Linux-Based Low-Latency Multichannel Audio System CTAG face2|4

...

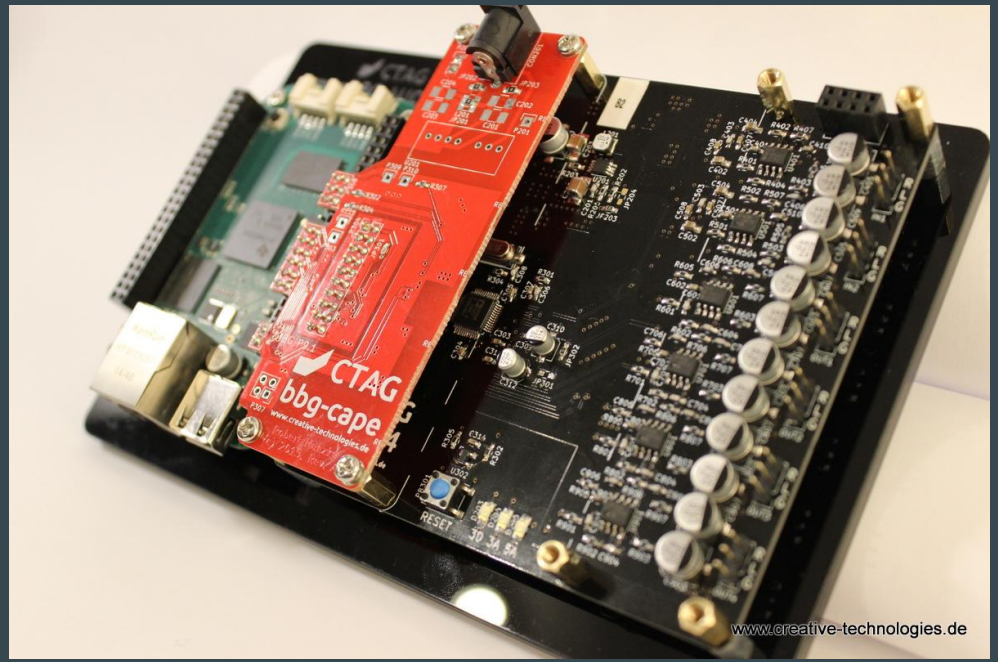
Henrik Langer, B.Sc.
Robert Manzke, Ph. D.

Department of Computer Science and Electrical Engineering
Kiel University of Applied Sciences

Contents

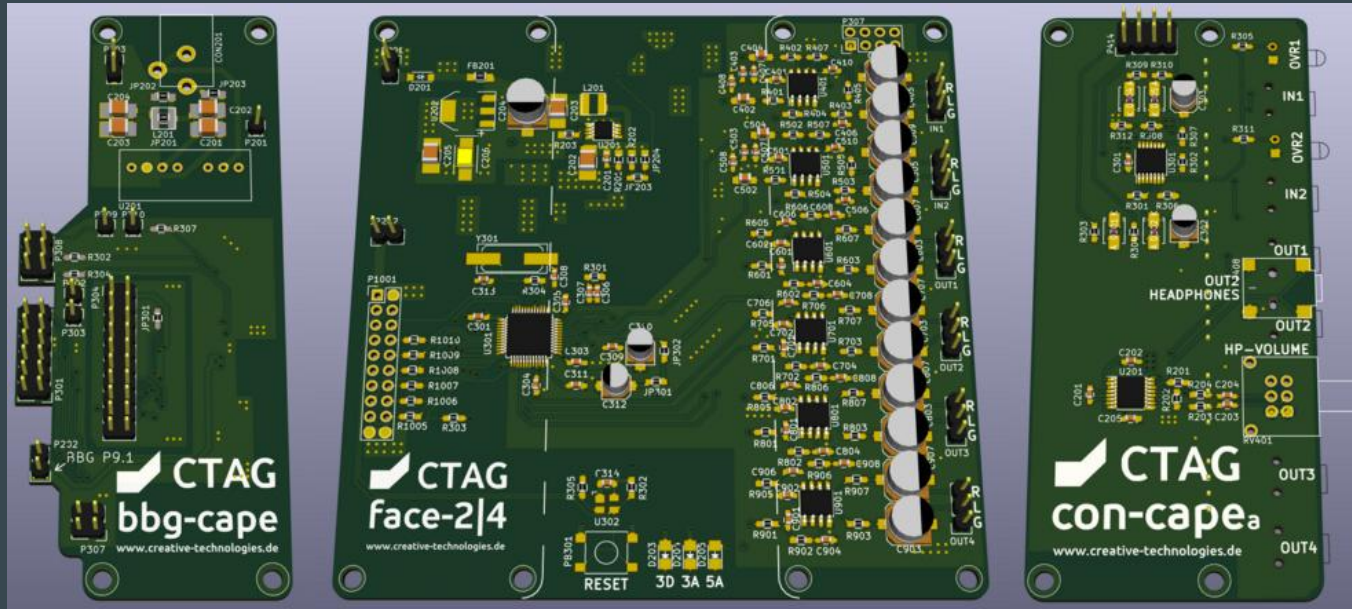
Features of Audio System

- 4 x 24 Bit single-ended inputs
- 8 x 24 Bit single-ended outputs
- 96 kHz sample rate with 8 audio channels
  - 192 kHz with 4 audio channels
- Asynchronous sample rates for playback and capture
- ~ 3,2 ms round-trip-time
- Automatic resampling of 44,1 kHz audio data



