Codecs and Processors

You can’t choose one without the other

Patrick Willekens
Product Manager AUDIO
ARM
Agenda

- The codec selection
  - What do you want to make?
  - What do you need to make it?
  - What are your needs according to location in the value chain?
- The processor selection
- The codec vendor selection
- Conclusion
The codec selection

- What device do we need to make? (= list of base features)

- What are the market(ing) requirements? (= list of extra features)

- Translate these features to multimedia requirements portfolio
  - Speech
  - Audio
  - Graphics
  - Video
  - Imaging
  - Security
The codec selection

- **Speech**
  - Conversation (over 3G, WiFi…)
  - Recording
  - Recognition (speed dialling…)
  - Synthesis (Txt2speech, navigation)

- **Audio**
  - Music playback (MP3, AAC…)
  - Movie surround soundtracks
  - Streaming audio
  - Digital surround radio

- **Graphics**
  - Multi-player interactive games
  - Talking avatars
  - 3D menus
  - GPS

- **Video/imaging**
  - Video conferencing
  - Camera (10Mpixel, 3xzoom)
  - Camcorder (HD)
  - Video play (WMV9, MPEG4)
  - Streaming video
  - Interactive TV (MBMS)

- **Messaging**
  - MMS

- **Security**
  - e-wallet, banking…
The codec selection

- What do you need exactly?
  - Do you need specific codecs?
    - Software algorithm that executes a specific task
      - Uncompressed audio -> compressed audio
      - Speech with echo -> speech without echo
    - E.g. MP3 decoder, SBC encoder, acoustic echo canceller...
  - Do you need a complete processing engine?
    - Bundles all codecs required to perform a specific task
    - E.g. VoIP engine bundles all codecs required to make a VoIP voice or video call
  - Do you need an entire software stack?
    - An engine bundled with applications and user interface to create complete SW solution
    - E.g. VoIP phone bundles an engine with phone applications in a UI
The codec selection

- Software requirements typically originate from location of your product in the value chain

Picture provided as courtesy from Spirit DSP
The codec selection

- First layer: chip layer
  - Characteristics
    - Semiconductor manufacturer creates chip that is ‘application-ready’ by embedding standard or proprietary codecs
  - Needs
    - Algorithms (codecs, speech enhancement, telephony…) optimized for processor that is used to run the algorithm (ARM® CPU, DSP…)

Semiconductor chip vendors:
Create chip with embedded audio software
The codec selection

- Second layer: device layer
  - Characteristics
    - OEM/ODM creates entire hardware solution by integrating hardware components into a full device
  - Needs
    - SW engine that creates a complete solution on top of hardware
      - Need high voice, audio & video quality, tuned to HW design
      - Need real-time network optimization
      - Need voice & video synchronization

OEM/ODM developers: Hardware with optimized software engine
The codec selection

- Third layer: application layer
  - Characteristics
    - ISV creates application software that interfaces with available hardware
  - Needs
    - User-friendly API so they can concentrate on core functionality
The codec selection

- Fourth layer: service layer
  - Characteristics
    - Application developer and service provider allow simultaneous communication between multiple users (e.g. online gaming)
  - Needs
    - Complete wideband conferencing solution for 100s of connections with low hardware cost.

Service provider: Server solution
The codec selection

- Two examples of multi-layer software solutions

**Spirit DSP**
Both TeamSpirit® CPE Voice/Video Engine
And TeamSpirit™ Media Server
(application layer and UI optional)

**Trinity Convergence**
VeriCall® VoWiFi Phone Edition
(includes application layer and UI)
Agenda

- The codec selection
- The processor selection
  - Finding the right processor configuration
  - Processor selection
  - Codec performance for a given processor
- The codec vendor selection
- Conclusion
The processor selection

- Finding the right processor (1)
The processor selection

- Finding the right processor (2)
  - Scenario 1: All multimedia on main processor
    - Only one processor needed (less area)
    - But processor needs to be faster and danger of interrupted music/video, depending on processor load
  - Scenario 2: All multimedia on dedicated co-processor(s)
    - Uninterrupted processing, combined with intelligent power-down or run-slow for power savings
    - But overall efficiency might suffer under some use scenarios
  - Scenario 3: An intelligent partitioning
    - Find the power/performance bottlenecks and split the algorithm
    - E.g. for audio: filtering, copy protection, stream analysis, stream post-processing on main CPU and audio number crunching on dedicated coprocessor
The processor selection

