Software Architecture for a Multiple AVB Listener and Talker Scenario

Christoph Kuhr and **Alexander Carôt**
Department of Computer Sciences and Languages, Anhalt University of Applied Sciences
Lohmannstr. 23, 06366 Köthen,
Germany,
{christoph.kuhr, alexander.carot}@hs-anhalt.de

Abstract

This paper presents a design approach for an AVB network segment deploying two different types of AVB server for multiple parallel streams. The first type is an UDP proxy server and the second server type is a digital signal processing server. The Linux real time operating system configurations are discussed, as well as the software architecture itself and the integration of the Jack audio server. A proper operation of the JACK server, alongside two JACK clients, in this multiprocessing environment could be shown, although a persisting buffer leak prevents significant jitter and latency measurements. A coarse assessment shows however, that the operations are within reasonable bounds.

Keywords

AVB, JACK, signal processing, public internet, multimedia streaming

1 Introduction

1.1 Soundjack and fast-music

Soundjack [1] is a realtime communication software that establishes up to five peer to peer connections. This software was designed from a musical point of view and first published in in 2009 [2]. Playing live music via the public internet is very sensitive to latencies. Thus, the main goal of this application is the minimization of latencies and jitter. The goal of the research project fast-music, in cooperation with the two companies GENUIN [3] and Symonics [4], is the development of a rehearsal environment for conducted orchestras via the public internet. 60 musicians and one conductor shall play together live. Further field of research is the transmission of low delay live video streams and motion capturing of the conductor.

1.2 Concept for a Realtime Processing Cloud

A specialized and scalable server infrastructure is required to provide the realtime streaming requirements of this research project. The service time property of an Ethernet frame arriving on a serial network interface at the wide area network (WAN) side of this server cloud, is of paramount importance for the software design. During the service time of a single UDP stream datagram, no concurrent stream datagrams can be received. Thus, the latencies of all streams arriving on such an interface are accumulated.

In addition to connecting the 60 streams to each other, the Soundjack cloud provides digital signal processing algorithms for audio and video streams. Digital signal processing is computationally expensive and may cause unwanted latencies. Thus, a GPU based signal processing in realtime will be investigated in this research project as well. A basic and scalable concept to address these two requirements is shown in fig. 1.

Audio Video Bridging / Time-Sensitive Networking (AVB / TSN) enables computer networks to handle audio and video streams in realtime. AVB is a set of IEEE 802.1 industry standards, operating on layer 2 of the OSI model [5].

- IEEE 802.1AS [6]
  Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks
- IEEE 802.1Qat [7]
  Virtual Bridged Local Area Networks - Amendment 14: Stream Reservation Protocol (SRP)
- IEEE 802.1Qav [8]
  Virtual Bridged Local Area Networks - Amendment 12: Forwarding and Queueing Enhancements for Time-Sensitive Streams

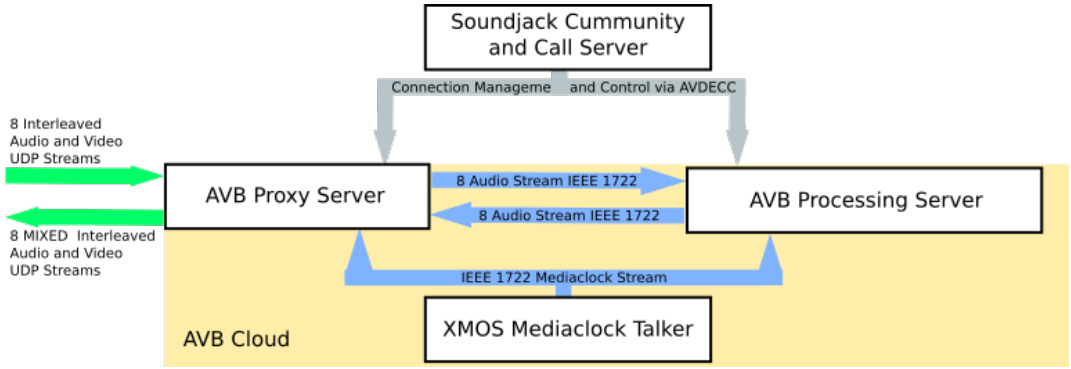

Figure 1: Soundjack Realtime Processing Cloud Concept

- IEEE 1722 [9]
  IEEE Standard for Layer 2 Transport Protocol for Time-Sensitive Applications in Bridged Local Area Networks
- IEEE 1722.1 [10]
  IEEE Standard for Layer 2 Transport Protocol for Time-Sensitive Applications in Bridged Local Area Networks

