Multimedia Codecs on i.MX

Sridharan Subramanian
Senior Product Manager – Software and Platforms
This session will present an overview of optimized multimedia Linux® and WinCE® Codec offerings across all i.MX platforms (includes video processing unit as well as SW Codecs).
Agenda

► Value Proposition

► Codecs Basics

► i.MX Multimedia Codecs

► i.MX Multimedia Approach and Methodologies

► Codecs Validation

► Codecs Performance

► Multimedia Frameworks
Value Proposition
► Multimedia Codecs are a key platform feature for enabling Multimedia applications

• Enables differentiated and accelerated product development

• Software Codec Packages (libraries, parsers, wrappers, docs) that support multimedia use cases on i.MX ARM cores

• Video Processing Unit hardware accelerated Codec packages that leverage the i.MX dedicated video accelerator block
Codecs Basics
Software Stack Architecture

End customer UI/MMI Application
- Internet browser
- DVB-H
- DVB-T
- Camera/Player/VT
- VoIP/V2IP
- IM

Multimedia and Applications Framework
- Standard Wrappers (Gst, Dshow, OMX)
- Codecs API
- Codecs
- OS abstraction layer
- BSP (Drivers)
- OS (Linux, Windows CE, RTOS)
- Hardware

Multimedia Infrastructure/Services
i.MX Optimized Multimedia Codecs

Key Features

► Comprehensive suite of optimized codecs (~40+ Audio/Video/Image Codecs)
► Highly optimized software that is coded by Freescale processor experts
► Consistent application programming interface (API) and frameworks across all software packages including OpenMAX support
► Codec APIs have been optimized from system design perspective and achieve optimal system performance along with related middleware wrappers
► Supplemented with Freescale development tools, sample test streams and documentation

Codec Software Packages include:

► Codec libraries with a standard C-callable API
► Gstreamer/DShow/OMX plugins that provide an API layer between the multimedia framework and the codec library
► Audio/video file containers (parsers) that support popular multimedia content, such as .aac, .avi, .asf, .mp3 and .mp4 files
► Bundle of Freescale audio/video sample test streams
► Complete documentation, including API documentation, release notes and data sheets
## i.MX Platforms and Multimedia Capability

<table>
<thead>
<tr>
<th>SoC</th>
<th>ARMv5 Codecs</th>
<th>ARMv6 Codecs</th>
<th>Neon Codecs</th>
<th>Hardware Codecs</th>
</tr>
</thead>
<tbody>
<tr>
<td>MX31</td>
<td>Yes</td>
<td>Yes</td>
<td>-</td>
<td>MPEG4-SP encode (VGA)</td>
</tr>
<tr>
<td>MX27</td>
<td>Yes</td>
<td>-</td>
<td>-</td>
<td>MPEG4, H.264 decode/encode (D1)</td>
</tr>
<tr>
<td>MX37</td>
<td>Yes</td>
<td>Yes</td>
<td>-</td>
<td>MPEG4, H.264, VC1 decode (D1)</td>
</tr>
<tr>
<td>MX35</td>
<td>Yes</td>
<td>Yes</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MX25</td>
<td>Yes</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MX51</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>All video codecs (720p decode, D1 encode)</td>
</tr>
</tbody>
</table>
# Codecs Portfolio

<table>
<thead>
<tr>
<th>Video</th>
<th>Audio</th>
<th>Speech</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPEG4 SP/ASP Encoder/Decoder *</td>
<td>WMA10 Decoder (Std, Pro, Lossless)</td>
<td>G.726 Encoder/Decoder</td>
<td>GIF Decoder</td>
</tr>
<tr>
<td>H.263 Baseline Encoder/Decoder *</td>
<td>MP3 Encoder/Decoder</td>
<td>G.723.1 Encoder/Decoder</td>
<td>PNG Decoder</td>
</tr>
<tr>
<td>WMV9/VC-1 SP/MP/AP Decoder *</td>
<td>AAC-LC Decoder</td>
<td>AMR-NB Encoder/Decoder</td>
<td>BMP Encoder/Decoder</td>
</tr>
<tr>
<td>Real Video Decoder *</td>
<td>AACPlus Enhanced Decoder</td>
<td>AMR-WB Encoder/Decoder</td>
<td>WBMP Decoder</td>
</tr>
<tr>
<td>H.264 BP/MP Encoder/Decoder *</td>
<td>Real Audio 8 Decoder</td>
<td>G.711 Encoder/Decoder</td>
<td>JPEG Encoder/Decoder *</td>
</tr>
<tr>
<td>MPEG2 MP Decoder/Encoder *</td>
<td>SBC Encode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DivX Decoder *</td>
<td>WMA Encode</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Note: * indicates VPU accelerated or s/w codecs
## Audio/Video Effects Portfolio

