Open Hardware Multichannel Sound Interface for Hearing Aid Research on BeagleBone Black with openMHA: Cape4all

Tobias Herzke[1,4] and **Hendrik Kayser**[1,2,4] and **Christopher Seifert**[3,4] and **Paul Maanen**[1,4] and **Christopher Obbard**[5] and **Guillermo Payá-Vayá**[3,4] and **Holger Blume**[3,4] and **Volker Hohmann**[1,2,4]
[1]HörTech gGmbH, Marie-Curie-Str. 2, D-26129 Oldenburg, Germany
[2]Medical Physics, Carl von Ossietzky Universität Oldenburg, D-26111 Oldenburg, Germany
[3]Institute of Microelectronic Systems, Leibniz Universität, D-30176 Hannover, Germany
[4]Cluster of Excellence "Hearing4all"
[5]64 Studio Ltd, Isle of Wight, UK
info@openmha.org

Abstract

The paper describes a new multichannel sound interface for the BeagleBone Black, *Cape4all*. The sound interface has 6 input channels with optional microphone pre-amplifiers and between 4 and 6 output channels. The multichannel sound extension cape for the BeagleBone Black is designed and produced. An ALSA driver is written for it. It is used with the openMHA hearing aid research software to perform hearing aid signal processing on the BeagleBone Black with a customized Debian distribution tailored to real-time audio signal processing.

Keywords

Hearing aids, audio signal processing, sound hardware

1 Introduction

Hearing aids are the most common form of mitigation for mild and moderate hearing losses. Hearing aids help the wearer to follow conversations and acoustic events in different situations. In the complex acoustic environments that we encounter in our daily life, information about the acoustic scene is inferred at higher stages of the human auditory system and exploited in the brain for, e.g., speech understanding. A hearing loss causes — in addition to reduced sensitivity to soft sounds — a partial loss of this information. Effective signal processing algorithms are required for compensation. For this reason, improving signal processing in hearing aids is an active research topic.

Part of the work in hearing aid research is to develop novel signal processing algorithms that can be used in hearing aids to improve the hearing experience for hard-of-hearing people. Usually, simulations are run and evaluated in terms of objective measures after such an algorithm has been developed mathematically. Results from simulations do not necessarily reflect the benefit of the algorithm a) when integrated in a complete signal processing chain of a hearing aid and b) in a real-world scenario. To assess the usefulness of new hearing aid algorithms for hearing-impaired people, new potential hearing aid signal processing algorithms also have to be tested with hearing impaired test subjects in realistic situations. Running an algorithm under test on an end-user hearing device is practically infeasible as it requires access to a proprietary system of a hearing aid manufacturer, and a large effort for the down-to-hardware implementation is required on such devices. Instead, a software platform can be used to simulate the hearing aid processing chain. The open Master Hearing Aid (openMHA, [HörTech gGmbH and Universität Oldenburg, 2017], [Herzke et al., 2017]) is such a platform. openMHA can be utilized to conduct field tests of hearing aid processing methods running on portable hardware.

The following sections first introduce the software and hardware platforms utilizable to evaluate hearing aid algorithms with hearing-impaired test subjects. We work out the need for a custom multichannel sound interface for a small, portable computer. The subsequent sections report on the hardware design process that resulted in the *Cape4all*[1] BeagleBone sound interface, the sound driver development, and finally the possible usage of the sound interface for hearing aid research.

[1]developed in the cluster of excellence "Hearing4all"

2 Software and Hardware Platform for Hearing Aid Research

HörTech and the University of Oldenburg have developed the openMHA [HörTech gGmbH and Universität Oldenburg, 2017], [Herzke et al., 2017] software platform for the development and evaluation of hearing aid algorithms, where individual hearing aid algorithms can be implemented as plugins and loaded at run-time. The platform provides a set of standard algorithms to form a complete hearing aid. It can process audio signal in real-time with a low delay (<10 ms) between sound input and sound output. (The actual delay depends on the sound hardware used for input and output, configuration options like sampling rate and audio buffer size, and also on delay introduced by some signal processing algorithms.)