Goals of the Audio System

- Open platform for all kinds of audio applications
- High sound quality for music purposes
- Low latency (especially required for music applications)
- Low cost
- Small form factor
- Mobility



Hardware Components

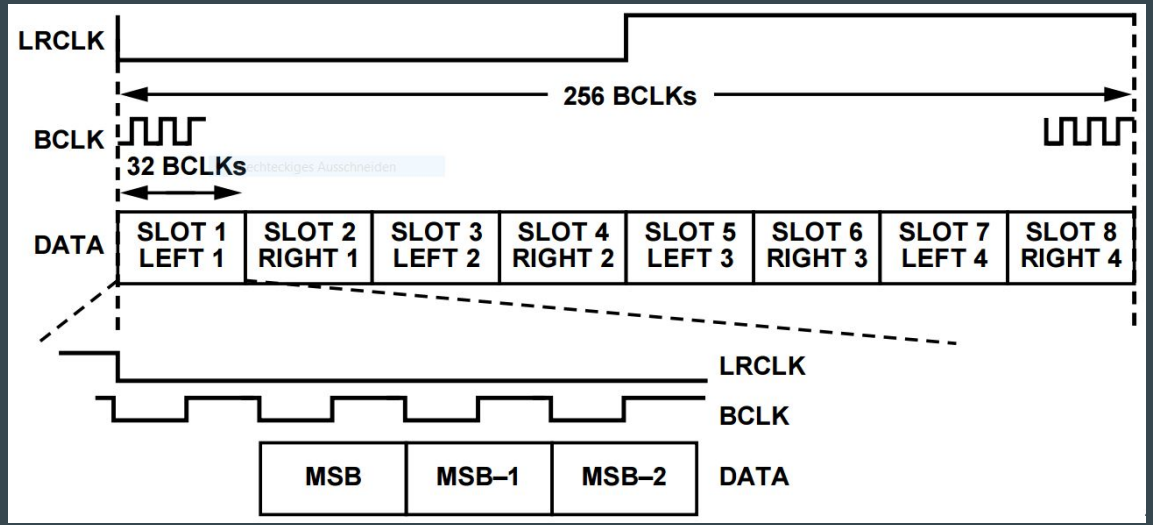
- BeagleBone Green (similar to Raspberry Pi Model B)
  - AM335x 1GHz ARM® Cortex-A8 Processor
  - 512MB DDR3 RAM
  - Multichannel Audio Serial Port (McASP)
  - NEON floating-point

- AD1938 Audio Codec by Analog Devices Inc.
  - 4 differential inputs
  - 8 single-ended outputs
  - up to 192 kHz sample rate
  - asynchronous sample rates for playback and capture

ANALOG DEVICES

Audio Card Driver - Data Transmission

- To transfer more than two audio channels on a single line, time-division multiplexing (TDM) has been used
  - Sender and receiver negotiate on number of audio channels and slot width
  - Positive or negative edge indicates begin of first audio channel