<table>
<thead>
<tr>
<th>Video effects</th>
<th>Audio effects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deinterlacing</td>
<td>Bass boost</td>
</tr>
<tr>
<td>Denoising</td>
<td>Modal equalizer</td>
</tr>
<tr>
<td>Error concealment</td>
<td></td>
</tr>
</tbody>
</table>
i.MX Multimedia Approach
i.MX Multimedia Approach

► Software Optimization

- Software optimizations enable high quality multimedia solutions – algorithm choice, fast algorithmic implementations, hotspot optimizations enable many use cases to be met in software
- Platform aware software – optimized for L1/L2 hierarchy, Cache settings.
- Leverage multimedia enabled ARM core ISA
- Migration of common ARM core and multimedia acceleration architectures across the Freescale i.MX multimedia applications processors maximizes software re-use

► Selective Hardware-Software Partitioning

- Selective use of partial accelerators - optimal hardware / software partitioning

► Smart Hardware Acceleration

- Through system performance analysis, major multimedia performance and power hotspots are identified and accelerated in hardware
Codecs Features

- Full featured, Standard Compliant, Robust Multimedia codecs
- Reference C code based on innovative algorithms
- Simple Interfaces with well designed APIs
- Extensive Debug and Error reporting capability
- Portable, Re-entrant implementation suitable for Multi-threaded, Real time operation
- Hand assembly optimized for ARM11 (v6) architecture
- Extensive use of SIMD instructions, Cache and Pipeline optimizations
- Tight integration of FSL IPU Hardware block with Video codecs
- Well designed Error Resilience and Error Concealment result in high performance even for lossy wireless links
- Extensively tested, verified and profiled on ARMulator, CCM and ARM based HW platform
- Extensively tested for Inter-Operability and Conformance
Codec Development Flow

- Floating Point Algorithm – Performance, Conformance, Quality, Resilience
- Fixed Point C reference – Performance, Conformance, Quality, Resilience
- Fixed Point C reference – Re-entrancy, Multi-Threaded
- C level Optimizations
- C/Assembly/Accelerator Partitioning
- Assembly Optimizations
- Integration - C/Assembly/Accelerator
- Extensive Profiling and Benchmarking
- Unit and Release Testing
- Certification, Interoperability
Optimization Methodologies

► Hand assembly optimized for the specific ARM architecture

► Extensive use of SIMD/NEON instructions, Cache and Pipeline optimizations
  
  • Cache line sized access
  • Efficient Data Alignment
  • Data re-ordering
  • Multiple Loads/Stores
  • Packed data operations
  • Loop unrolling
  • Delayed execution
  • Efficient register usage
Hardware/Software Partitioning

► Selective HW/SW partitioning to enable efficient video playback and power usage

► Image processing unit (IPU) – For video processing outside the codecs

► Video processing unit (VPU) – For full and partial acceleration of certain video codecs
HW/SW Optimization – MPEG4/H264 Decoder

► IPU provides support for MPEG-4 Post-filtering (PF) and H.264 in-loop de-blocking

► IPU PF block offloads the de-blocking operation from the ARM1136JF-S core – a gain of almost 30% of decoder MHz

► Decoding happens sequentially on the ARM1136JF-S core; deblocking on the IPU at the end of decoding frees the ARM1136JF-S core to perform other tasks – Audio, MMFW, etc.
H.264 Decode Datapath