AVB extends a generic Ethernet computer network by the means of synchronization, resource reservation and bandwidth shaping. This way lower latencies and jitter, the avoidance of packet bursts and bandwitdh shortage are addressed.

AVB networks require special hardware for timestamping Ethernet frames with seperate bandwidth shaped transmission queues for AVB traffic. The Intel corporation provides the I2XX Series of NICs with the open source Open-AVB [11] driver.

Two server types are required for the Soundjack cloud, an AVB proxy server and an AVB processing server. Both server types are connected to the same AVB network segment. Each server is also connected to a non-AVB network segment, together with a Soundjack session server, which acts as an IEEE 1722.1 AVDECC controller endpoint. IEEE 1722.1 AVDECC traffic is not necessarily time-sensitive, thus a non-AVB network segment is used for command and control purposes. The Soundjack session server also provides the online services to the Soundjack client software and handles the connection management of public internet streams, establishes peer to peer and client-server connections.

All AVB servers are registered for a mediaclock stream, which is supplied by an XMOS/Atterotech development board [12]. The mediaclock stream maintains a constant mediaclock to synchronize the packet transmission times of the AVB servers. Without such a synchronization, each server would depend on the precise clock of an audio interface hardware, the CPU clock indicates too much jitter, which in turn would also require a central synchronization mechanism to provide a fully mediaclock-synchronized network segment.

2 Software Requirements for a Multiple AVB Listener and Talker

The AVB server software requires a proper configuration of the operating system and the AVB hardware support, to use the timestamping, bandwidth reservation and shaping. A multiprocessing design, as shown in fig. 2, takes care of all aspects required for multiple independent AVB talkers and listeners.

2.1 Operating System

The ability of Linux to communicate with raw sockets [13, p. 655] and also to be patched to operate in realtime mode, makes it the operating system of our choice. We decided to use the Linux Mint distribution release 18 Sarah, which is based on Ubuntu/Debian. Linux Mint 18 uses the Systemd init process, which makes it easier to dynamically handle OS services.

AVB requires three background services. A gPTP daemon, a MAAP daemon and a MRP daemon. Each requires super user permissions for raw socket communication.

In addition to the background services, a one-time-task to unload the generic Intel e1000/IGB kernel modul and replace it with the Open-AVB AVB IGB kernel module is required. The AVB talkers running on the system need the hardware transmit queues of the Intel I210 Ethernet NIC to be redirected to the bandwidth shaper transmit queues, so that the I210 NIC might use the FQTSS mechanism for enqueueing AVTP

