other components remain same.

ASI Playback Engine consists of the following components:

## OpenMAX AL API Wrapper

This module is wrapper for OMX IL Decode and Display chain. It provides high level API based on OMX AL to the application. The API supports configuration and starting/stopping of the chain.

## Demux

This module based on xport open source transport stream demuxer. The demuxer provides the API to select the PCR, video and audio PIDs. PCR is used to generate clock for AV Synchronization. The demux can also parse the PMT and can identify the PCR, video and PIDs. This mode can be used for debugging.

## OMX Video Decode Chain

This module creates and configures the decode display chain. This chain contains components based on TI’s OpenMAX IL implementation that run M3 media processors. An OMX component is wrapped in an object that maintains a local context and logic for pipe based communication with other objects.

## OMX Video Audio Decode Chain

This module creates and configures the audio playback chain. This chain contains components based on TI’s OpenMAX IL implementation that run C674 DSP. An OMX component is wrapped in an object that maintains a local context and logic for pipe based communication with other objects. The decoded audio is resampled (not implemented yet) and played using ALSA subsystem.

# Decode Chain Buffering Scheme

\* This component is not supported yet.

\*\*Before file reader, there is an additional buffer of 1MB in the FPGA. In the normal operation, this buffer should be near empty.

Buffers are allocated on the output port of the component except for Decoder, where buffers are allocated for both input and output port.

# SDI Capture Engine

SDI Capture Engine consists of the following components:

## OpenMAX AL API Wrapper

This module is wrapper for OMX IL Capture Display chain. It provides high level API based on OMX AL to the application. The API supports configuration and starting/stopping of the chain.

## OMX Video Capture and Capture Decode Chains

This module creates and configures the capture display chain. An object created in this module to represent by a OMX component. Each object contains local context, pipe based communication with other objects and a thread for circulating buffers.

# Module Dependency Diagram

The following gives an overview of partitioning of application code in to various implementation files.

dec\_omxal

dec\_main

Cap\_main

adec\_ilclient

vdec\_ilclient

acap\_ilclient

vcap\_ilclient

Shared sub modules  
strmconn, dec\_clock, minini, dbglog, dec\_platform\_utils

# ASI OpenMAX AL API

## List of interfaces supported

The following table lists the OpenMAX AL interfaces from the decoder module.

|  |  |  |
| --- | --- | --- |
| **Interface** | **Description** | **Remarks** |
| xaCreateEngine | Main Entry | Application uses this to access xaEngineItf. |
| xaEngineItf | Supported Methods:  CreateMediaPlayer | This interface is create playback session for a source stream and obtain xaPlayItf. |
| xaPlayItf | Supported Methods:  SetPlayState | This interface is used to set the playback state. |
| xaConfigExtensionsItf | Supported Methods:  SetConfiguration  GetConfiguration | This interface is used to configure playback parameters. |

## xaEngineItf::CreateMediaPlayer

This method can be used to create a media playback session. The following table lists the data source supported. The following structure is used to pass the data source information.

typedef struct XADataSource\_ {

void \* pLocator;

void \* pFormat;

} XADataSource;

The field pLocator in the XADataSource should point to the following structure.

typedef struct XADataLocator\_URI\_ {

XAuint32 locatorType;

const XAchar \* pURI;

} XADataLocator\_URI;

The field pFormat in the XADataSource should point to the following structure.

typedef struct XADataFormat\_MIME\_ {

XAuint32 formatType;

const XAchar \* pMimeType;

XAuint32 containerType;

} XADataFormat\_MIME;

|  |  |
| --- | --- |
| **Data Source** | **Description** |
| Multicast UDP | locatorType: = XA\_DATALOCATOR\_URI  pURI = <multicast address> Example: udp://224.1.1.8:1234  formatType = XA\_DATAFORMAT\_MIME  pMimeType = “mp2t”  containerType = XA\_CONTAINERTYPE\_MPEG\_TS |
| Unicast UDP | locatorType: = XA\_DATALOCATOR\_URI  pURI = <receive port address> Example: udp://:1234  formatType = XA\_DATAFORMAT\_MIME  pMimeType = “mp2t”  containerType = XA\_CONTAINERTYPE\_MPEG\_TS |

## Configuration Extensions

DVM Specific playback parameters are set by the application using OpenMAX AL configuration extensions. This interface includes supplying the parameter as a key-value pair. The keys is an ascii string and the value is generally is a pointer to a structure or integer value.

XAresult SetConfiguration (

XAConfigExtensionsItf self,

const XAchar \*pConfigKey,

XAuint32 valueSize,

const void \*pConfigValue

);