In its current version 4.5.5, the openMHA software platform can execute on computers with Linux and Mac OS operating system, e.g., in a laboratory environment. Toolboxes for generating virtual sound environments in a laboratory exist (e.g. TASCAR [Grimm et al., 2015]) but the sound environment in a lab — and even more the subject behavior in a lab environment — will always differ from real environments encountered by hearing aid users in real life. To test real-life situations, we have to go outside and into real situations with hearing-impaired users wearing a mobile computer that executes the openMHA and provides the first chance to test new algorithms in real-world situations. In the past, we have used laptops for this purpose but with the advent of small, ARM-based single board computers like the Raspberry Pi, BeagleBone, and several others these become an option for executing openMHA that imposes less weight to carry around for the test subjects. The processing power of these devices is significantly lower than that of PCs and laptops, which will always limit the extent and setup of algorithms that can be executed on such a mobile platform (compared to a PC).

openMHA is meant as a common platform to be used by different hearing aid research labs to combine their work. By providing a solid base platform, we want to encourage researchers to implement and publish their algorithms as openMHA plugins so that work can be shared and results can be reproduced by independent labs.

For this purpose, openMHA includes a toolbox library that already contains functions and classes useful to more than one algorithm to speed up implementation of new algorithms. As a key to usability of the software in different usage scenarios openMHA also includes several manuals for different entry levels ranging from plugin developments over application engineering based on available plugins and functionality to the application of the software in the context of audiological research and hearing aid fitting controlled through a graphical user interface (GUI). Step-by-step tutorials on the implementation of openMHA plugins as well as examples of configurations are provided to enable an autonomous familiarization for new users.

Some hearing aid algorithms — such as directional microphones — need to process the sound from more than one microphone per ear which is why a multichannel sound card is generally needed to capture the sound from all hearing aid microphones. Professional sound cards can be used for this purpose in stationary laboratory setups. Bus-powered USB sound cards can be used with laptops in mobile evaluation setups, but the choice of bus-powered interfaces with more than 2 input channels is limited. We have observed that the total delay between input and output sounds that can be achieved with USB sound cards is always larger than what can be achieved with similar sound cards with PCI or Expresscard interface. This difference in delay is in the order of 2 ms, which will already affect some hearing aid algorithms. We have also observed that the delay may vary from one start of the sound card to the next with USB sound cards, in the range of 1 ms, which is detrimental to some processing algorithms such as acoustic feedback reduction. (Feedback reduction algorithms are an essential part of a hearing aid processing chain and need the system to be as invariant as possible to work effectively.) The Inter-IC Sound (IIS or I^2S) bus — transporting sound data from the SoC[2] to the audio codecs with the AD/DA converters (and back) — is accessible on expansion headers on many of the single-board ARM computers, making it possible to create custom sound interface hardware.

Third parties already provide multichannel sound interfaces for popular boards like the BeagleBone Black and the Raspberry Pi. Of these two devices, the BeagleBone Black has the advantage of hardware support for multichannel

[2]Abbreviation for System on a Chip, the combination of a microprocessor and several peripherals (e.g. graphics unit, sound interface) on a single chip.

audio input/output. See Section 3.1 for details.

One multichannel sound interface option for the BeagleBone Black is the *BELA* cape [Moro et al., 2016]. It provides stereo in/out and additional 8 analogue data acquisition channels. These additional 8 analogue data acquisition channels can also be used to capture audio but do not provide anti-aliasing filters, and achievable sampling rates depend on the number of channels in simultaneous use. The *BELA* cape makes use of real-time hardware present on the BeagleBone Black. Audio processing algorithms can be compiled to execute on this real-time hardware, process the input channel data, and produce output channel data. Existing Linux audio processing applications using ALSA[3] or JACK[4][Davis, 2003] and common features of the operating system cannot execute on this real-time hardware.

Another multichannel audio interface developed for BeagleBone platforms is the *CTAG face 2|4* [Langer and Manzke, 2015], [Langer, 2015]. Its hardware design is available open-source from GitHub and drivers have been included in official BeagleBoard SD card images. Providing capabilities for multichannel signal processing this device is in principle suitable for hearing aid processing on the BeagleBone Black. A drawback that remains here is the necessity to add external power supply for the microphones connected to the device.