H.264 Decode + Post Processing Data Path

H.264 Encoder
Compressed Data

H.264 Decode (ARM11) → H.264 Inner Loop Deblock → Scaling → Color Space Conversion → Graphics Overlay → Rotation

Uncompressed Image
i.MX31 Image Processing Unit

H.264 Decode

Breakdown of C Optimized Codec and Post-Processing

- H.264 Decode: 28%
- Color Space Conversion: 19%
- Scaling: 26%
- Graphics Overlay: 10%
- Rotation: 7%
- H.264 Inner Loop Deblocking: 10%

 Portions of Codec and Post-Processing Handled by IPU Hardware Accelerator (outlined areas)

- H.264 Decode: 28%
- Color Space Conversion: 26%
- Rotation: 19%
- Graphics Overlay: 10%
- H.264 Inner Loop Deblocking: 10%
Visual Processing Unit Acceleration

► Use of dedicated HW accelerators to improve video performance

► Multi-standard video decoder

► Sub-blocks reuse for die area efficiency

► High parallelism for low operating frequency

► Built in support for rotation/mirroring

► Built in support for post-processing filtering (de-blocking and de-ringing)
Codecs Validation
Gstreamer Overview

Gstreamer is an open source multimedia application development framework in Linux that creates and links different components for playback and recording.

- Allows the construction of graphs of media handling components, ranging from simple Ogg/Vorbis playback to complex audio (mixing) and video (non-linear editing) processing.

- Applications can take advantage of advances in codec and filter technology transparently.

- Developers can add new codecs and filters by writing a simple plugin with a clean, generic interface.

- Supports elements for Decoders, Encoders, Demuxers (for splitting the audio and video into separate streams), Muxers (for merging the streams back together), Filter elements.
Audio video Playback software Architecture

FSL Video Playback Application

File Source Plug-in

File Demuxer plugin

•Pipeline bin

GST Queue

Audio Decode Plugin

Audio rendering plugin (osssink) (V4L)

GST Queue

Video Decode Plugin

Video rendering plugin (V4L)

Display

GST Core Library

•Gstreamer provided component

•FSL provided Component
GStreamer Plug-ins

- Codec and parsers are added to the GStreamer framework in the form of plug-ins
- The plug-in module encapsulates the codec so that it can be used by GStreamer
- Video pipeline optimization for fine playback
- V4L drivers; Gstreamer videosink – efficient buffer management
OpenMax

► Standard for common APIs for multimedia applications

► Allows silicon and software vendors to provide optimized codecs without a proprietary API

► Encapsulate high-performance multimedia related functionality. Hardware accelerator extensions are transparent to the multimedia application developer.

► Standardization allows reuse of applications across multiple platforms.
Linux OpenMax software stack

Application
- Rhythmbox
- amarok
- Totem
- Mplayer
- Customers Application

Framework
- Gstreamer
- Customers Application Framework

Integration
- Gst-openmax Stands right here
- Openmax IL
- Standard 1.0 and 1.1 available

Profile
- MP3 decoder
- Audio Sink
- Video sink
- H.264 decoder
- MPEG-4 decoder

Components
- Openmax IL Fsl mp3
- Openmax IL ALSA
- Openmax IL V4L
- Openmax IL VPU
Windows CE integration

- Codec and parsers are added to the Windows CE (WinCE) framework in the form of DShow/DMO plug-ins.
- Memory allocation
- Renderer; DirectDraw
- Trick modes/flush
- VC-1/WMV integration
WMP integration – Filter graph

1. Browsers/Applications
2. Media player control
3. COM interfaces
4. DirectShow filter graph manager
5. Media Source (internet or file system)
6. DirectX foundation
7. Media destination (audio/video hardware)

- Source filter (ASF, AVI, etc.)
- Transform filter
- Renderer filter
Transform Filter

VPU Codec → DShow Filter Class

API Calls

YUV 420

IPU/eMMA Post-Processing Block

RGB 565

Renderer

YUV 420

RGB 565
Q&A

Thank you for attending this presentation. We’ll now take a few moments to review the audience questions, and then we’ll begin the question and answer session.