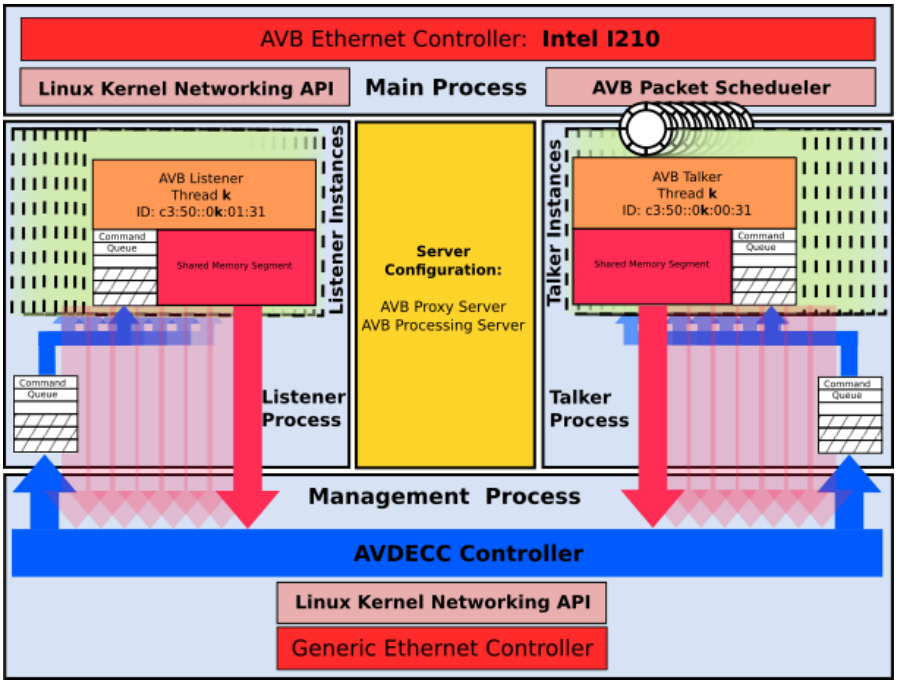

Figure 2: General AVB Server Architecture (MRP, Talker and Listener Processes)

packets from the DMA memory.

The Open-AVB project provides Shell scripts to setup those services.

The Linux kernel may be patched, configured and compiled for realtime operation [14]. A Linux realtime kernel with either the SCHED_FIFO or the SCHED_RR scheduling enabled, handles CPU tasks based on their priorities. A task requesting the CPU, that is scheduled by either scheduler, has a latency solely depending on tasks with a higher or equal priority. Examples for tasks that can still delay the execution of high priority task are DMA bus mastering, ACPI power management, CPU frequency scaling and hyperthreading techniques [15]. These interfering tasks have to be taken into account and carefully tuned, when configuring a realtime Linux system such as:

- Using POSIX realtime mutexes instead of spinlocks.
- Interrupt handlers are moved to the userspace process.
- Avoid priority inversion by priority inheritance.

Another scheduler was introduced in the Linux kernel version 3.14 [16], SCHED_DEADLINE. SCHED_DEADLINE is based on the earliest deadline first (EDF) scheduling enhanced by the constant bitrate server algorithm (CBS). The EDF scheduling with CBS was specifically developed for multimedia applications [17] [18]. This scheduler does not rely solely on the priority, but assigns an absolute deadline and a budget to each task. At each CPU cycle a specific budget is available to the scheduler. If a task is out of budget it is preempted to ensure the execution of another task. With EDF scheduling however, it is important to avoid deadlocks that result from CPU over-utilization. The kernel has an inbuilt mechanism to minimize the risk, by disabling CPU affinity for EDF-scheduled tasks.

The kernel we use in this project is mainstream release 4.8.6 with the realtime patch 4.8.6-rt5, which takes care of the above mentioned realtime mutexes, userspace interrupt handlers and priority inheritance. Besides patching the kernel for realtime operation, several optimization steps are performed. Kernel modules for unnesscary hardware support, e.g. most network interface drivers and peripheral device drivers were removed. Furthermore, the kernel module for NVidia's proprietary graphic adpater and CUDA driver was patched to be used with a realtime kernel.

The optimization of the OS mainly concerns AVB. Since the AVB implementation requires the Direct Memory Access (DMA) [19, p. 412] memory for operation, it is required to use the Memory Management Unit (MMU) in soft mode in `/etc/default/grub`, so that direct