- Example High-End Phone Apps Processor

[Diagram showing various components and connections, including:
- DMC (PL340/PL341)
- DDR/DDR2 SDRAM
- NOR
- NAND
- Camera Interface
- Video Encoder
- Video Decoder
- Mali200™/MaliGP2 or Mali55™
- AudioDE™ SS
- LCD Controller
- VGA Controller
- PL301 AMBA™ Matrix
- ARM1176JZF or Cortex-A8
- L1 Cache
- L2 Cache
- Boot ROM
- On-Chip SRAM
- GIC PL390
- Touch screen
- ARM Artisan Physical IP
- ARM IP
- Connected Community
- Customer/Third-party IP
- Secure/Aware
- ARM Developers’ Conference & Design Pavilion 2007]
The processor selection

- ARM® AudioDE™ audio processor (80 MHz@TSMC130G)
  - For ultra low-power audio/speech processing
  - Optimised variable length – LIW architecture

- ARMv4 architecture
  (e.g. ARM7TDMI-S™ processor: 133MHz@TSMC130G)
  - For low-cost audio processing

- ARMv5 architecture
  (e.g. ARM926EJ-S™ processor: 266MHz@TSMC130G)
  - Most popular all-purpose processor for audio processing
  - DSP instruction extensions and single cycle MAC
The processor selection

**ARMv6 architecture**
(e.g. ARM1136JF-S™ processor: 350MHz@TSMC130G)
- More powerful compatible alternative to ARMv5 architecture
- Addition of SIMD media instructions: 
  1.75X faster at media processing than ARMv5 processor
- Optional tightly integrated VFP (Vector Floating Point) unit

**ARMv7 architecture**
(e.g. Cortex™-A8 processor: 600MHz@TSMC130G)
- Next-generation processor with increased performance
  - Thumb-2 instruction set for greater performance, energy efficiency, and code density
  - NEON™ signal processing extensions in separate engine to accelerate media codecs such as H.264 and MP3
- 2 to 4x faster than ARM11xx processor for typical media applications
- Optimized codec availability still low but improving quickly
The processor selection

- Codec performance on a given processor depends on CPU architecture and instruction set
  - Required MHz (MIPS) is different
  - Memory requirement is different
    - Program memory
    - Data memory/cache memory
  - Impact on power/performance can be large

<table>
<thead>
<tr>
<th>Codec</th>
<th>ARM7 Peak/Avg MIPS</th>
<th>ARM7 Other Memory</th>
<th>ARM9 Peak/Avg MIPS</th>
<th>ARM9 Other Memory</th>
<th>ARM9E Peak/Avg MIPS</th>
<th>ARM9E Other Memory</th>
<th>ARM Developer</th>
<th>Other Memory</th>
<th>Other Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP3 decoder</td>
<td>25/19.5</td>
<td>19.7</td>
<td>19.5</td>
<td>22/17.5</td>
<td>19.7</td>
<td>19.5</td>
<td>10/7.5</td>
<td>17.5</td>
<td>18.9</td>
</tr>
<tr>
<td>SBC encoder</td>
<td>21.1</td>
<td>8.6</td>
<td>3.3</td>
<td>16.1</td>
<td>8.6</td>
<td>3.3</td>
<td>10.1</td>
<td>8.6</td>
<td>3.3</td>
</tr>
<tr>
<td>G.729AB codec</td>
<td>29.49</td>
<td>41.4</td>
<td>22.3</td>
<td>19.56</td>
<td>39.6</td>
<td>22.4</td>
<td>17</td>
<td>76</td>
<td>14.8</td>
</tr>
</tbody>
</table>

Dyn power 1/2
The processor selection

Key Technology
Additions by
Architecture Generation

- Jazelle-RCT
- VFP (v3)
- NEON™ Advanced SIMD
- Thumb-2 (required)
- Thumb®-2 (option)
- TrustZone™
- SIMD
- VFP9-S (v2) (option)
- VFP11 (v2) (option)
- Jazelle™
- DSP extensions (E)I
- Thumb ISA (T)
- ARM ISA
- e.g. ARM7TDMI-S
- e.g. ARM926EJ-S
- e.g. ARM1176JZ-S
- e.g. ARM Cortex A8
- v4T
- v5TE
- v6
- V7 (A/R)