The *Octo Audio Injector* sound card `http://www.audioinjector.net/rpi-octo-hat` offers 6 input channels and 8 output channels for the Raspberry Pi. Although the Raspberry Pi offers no hardware support for more than two sound channels, this sound card manages to offer enough input channels to connect 2 hearing aids with 3 microphones each. A disadvantage of this sound card for hearing aid research is that additional external microphone preamplifiers are needed to raise the microphone signals to line level, which adds to the hardware that test subjects would have to carry around. An example setup for teaching hearing aid signal processing [Schädler, 2017], [Schädler et al., 2018] uses the stereo version of this sound card together with external microphone preamplifiers.

[3]Acronym for Advanced Linux Sound Architecture, name for a system of Linux kernel sound card drivers and user space API to exchange sound data with these drivers.

[4] Self-referencing acronym for JACK Audio Connection Kit, a user-space server application and library to connect inputs and outputs of audio applications and sound cards.

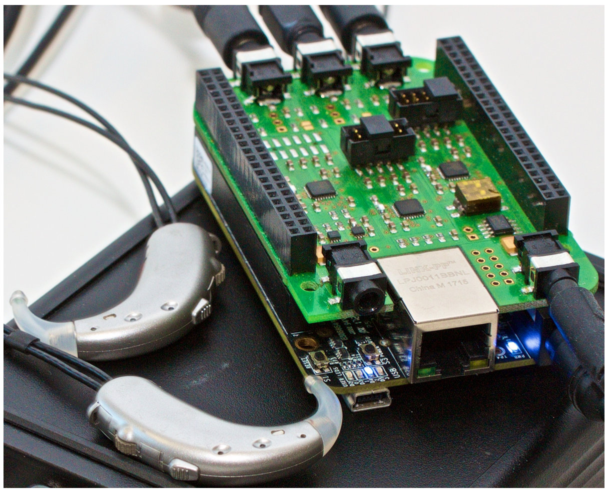
Figure 1: *Cape4all* with two hearing aids (each containing three microphones) connected.

3 Development of the Cape4all Multichannel Sound Interface for Hearing Aid Research

We have a need for a compact multichannel sound interface for a single-board ARM computer with integrated microphone pre-amplifiers for hearing aid research. Since such a multichannel sound interface was not available, we decided to develop such a sound interface ourselves.

3.1 Choice of ARM Board Basis for a Multichannel Sound Card

In the ongoing developments of the Cluster of Excellence "Hearing4all"[5] several audio interfaces were developed proving the inter IC sound (IIS or I^{2}S) in combination with the Analog Devices ADAU1761 [Analog Devices Inc., 2009] stereo audio codec useful [Seifert et al., 2015]. To gain multichannel capabilities, a time division multiplex (TDM) scheme specified for I^{2}S is used. The chosen ADAU1761 codecs support a TDM output scheme. To allow the usage in combination with an ARM-based platform and therefore with openMHA, the BeagleBone Black with native I^{2}S TDM support by the integrated McASP[6] interfaces was chosen.

3.2 Hardware Design

The *Cape4all* hardware was designed by the Leibniz University Hannover based on [Seifert

[5]`http://hearing4all.eu/`

[6]Abbreviation for Multichannel Audio Serial Port.

et al., 2015].

In addittion to the I^2S TDM output capabilities the Analog Devices ADAU1761 audio codecs have integrated microphone amplifiers. Up to 3 microphones for each ear on a bilateral fitting are assumed in the context of hearing device development. Therefore, 3 stereo audio codecs are integrated on the *Cape4all* PCB[7] allowing up to 6 input and output channels simultaneously. Due to the TDM scheme, only five signal connections are required to transport and synchronize all 3 codecs with 6 input and output channels and the McASP interface of the BeagleBone Black.

The board provides standard stereo jacks for connecting off-the-shelf sound hardware as well as pin headers for custom designs. 3 stereo jacks are mounted on the board for the 6 input channels, and 2 additional stereo jacks for the first 4 output channels. The remaining output channels are only accessible through the pin headers. An on-board voltage regulator provides microphone bias voltage which can be switched on and off as needed and routed to different connectors. The bias voltage can be altered by exchanging on-board resistors. For more details, see the reference manual provided with the hardware design files and the driver as download from `https://github.com/HoerTech-gGmbH/Cape4all`. Figure 1 shows the hardware in use.

