Audio Subsystem Solutions for Consumer SoCs

Pieter van der Wolf

MPSoC 2011
July 4-8, 2011

Trends in audio

Internet-enabled devices
- Support many audio formats

Multi-channel audio content
- Support multi-channel audio formats

HD audio content
- High sampling rates, 24-bit precision

Sound processing functions
- Algorithms for improved audio quality

- Not a side activity on host processor
- Off-load to separate audio DSP
Outline

• Audio subsystem solutions
  – Building audio subsystems
  – Integrating audio subsystems
  – Reusing audio subsystems
• Conclusions

Audio processing
Use case Blu-ray playback

Mandatory: LPCM, DD, DTS
Opt: DTS-HD HRA, DD+
Opt: TrueHD and DTS-HD MA (5.1/192kHz or 7.1/96kHz)

Primary decode

DD+ (5.1/48kHz)
DTS Express (5.1/48kHz)

Secondary decode

Interactive audio from BD-J
7.1, PCM

BD-J decode

Source, Mix

PCM 2/5.1/7.1 192kHz

DAC

PCM 2/5.1 48kHz

Dolby or DTS encode

SPDIF

Selected

HDMI

DTV watch and record same channel

TS Demux

Decoding

7.1 ch

Down-mixing...

2 ch

Sound processing

DAC

LR

To speakers (with drivers)

SPDIF-out

HDMI

To receiver

Hard disk

Including programmable delays for A/V sync
DTV watch and record different channels

Including programmable delays for A/V sync

Building an audio solution from IP

- DSP
- Codecs
- Sound processing
- I2S
- SPDIF
- Analog front-end

- DMA
- Clocks
- Interrupts
- Reset
- Drivers
- OS / RTE
- Media streaming framework
- IPC
- Host integration
- Audio formats
- APIs
- Verification
- A/V sync
- Development platform
Alternative: audio subsystem

Complete Audio Solution:
- Software + Hardware
- Codecs + Sound Processing
- Digital + Analog Peripherals
- Host Plug-in Software
- Development Platform

Host integration

- SW plug-in on Host CPU with standard API
- Audio functions visible on Host
- Build and execute use cases on Host (incl. audio, video, etc.)
Host integration, SW plug-in

- SW plug-in implements small stubs for audio functions
- Audio functions can be created, started, stopped, etc.
- Audio functions can be hooked up in use cases
- Plug-in hides core crossings

Sample clocks

<table>
<thead>
<tr>
<th>Input/Output</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optical disc</td>
<td>HDD / SSD</td>
</tr>
<tr>
<td></td>
<td>No sample clock needed</td>
</tr>
<tr>
<td></td>
<td>• E.g. ripping</td>
</tr>
<tr>
<td></td>
<td>• No use of audio peripherals</td>
</tr>
<tr>
<td>I2S / SPDIF</td>
<td>Master</td>
</tr>
<tr>
<td></td>
<td>• E.g. playback from storage</td>
</tr>
<tr>
<td></td>
<td>• Generate sample clock(s)</td>
</tr>
<tr>
<td></td>
<td>• Use of audio peripherals</td>
</tr>
<tr>
<td>I2S / SPDIF</td>
<td>Slave to external clock</td>
</tr>
<tr>
<td></td>
<td>• E.g. playback from I2S / SPDIF</td>
</tr>
<tr>
<td></td>
<td>• Use external sample clock(s)</td>
</tr>
<tr>
<td>Broadcast</td>
<td>Slave to transmitted clock</td>
</tr>
<tr>
<td></td>
<td>• E.g. broadcast playback</td>
</tr>
<tr>
<td></td>
<td>• Reconstruct sample clock(s)</td>
</tr>
<tr>
<td></td>
<td>• Synchronize to transmitted clock</td>
</tr>
</tbody>
</table>
A/V synchronization (lip sync)

Conclusions

- Audio subsystems have added value over a “bag of IP”
  - HW-SW integration, Clocks, Host integration

- Audio subsystem reuse
  - Instances derived from configurable template
  - Just like configurable IP

- Audio subsystems from external suppliers
  - Reduce development effort
  - Free-up engineering resources
  - Reduce time to market