Thumb as subset of Thumb-2

ARM Developers’ Conference & Design Pavilion 2007
The processor selection

- Most CPUs can do all speech/audio codecs
- It’s the combination of codecs that determines the selection

![Diagram showing processor selection over time with categories: AudioDE, ARMv4 (ARM7), ARMv5 (ARM9E), ARMv6 (ARM11), ARMv7 (Cortex A8), and ARMv8. The years 2005 to 2008 are indicated with performance bars for each category.]

Audio codecs for home entertainment/theatres/gaming
Audio and speech codecs for non-portable audio applications
Audio and speech codecs for portable audio applications
Agenda

- The codec selection
- The processor selection
- The codec vendor selection
  - Make a short list
  - Codec comparison
  - Codec development
- Conclusion
The codec vendor selection

- Make a list of vendors that offer what you need
  - A great starting place: ARM Connected Community
    http://www.arm.com/community
    (filter companies by product type and look at software categories, e.g. see audio list)
  - Ask your ARM sales manager to contact their colleagues in product marketing, segment marketing or technical marketing
  - And the www of course…

- Compare the vendors
  - Look at all codec criteria
    - MCPS (Mega Cycles Per Second) needed
    - Data memory needed
    - Program memory needed
Codec comparison

- How to specify performance? (1)
  - Specification of measurement environment
    - Software: RealView ARMulator Instruction Set Simulator (RVISS)
      (part of RealView Development Suite) + version (eg. 2.2)
    - Hardware: eg TI DMxx SoC with ARM926EJ-S
  - Specification of cache sizes and memory bus clock rate (MCCFG)
    - Eg for ARMulator: 16Kb of I-Cache and 16 Kb of D-Cache,
      MCCFG setting (clock ratio between core and bus, 1 for 0-wait)
  - Specification of stream details
    - Peak MCPS (= the worst case frame in worst case stream)
    - Average MCPS (= the average frame for a typical stream)
      - Eg MP3 decoding on 128kbps 44.1KHz encoded stream or
        320kbps 48KHz encoded stream?
      - Which ‘typical stream’? Stream contents can influence the results.
Codec comparison

How to specify performance? (2)

- Specification of **MCPS details**
  - **Instruction Cycles**
    - Nrof instructions (assuming every instruction = one cycle)
    - An indication for inherent complexity of the algorithm
  - **Core Cycles**
    - Nrof cycles taken by all instructions (some instructions take more than one cycle like load, store, pipeline stalls etc) with 0-wait state memory
    - If core>>instruction, algorithm is data intensive : performance impact when high access-latency memory system

- **Bus (Total) Cycles**
  - MCPS based on memory bus cycles
  - Indication how algorithm would exploit the cache memory

<table>
<thead>
<tr>
<th>Processor</th>
<th>MCPS</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>Instruction</td>
<td>Core</td>
</tr>
<tr>
<td>ARM926EJ-s</td>
<td>9.31</td>
<td>17.34</td>
<td>23.59</td>
</tr>
<tr>
<td>ARM1136J-s</td>
<td>8.34</td>
<td>16.5</td>
<td>25.03</td>
</tr>
</tbody>
</table>
Codec comparison

- How to specify memory consumption?
  - At least a split between
    - Program Memory (Kb)
    - Constant Memory (Kb)
    - RAM Data (Kb)
  - Preferably also distinction between
    - Persistent memory
      - data structures that need to be retained between program executions
    - Scratch memory
      - Temporary data structures that don’t need to be kept between program executions

<table>
<thead>
<tr>
<th>Platform</th>
<th>ARM9E</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average MIPS</td>
<td>10</td>
</tr>
<tr>
<td>Peak MIPS</td>
<td>12</td>
</tr>
<tr>
<td>Program Memory, KB</td>
<td>35</td>
</tr>
<tr>
<td>Const Memory, KB</td>
<td>44</td>
</tr>
<tr>
<td>Persistent Memory, KB per channel</td>
<td>28</td>
</tr>
<tr>
<td>Scratch Memory, KB</td>
<td>10</td>
</tr>
</tbody>
</table>
## Codec comparison