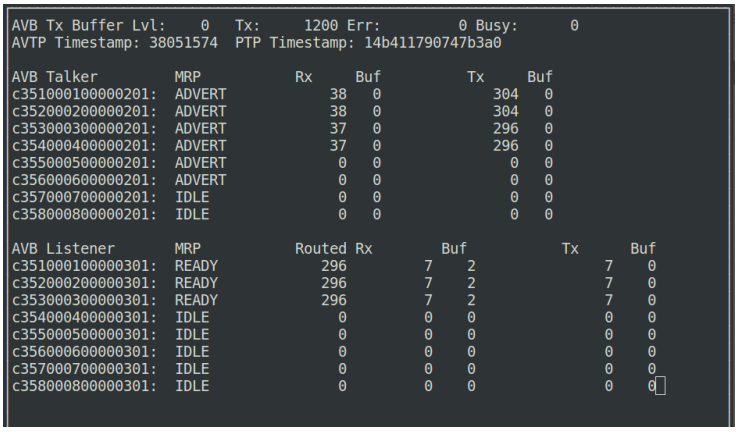

Figure 3: NCurses Shell User Interface

hardware addresses are used instead of a virtual address space, when necessary. The parameter `iommu=soft` prevents the usage of the IOMMU when communicating with the IGB DMA memory, but allows the Focusrite Solo to use it.

```
GRUB_CMDLINE_LINUX_DEFAULT="text iommu=soft"
```

It is also necessary to take care of the priorities for the interrupts, because it has a major influence on the task scheduling. The most important interrupt is the one of the NIC providing the mediaclock stream followed by the interrupts for the USB audio interface device. Linux provides the `/etc/default/rtirq` script to enforce those priorities, which are defined by the order of the RTIRQ_NAME_LIST attributes.

```
RTIRQ_NAME_LIST="enp4s0 enp4s0-TxRx-0
enp4s0-TxRx-1 enp4s0-TxRx-2 enp4s0-TxRx-3
snd usb snd_usb_audio enp2s0 i8042"
```

Further optimizations, e.g. the deactivation of the swappiness or configuring limits in a range a user might operate in, aim to increase the realtime responsiveness of the operating system as a whole [20]. Finally, the system memory is unlocked and realtime priority is assigned to the user-space application.

2.2 Software Architecture

The AVB server software is running five processes in parallel, to distribute processing time more evenly over the available CPU cores. The parent process forks four children and operates afterwards as mediaclock receiver and AVTP packet scheduler. The first child process is the management process and runs the AVDECC controller instance. All AVDECC operations are sent via command queues to a talker or listener instance, except the creation of talkers and listeners themselves. The creation of talkers and listeners is not covered by the AVDECC standard, thus a vendor specific command and response [10, p. 151] has been implemented. Status variables and stream states are written to and accessed by a POSIX shared memory segment. The management process also provides a terminal user interface, as shown in fig. 3, to monitor counters and states of the AVB streams in realtime. Talker and listener endpoint thread instances are created by the talker and the listener processes, respectively. The fifth process is the MRP process, handling all of the stream reservations of the endpoint instances. Figure 2 shows the general AVB server software architecture.

A talker instance reads audio and video data from some process and puts the data into the payload of an AVTP packet, which is then pushed to its circular buffer. The circular buffer is subsequently and continuously read by the AVTP packet scheduler. If the server is configured as AVB proxy server, it receives UDP streams from the assigned Soundjack client and thus provides audio and video data. If the server is configured as AVB processing server, audio and video data is provided by a realtime signal processing application.

A listener instance receives an AVTP stream from the Linux kernel network API. AVTP packets are filtered based on the destination MAC address and the ether type field with a Berkley Packet Filter (BPF) [21] [13, p. 705] mask. AVTP packets that match the filter expression are pushed to the circular buffer of the respective listener instance.

There are some aspects the software needs to take care of to make use of the realtime kernel. First of all, the memory required for dy-