Source: Analog Devices AD1938 Datasheet

Audio Card Driver Architecture

Audio Codec Driver

- Contains only audio codec (AD1938) specific code

Machine Driver

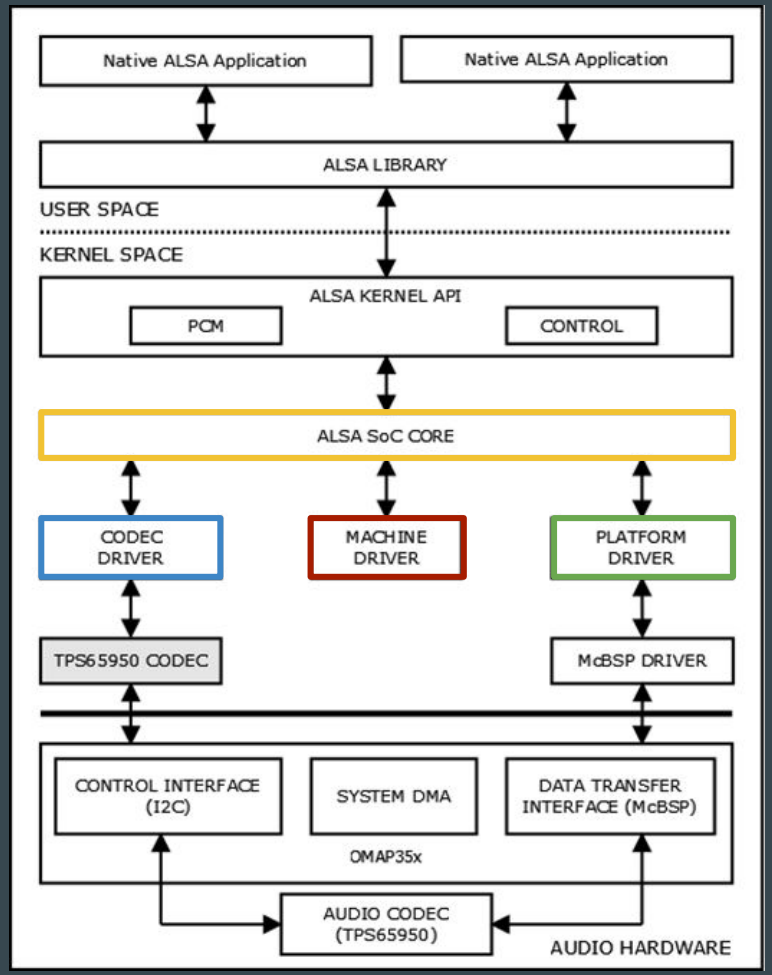
- “Glues” together CPU and codec DAI
- Detailed knowledge of CPU DAI and codec DAI is required
- Detailed knowledge of ASoC API is required

Platform Driver

- Contains only platform (AM335x) specific code

ALSA SoC Core

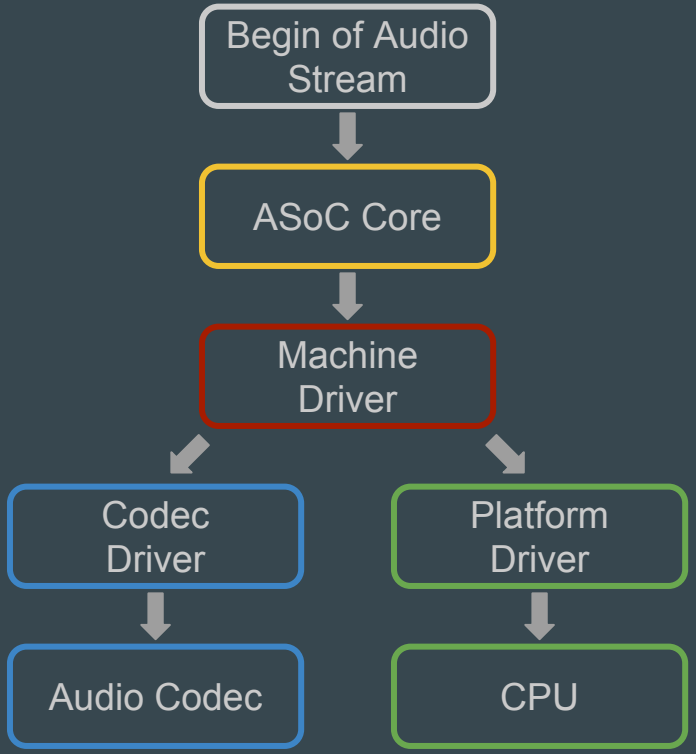
- Every driver component has to be registered in ASoC Core
- ASoC Core is responsible for internal communication