3.3 Hardware Tests and Design Revisions

In the testing process of previously built audio interface boards using the ADAU1761 stereo audio codecs, it was revealed that the internal components of the codecs create bus collision. The I^2S TDM bus digital output pins of the codecs do not provide high-resistance state, driving the signal high or low preventing another codec to put data on the same signal. The documentation of the codecs did not give any details helping to avoid the bus collision. In order to avoid this, an OR-gate was added to the board design to merge the signals of the codecs to one signal. This solves the problem on voltage level but does not prevent timing collision due to wrong configuration of the codec outputs. The correct codec configuration is ensured by the ALSA driver (see Section 4). In normal TDM configuration, filling 6 of the available 8 timeslots, all 3 audio codecs are working correctly. For further details on I^2S TDM signaling see [Seifert et al., 2013].

[7]Abbreviation for Printed Circuit Board.

3.4 Release as Open Hardware

The hardware design files are released under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License on GitHub `https://github.com/HoerTech-gGmbH/Cape4all`.

4 Driver development

The ALSA sound driver for the *Cape4all* sound interface was developed by 64 Studio.

As the Linux kernel already has support for both the McASP Audio Serial Port [Pandey et al., 2009] used on the BeagleBone Black and the ADAU1761 codec [Clausen, 2014] used on the *Cape4all*, the development by 64 Studio was to create a glue-driver explaining to the SoC the order the codecs are arranged on the *Cape4all*. The driver registers the cape as effectively one PCM device with three mixer sub-devices (corresponding to the three physical ADAU1761 codecs), each with their own set of controls in the ALSA mixer. Also, the driver sets up the codec's clock-path, TDM slots and various other default settings.

As the driver exposes the *Cape4all* as a regular ALSA device with three mixer sub-devices, each with their own ALSA controls, application software may communicate with these devices without any modifications.

4.1 Limitations

The McASP used on the BeagleBone Black is clocked from a 24.576 MHz crystal. This limits the available sample rates to be a whole divisor of this clock, for instance 24 kHz or 48 kHz is acceptable but 22.05 kHz or 44.1 kHz is not.

The ADAU1761 codecs do not directly support sharing 6 channels between 3 separate codecs on a TDM bus. As a workaround, the TDM mode for transferring 8 channels is used, where 2 channels contain no data. A consequence is that the sound card appears to have 8 channels in ALSA but only the first 6 channels, corresponding to the physical channels, should be used.

4.2 Release

The driver code is released as open source software under the GNU General Public License, Version 2 or later, in the same git repository as the hardware design files on GitHub, `https://github.com/HoerTech-gGmbH/Cape4all`.

5 Usage

As Linux distributions created by SoC development board manufacturers are typically not being suited to audio signal processing and contain a lot of applications that are not useful in this context, a custom Debian distribution has been prepared by 64 Studio. E.g. the JACK Audio Server contained in this custom distribution was built without DBUS support to allow the system to run without a GUI and the final Debian system was tweaked by 64 Studio for basic real-time performance. An image file containing this distribution is available for download together with the hardware design. It contains just the software needed to run openMHA, has device-tree and custom Kernel built-in as well as custom tweaks for increased real-time audio performance.

These steps are needed to prepare a BeagleBone Black for multichannel signal processing with openMHA and *Cape4all*:

- Download and copy image to SD-card
- Download and compile openMHA on the system
- Set up system for higher audio performance according to manual provided
- Start JACK Audio Server with settings according to the openMHA configuration to be run
- Read example configuration provided with openMHA and start processing

The openMHA processes can be accessed at runtime through a TCP/IP connection. This connection can be used to read out and change parameters of the running system. By this means it is possible to run a GUI on a laptop or tablet computer that can be used to control the processing parameters remotely. For details, refer to the openMHA application manual.