- **An example**
  
  **MP3 decoder for ARM9E from 5 different vendors**

<table>
<thead>
<tr>
<th>Vendor</th>
<th>Peak/Avg MIPS</th>
<th>Program Memory</th>
<th>ROM data</th>
<th>RAM data</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vendor1</td>
<td>10/7.5</td>
<td>17.5</td>
<td>6.6</td>
<td>12.3</td>
<td>48KHz@320Kbps/44.1KHz@128Kbps</td>
</tr>
<tr>
<td>Vendor2</td>
<td>20</td>
<td>60</td>
<td></td>
<td>20</td>
<td>44.1KHz@160Kbps</td>
</tr>
<tr>
<td>Vendor3</td>
<td>21.8/14.6</td>
<td>33</td>
<td>10.9</td>
<td>19.3</td>
<td>44.1KHz@128Kbps</td>
</tr>
<tr>
<td>Vendor4</td>
<td>17</td>
<td>21.2</td>
<td>6.2</td>
<td>22.4</td>
<td>48KHz@320Kbps</td>
</tr>
<tr>
<td>Vendor5</td>
<td>13.8</td>
<td>39.3</td>
<td></td>
<td>25</td>
<td>44.1KHz@128Kbps</td>
</tr>
</tbody>
</table>

- **MCPS**: Factor 2 difference between average and optimal implementation
- **PMEM**: Factor 4 difference between average and optimal implementation
- **DATA**: Factor 2 difference between average and optimal implementation

Encoded streams are different!
Codec comparison

- Why this large difference?
  - How is the code programmed?
    - Is code written in C code and processed with C compiler?
    - Is code hand-optimized in assembly code?
      (can cause a factor of 2 for critical loops)
  - What was the code optimization target?
    - Minimum number of MIPS?
      (e.g. by exploiting available hardware to maximum)
    - Minimum program memory size?
      (e.g. by using subroutines as much as possible)
    - Minimum data consumption?
      (e.g. by analysing lifetimes of variables)
Codec comparison

- Why this large difference?
  - Codec performance/memory details are not specified the same way

- Not all vendors have the same algorithm knowledge
  - Reference code is typically not suited for implementation
    - E.g. 32-bit floating point C code while hardware is 24-bit
  - Implementation code should take into account
    - Available processing units (e.g. 1 MAC?, 2 MACs?)
    - Available instruction set (e.g. SIMD instructions ?)
    - Available data widths (16bit, 24bit, double precision widths…)
  - Rewriting algorithms requires deep DSP knowledge
    - E.g. bit-true exactness when going from 32 to 24 bit?
Codec comparison

- Why this large difference?
  - Not all vendors port code to new platforms
    - ARM9E code compiles for ARM11 as well but does not use the new features of ARM11 (SIMD Media Instructions, 64-bit load and store, faster multiply & MAC with SMMLS and SMMLA instructions …)

  ![Diagram showing performance comparison between ARM9E and ARM11]

  - the effect of ‘recompilation’, typically at least 10% performance improvement should be seen when code is optimized for ARM11

- But sometimes it doesn’t really matter of course…

ARM

ARM Developers’ Conference & Design Pavilion 2007
Codec development with OpenMAX DL

- **ARM released optimized source code versions of library functions for:**
  - AAC decoder (audio)
  - H.264 decoder (video)
- For ARM11 and Cortex-A8 (with Neon)

**OpenMAX AL**
- "Application Level"
- Media Application Portability
- Applications programmed using cross-vendor interfaces

**OpenMAX IL**
- "Integration Level"
- Media Graph Portability
- Integrate media networks using standard interconnect protocols

**OpenMAX DL**
- "Development Level"
- Media Component Portability
- Develop portable media components using low-level media APIs

Media infrastructure stack
Conclusion: the 5 P’s

- **Portfolio**
  - Defining the software needs is a first challenge

- **Performance**
  - Calculating the required performance for the portfolio of software gives a first indication of the preferred processor

- **Power**
  - The power budget will narrow down the processor possibilities

- **Processor**
  - The power-performance-area requirements, combined with commercial considerations will further determine processor choice

- **Partner**
  - Getting the software from the right vendor will complete the cycle