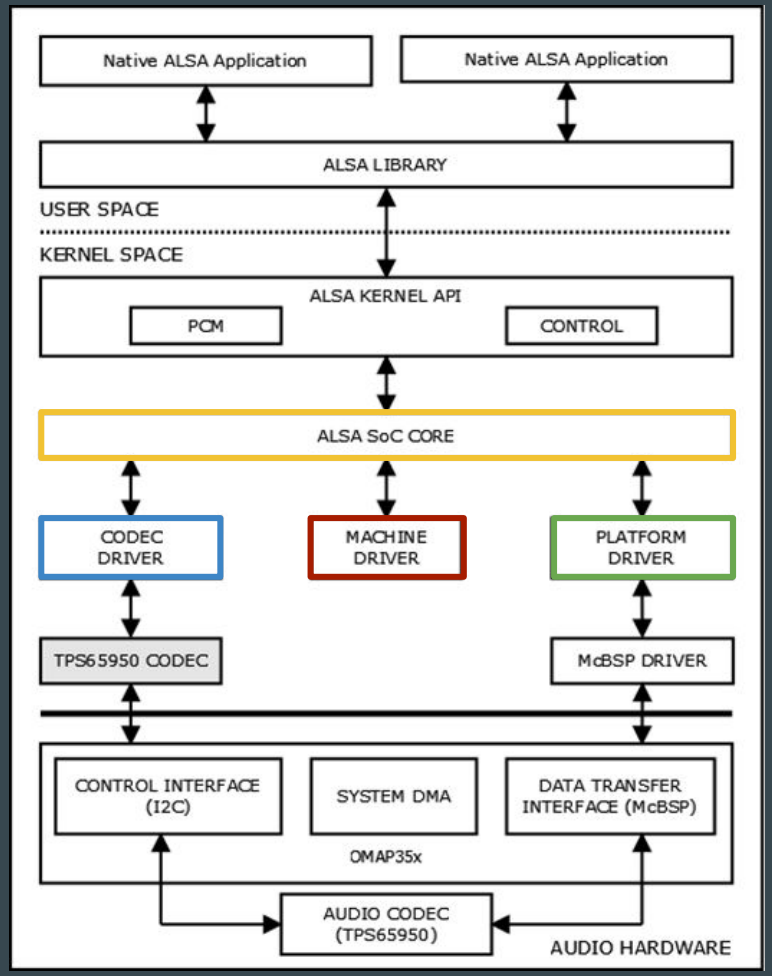


http://processors.wiki.ti.com/images/8/84/DM81xx_Asoc_architecture.png

Audio Card Driver Architecture

Example: Setting up Hardware Parameters





http://processors.wiki.ti.com/images/8/84/DM81xx_Asoc_architecture.png

Audio Card Driver - Device Tree

- A Device Tree describes a hardware platform (or part of it) in a tree data structure
  - Every node of a Device Tree references kernel modules to be automatically loaded in correct order
    - For example SPI node references “ti,omap4-mcspi” and configures clock rate, chip selects, ...
    - After successful configuration SPI node is given to audio card node
  - Based on node properties kernel modules can be dynamically configured without recompilation
    - e.g. TDM slots (audio channels) can be configured in runtime

Audio Card Driver - Device Tree Code Snippet

```
sound {
        compatible = "ctag,face-2-4"; //References Machine Driver
        model = "CTAG face-2-4 8CH";  //Name of audio card
        audio-codec = <&ad193x>;      //References SPI (audio codec) node
        mcasp-controller = <&mcasp0>; //References I2S (TDM) interface node
        audiocard-tdm-slots = <8>;    //Configures number of audio channels
        codec-clock-rate = <24576000>;//Configures master clock of codec
        …
};
```

Audio Card Driver - Parsing Properties of Device Tree

```
static int snd_davinci_audiocard_probe(struct platform_device *pdev) {
    …
    dai->codec_of_node = of_parse_phandle(np, "audio-codec", 0);
    if (!dai->codec_of_node)
        return -EINVAL;

    dai->cpu_of_node = of_parse_phandle(np, "mcasp-controller", 0);
    if (!dai->cpu_of_node)
        return -EINVAL;
    …
};
```

Audio Codec Driver - DAI Definition

```
static struct snd_soc_dai_driver ad193x_dai = {
    .name = "ad193x-hifi",
    .playback = {
        .stream_name = "Playback",
        .channels_min = 2,
        .channels_max = 8,
        .rates = SNDRV_PCM_RATE_48000 | SNDRV_PCM_RATE_96000 |
SNDRV_PCM_RATE_192000,
        .formats = SNDRV_PCM_FMTBIT_S32_LE | SNDRV_PCM_FMTBIT_S16_LE |
            SNDRV_PCM_FMTBIT_S20_3LE | SNDRV_PCM_FMTBIT_S24_LE,
    },
    ...
    .ops = &ad193x_dai_ops,
};
```