XAresult GetConfiguration (

XAConfigExtensionsItf self,

const XAchar \*pConfigKey,

XAuint32 \*pValueSize,

void \*pConfigValue

);

|  |  |  |
| --- | --- | --- |
| **Configuration Key** | **Configuration Value** | **Description/Remarks** |
| vid\_input\_param | XAVideoStreamInformation  typedef struct XAVideoStreamInformation\_ {  XAuint32 codecId;  XAuint32 width;  XAuint32 height;  XAuint32 frameRate;  XAuint32 bitRate;  XAmillisecond duration;  } XAVideoStreamInformation; | This API is used to supply video stream information. Only the following fields are used:   * codecid   + XA\_VIDEOCODEC\_AVC   + XA\_VIDEOCODEC\_MPEG2 * width, * height |
| aud\_input\_param | XAAudioStreamInformation  typedef struct XAAudioStreamInformation\_ {  XAuint32 codecId;  XAuint32 channels;  XAmilliHertz sampleRate;  XAuint32 bitRate;  XAchar langCountry[16];  XAmillisecond duration;  } XAAudioStreamInformation; | Audio stream information. Only the following parameters are used:   * codecid   + XA\_AUDIOCODEC\_AAC   + XA\_AUDIOCODEC\_PCM\*   + XA\_AUDIOCODEC\_AC3(10)\*\* * sampleRate |
| latency | 32 bit int | Maximum latency for displaying audio/video frame specified in milliseconds. This value specifies the decode path pipe-line delay. If this value is non-zero audio PCR in the TS stream is used to time the rendering of video and audio frames. A/V Frames are rendered when PTS of the frame is equal to PCR + latency. |
| deinterlace | 32 bit int | This API is used to specify whether video needs to be deinterlaced.  0 - Do not deinterlace  1 – Deinterlace |
| demux\_pcr\_pid | 32 bit int | Supplies PCR PID Value of the required program |
| demux\_aud\_pid | 32 bit int | Supplies audio PID Value of the required program |
| demux\_vid\_pid | 32 bit int | Supplies video PID Value of the required program |
| demux\_select\_prog | 32 bit int | 0 - demux\_pcr\_pid, demux\_aud\_pid and demux\_vid\_pid supplies PID values.  1 - demux\_pcr\_pid specifies program number, demux\_aud\_pid specifies audio channel number and demux\_vid\_pid supplies video channel number. PID values corresponding to the program and channels are selected by the demux. This used only for debug/testing. |
| version | 32 bit int | (major << 16 | minor) |

\* XA\_AUDIOCODEC\_PCM is supported only on SDI.

\* XA\_AUDIOCODEC\_AC3 is not defined in OpenMAX AL header files. The AL client application may define it locally and set it to 10.

# Configuration File

The decode/capture chains are configured by setting the various parameters at compile time are at run time (by the application). However some of the parameters can also be configured through a configuration file /etc/onyx.conf. The parameters supplied by the application override the parameter supplied through configuration file. If there is no configuration file present or parameters are not supplied then default values set at compile time are used.

The configuration file contains sections and each section contains one or more name and value pairs as shown below.

[display]

width=480

height =272

clock=9500

hfp=2

hbp=2

hsync=41

vfp=3

vbp=3

vsync=10

[decoder]

DECODER\_INPUT\_BUFFER\_COUNT=10

DECODER\_OUTPUT\_BUFFER\_COUNT=10

SCALAR\_OUTPUT\_BUFFER\_COUNT=10

DECODER\_MAX\_FRAME\_RATE=60

VSYNC\_MAX\_WAIT\_RUNNING=30000

VSYNC\_MAX\_WAIT\_STARTUP=500000

[debug]

A8\_DEBUG\_LEVEL=3

M3\_DEBUG\_LEVEL=1

## Display Section

This section can be used to configure the display parameters.

|  |  |  |
| --- | --- | --- |
| key | Description | Default Value |
| width | Width in pixels | 480 |
| height | Height in pixels | 272 |
| clock | Clock in kHz | 9500 |
| hfp | Horizontal Front Porch | 2 |
| hbp | Horizontal Back Porch | 2 |
| hsync | Horizontal Sync | 41 |
| vfp | Vertical Front Porch |  |
| vbp | Vertical Back Porch |  |
| vsync | Vertical Sync |  |

## Decoder Section

This section can be used to configure the display parameters.

|  |  |  |
| --- | --- | --- |
| key | Description | Default Value |
| DECODER\_INPUT\_BUFFER\_COUNT | Number of decoder input buffers | 10 |
| DECODER\_OUTPUT\_BUFFER\_COUNT | Number of decoder output buffers | 10 for H.264 8 for MPEG2 |
| SCALAR\_OUTPUT\_BUFFER\_COUNT | Number of scaler output buffers | 10 |
| DECODER\_MAX\_FRAME\_RATE | Decoder fram rate | 60 |
| VSYNC\_MAX\_WAIT\_RUNNING | Max running wait time in micro seconds | 30000 |
| VSYNC\_MAX\_WAIT\_STARTUP | Max startup wait time in micro seconds | 500000 |

## Debug Section

This section can be used set the console debug log levels for Linux(A8) and M3.

|  |  |  |
| --- | --- | --- |
| key | Description | Default Value |
| A8\_DEBUG\_LEVEL | One of the values specified below.  0 - DBGLVL\_ERROR  1 - DBGLVL\_WARN  2 - DBGLVL\_SETUP  3 - DBGLVL\_STAT  4 - DBGLVL\_TRACE  5 - DBGLVL\_FNTRACE  6 - DBGLVL\_FRAME  7 - DBGLVL\_PACKET  8 - DBGLVL\_WAITLOOP | 2 |
| M3\_DEBUG\_LEVEL | Bitwise or of following values  1 - OMX\_DEBUG\_LEVEL1  2 - OMX\_DEBUG\_LEVEL2  4 - OMX\_DEBUG\_LEVEL3  8 - OMX\_DEBUG\_LEVEL4  16- OMX\_DEBUG\_LEVEL5 | 1 |

# Open Issues

|  |  |  |
| --- | --- | --- |
|  |  |  |