6 Conclusions

Cape4all is a working, multichannel sound interface for the BeagleBone Black with integrated microphone pre-amplifiers which makes it suitable for hearing aid research, where pre-amplifiers are essential and where a small form factor matters.

A working ALSA driver has been developed that takes care of the proper initialization of the codecs and the multichannel capabilities of the BeagleBone Black and then drives the multichannel sound exchange between user space applications and the codecs on the sound interface.

Both, the hardware design files and the driver, have been published with open licenses on GitHub, `https://github.com/HoerTech-gGmbH/Cape4all`.

In its current state, the *Cape4all* can be run together with a JACK Audio Server on a BeagleBone Black reliably with a 4 ms buffer (128 samples per channel) at a 32 kHz sampling rate. This is the state directly after driver development before any optimization towards shorter audio buffers has been performed. This current state is an important step towards our goal of a mobile hearing aid algorithm evaluation setup, but it needs to be improved to achieve the target overall audio delay below 10 ms between input and output sounds, considering that some of the algorithms will add a small algorithmic delay. Therefore, we are going to further optimize the driver in collaboration with 64 Studio after the initial release to enable smaller audio buffer sizes.

7 Acknowledgements

This work was supported by the German Research Foundation (DFG) Cluster of Excellence EXC 1077/1 "Hearing4all".

Research reported in this publication was supported by the National Institute On Deafness And Other Communication Disorders of the National Institutes of Health under Award Numbers R01DC015429. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

Analog Devices Inc. 2009. ADAU1761 – SigmaDSP stereo, low power, 96 khz, 24-bit audio codec with integrated PLL. `http://www.analog.com/static/imported-files/data_sheets/ADAU1761.pdf`.

Lars-Peter Clausen. 2014. `https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/sound/soc/codecs/adau17x1.c`.

Paul Davis. 2003. Jack audio connection kit. http://jackaudio.org/.

Giso Grimm, Joanna Luberadzka, Tobias Herzke, and Volker Hohmann. 2015. Toolbox for acoustic scene creation and rendering (TASCAR) – render methods and research applications. In *Proceedings of the Linux Audio Conference*, pages 1–7, Mainz. Johannes Gutenberg-Universität.

Tobias Herzke, Hendrik Kayser, Frasher Loshaj, Giso Grimm, and Volker Hohmann. 2017. Open signal processing software platform for hearing aid research (openMHA). In *Proceedings of the Linux Audio Conference*, pages 35–42, Saint-Étienne. Université Jean Monnet.

HörTech gGmbH and Universität Oldenburg. 2017. openMHA web site on GitHub. `http://www.openmha.org/`.

Henrik Langer and Robert Manzke. 2015. Linux-based low-latency multichannel audio system (CTAG face2—4). `http://www.creative-technologies.de/linux-based-low-latency-multichannel-audio-system-2/`.

Henrik Langer. 2015. Linuxbasiertes Mehrkanal-Audiosystem mit niedriger Latenz.

Giulio Moro, Astrid Bin, Robert H Jack, Christian Heinrichs, Andrew P McPherson, et al. 2016. Making high-performance embedded instruments with bela and pure data.

Nirmal Pandey, Suresh Rajashekara, and Steve Chen. 2009. `https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/sound/soc/davinci/davinci-mcasp.c`.

Marc René Schädler, Hendrik Kayser, and Tobias Herzke. 2018. Pi hearing aid. *The MagPi (Raspberry Pi Magazine)*, 67:34–35.

Marc René Schädler. 2017. openMHA on Raspberry Pi. `https://github.com/m-r-s/hearingaid-prototype`.

Christopher Seifert, Guillermo Payá-Vayá, and Holger Blume. 2013. A multi-channel audio extension board for binaural hearing aid systems. In *Proceedings of ICT. OPEN. Conference ICT. OPEN*, pages 33–37.

Christopher Seifert, Guillermo Payá-Vayá, Holger Blume, Tobias Herzke, and Volker Hohmann. 2015. A mobile SoC-based platform for evaluating hearing aid algorithms and architectures. In *Consumer Electronics-Berlin (ICCE-Berlin), 2015 IEEE 5th International Conference on*, pages 93–97. IEEE.