Machine Driver - Connection of CPU and Codec DAI

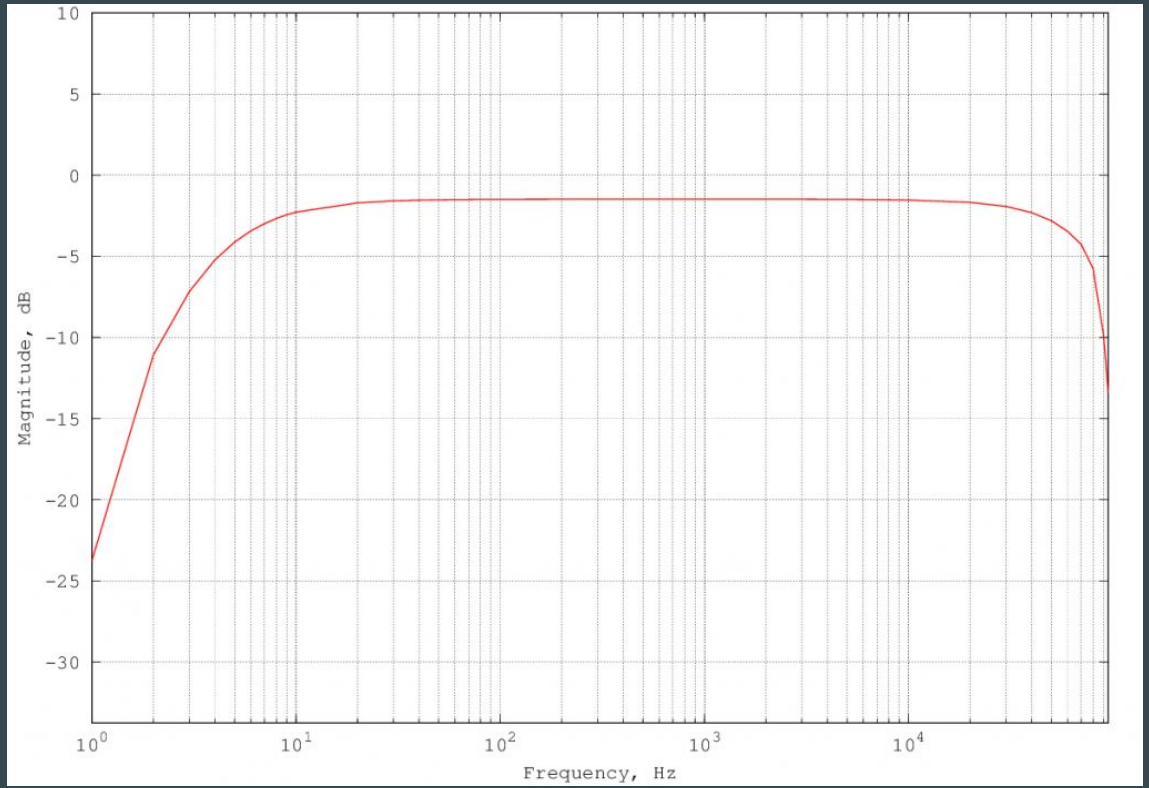
```
static struct snd_soc_dai_link snd_davinci_audiocard_dai = {
    .name = "AudioCard",
    .stream_name = "AudioCard TDM",
    .codec_dai_name = "ad193x-hifi",       //From audio codec driver
    .cpu_dai_name = "davinci-mcasp.0",     //From platform driver
    .dai_fmt = (SND_SOC_DAIFMT_I2S | SND_SOC_DAIFMT_NB_IF |
SND_SOC_DAIFMT_CBM_CFM),
    .ops = &snd_davinci_audiocard_ops,
    .init = snd_davinci_audiocard_init,
};
```

Evaluation - Technical Sound Quality

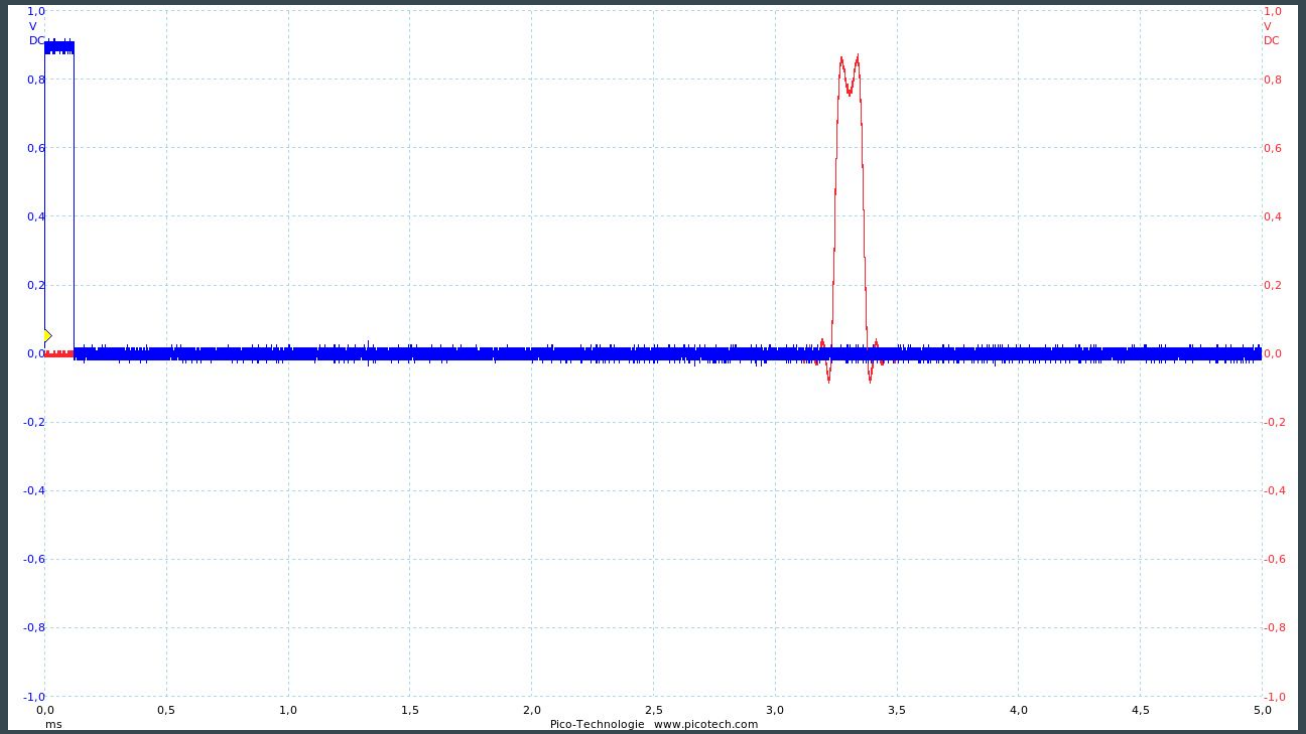
Total Harmonic Distortion (THD)	-89 dB
Total Harmonic Distortion + Noise (THD+N)	-88 dB
Dynamic range (DNR)	110 dB
Crosstalk	98 dB

Note: A script based on GNU Octave has been created for automatic evaluation

Evaluation - Amplitude Spectrum



Evaluation - Latency



~ 3.2 ms Round Trip Time

Further Development within Google Summer of Code

- Two project parts:
  - Porting the CTAG face2|4 Audio Card drivers to BeagleBoard-X15 (AM5728 SoC)
  - Create library to make use of C66x DSPs (libdsp-x15)
- Project was successfully completed in 3 months

GSoC 2016 - Porting Drivers to BB-X15 (AM5728 SoC)

- BeagleBoard-X15 Specification
  - TI AM5728 2×1.5-GHz ARM® Cortex-A15 Processor
  - 2GB DDR3 RAM
  - 2×700-MHz C66 DSPs (ideal for audio applications)
  - 2×ARM Cortex-M4 MCUs
  - 4×32-bit PRUs

  => Much more powerful than BeagleBone Black/Green

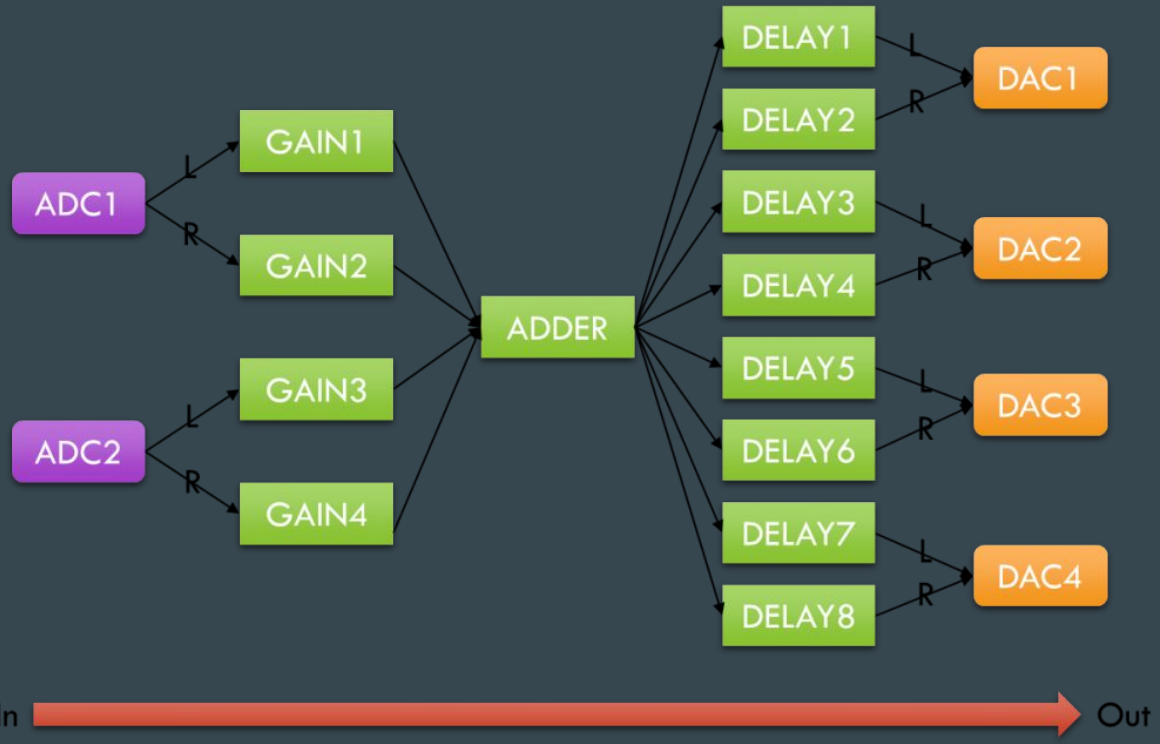
- Project included the following work tasks:
  - Modifications of ASoC Machine Driver (e.g. different configuration of clock signals)
  - Create new Device Tree

GSoC 2016 - DSP Library (libdsp-x15)

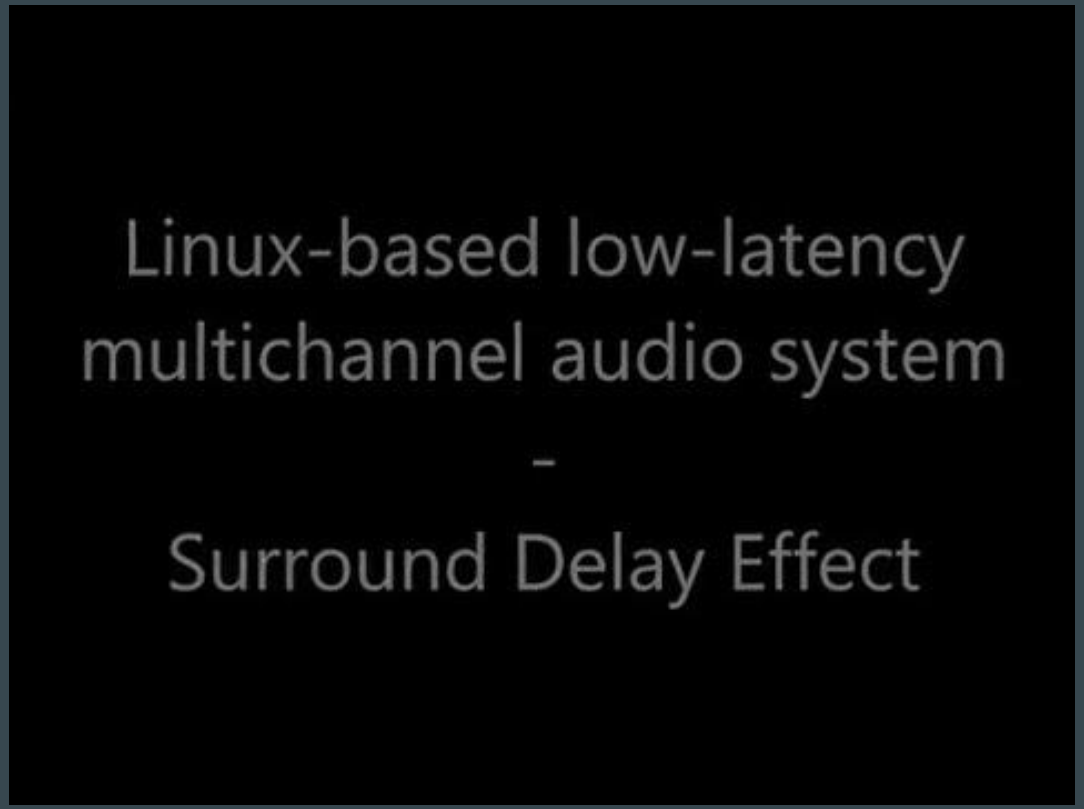
- Offered DSP operations by libdsp-x15
  - Fast Fourier Transformation (FFT)
  - Inverse Fast Fourier Transformation (IFFT)
  - Biquad Filter (often used in audio applications)
- Library uses Remote Processor Framework (remoteproc) of Linux via OpenCL to offload signal operations to C66x DSPs
- Zero-copy buffers are used (shared memory in continuous reserved memory area)
- Concurrent execution of DSP kernels (e.g. FFT and IFFT simultaneously)

Application Demos - Surround Delay Effect

- Based on DSPatch (C++ library)
- All audio inputs are mixed together and routed to 8 delay effect modules with different parameters
- Parameters can be changed in real-time via MIDI

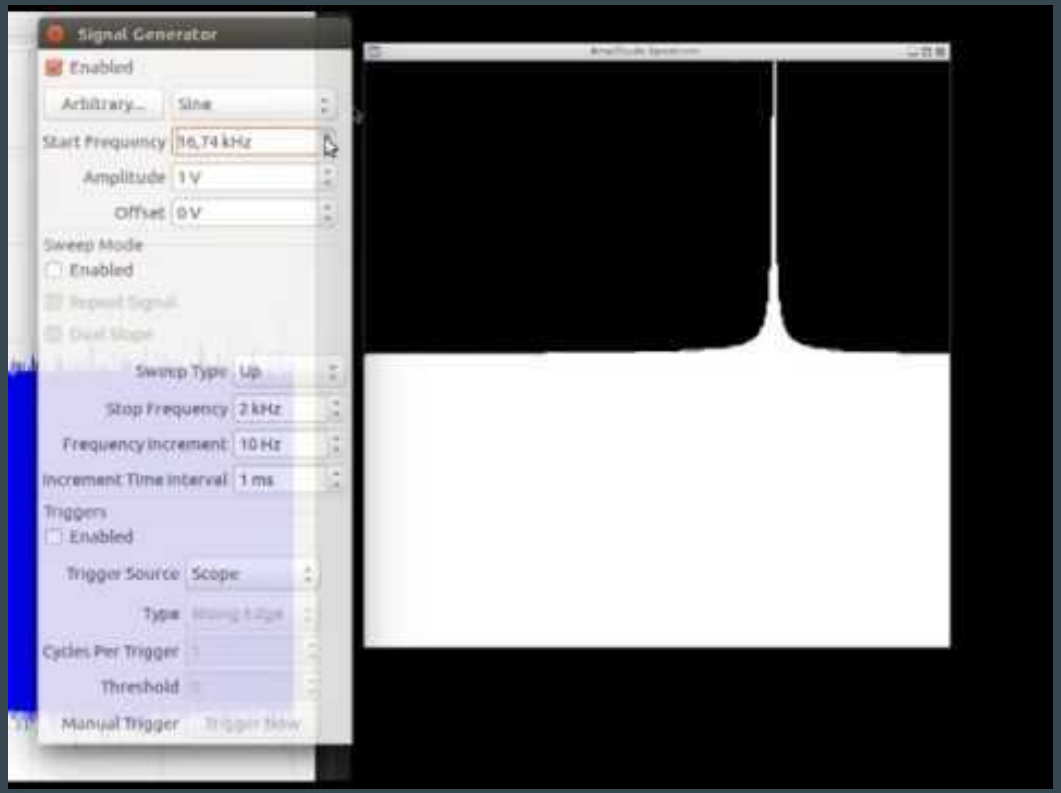


Application Demos - Surround Delay Effect



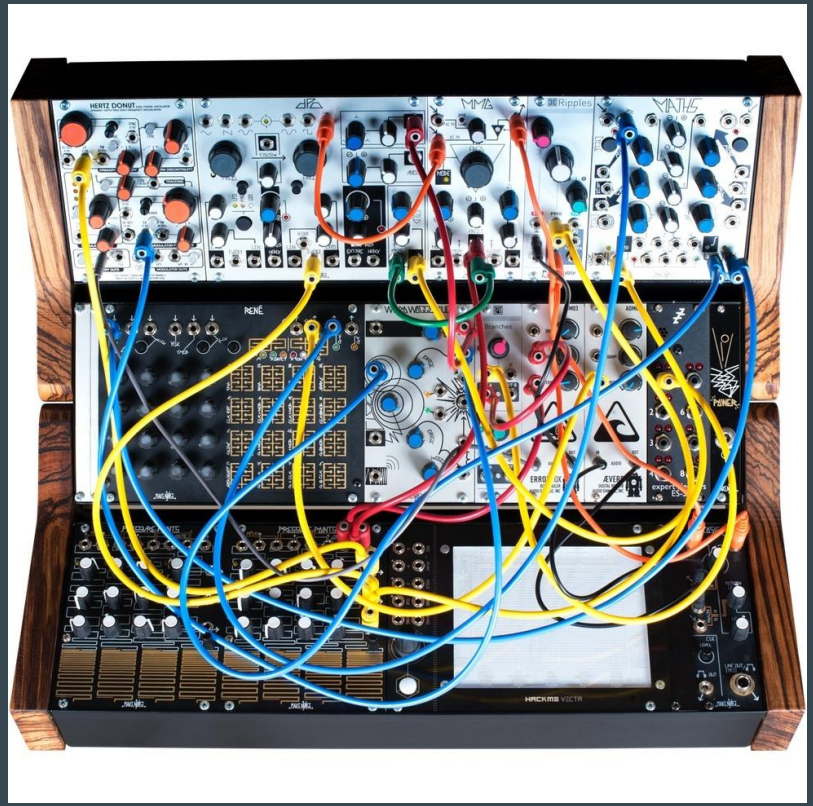
Application Demos - Realtime Spectrum Plot

- Audio signal is generated with function generator and captured in real-time via JACK Audio Connection Kit (JACK)
- Audio signal is transformed to frequency domain via FFT operation
- Amplitude spectrum is plotted via Simple DirectMedia Layer (SDL)



Future Work - Eurorack Synthesizer Modul

- Uses CTAG face2|4 for audio i/o
- Based on BeagleBoard-X15
- Offers touch LCD for GUI and user interaction
- Offers knobs for user interaction



http://rekkerd.org/img/201501/fraptools_uno_03.jpg

Future Work - Fusion of Bela and CTAG face2|4

- Bela platform was developed by Centre for Digital Music of Queen Mary University London
- Bela currently offers
  - Web IDE for graphical flow based programming (like puredata)
  - Ultra-low-latency via PRU
  - 16 sensor inputs for interaction
  - 2 audio outputs
- In the future Bela should offer
  - 8 HQ audio inputs and 16 HQ audio outputs (daisy chain of two AD1938 audio codecs)

=> Redesign of board required

References

- http://creative-technologies.de/linux-based-low-latency-multichannel-audio-system-2/
- https://github.com/ctag-fh-kiel/ctag-face-2-4
- https://github.com/henrix/beagle-linux
- https://github.com/henrix/libdsp-x15/wiki
- http://analog.com/media/en/technical-documentation/data-sheets/AD1938.pdf
- https://github.com/CircuitCo/BeagleBone-Black/blob/master/BBB_SRM.pdf
- http://bela.io/

Thank You