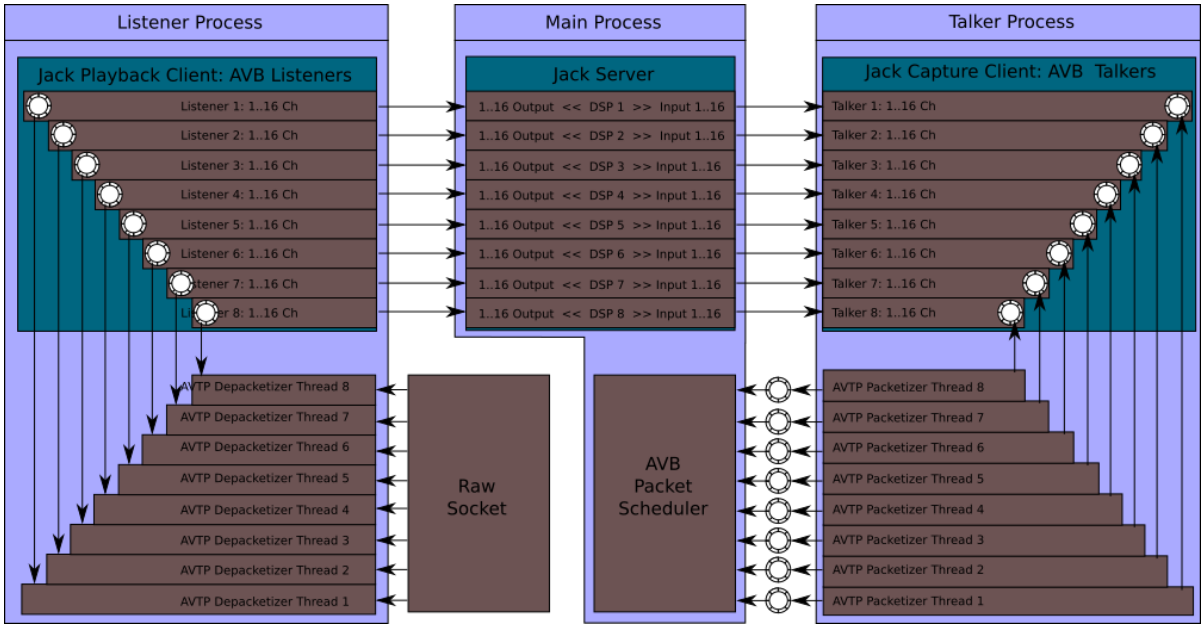

Figure 4: JACK Server and Clients

namic allocations at runtime has to be locked at application start. Otherwise, memory allocations would always be freed after their use and the application eventually crashes, due to memory page faults [22]. Secondly, locks have to be used to prevent the preemption of the task in time critical segments of code. The software also needs to be assigned a proper task priority, so that its scheduling takes place within the required deadlines.

2.3 Server Configurations

In case of the AVB proxy server configuration, the AVTP stream is converted to an UDP stream that is returning to a Soundjack client. In the case of the AVB processing server configuration, the audio and video data is provided to the realtime signal processing application.

2.3.1 AVB Proxy Server

IP packets are forwarded with best effort in the public internet. The Soundjack cloud in contrast, provides a fully managed and controlled AVB Ethernet network. The FQTSS amendment to IEEE 802.1Q prevents bursty traffic by the means of a credit-based bandwidth shaper, inside of the Soundjack cloud network segment. A proxy server is used as a wave trap to devide large and erratic UDP datagrams into more and smaller AVTP packets, that maintain a constant inter packet gap. With the credit-based bandwidth shaper the AVTP packets can travel inside the cloud network segment in a deterministic way.

The AVB proxy server accepts and returns UDP streams from and to Soundjack users, that have been assigned by the session server. To keep the latency introduced by the service times of the Ethernet NIC low, only eight streams are assigned to an AVB proxy server.

The UDP streams received on the WAN interface need to be transmitted in the AVB network segment at a different bitrate with a different payloading. A UDP datagram of a Soundjack stream contains 256, 512 or 1024 Bytes of raw audio. Lower amounts of bytes occur in cases of compression according to the chosen compression ratios. The resulting AVTP stream is sent from the AVB proxy server to the AVB processing server, which processes the eight streams and sends them back as AVTP packets with the same, but processed payload. In the opposite streaming direction, the AVB proxy waits untill sufficient AVTP packets are in a listener's circular buffer, constructs an UDP datagram and sends it back to the client, where it came from.

2.3.2 AVB Processing Server

The AVB processing server receives the audio and video streams from the AVB proxy server with a constant packet rate of $8kHz$. It executes signal processing applications for audio and video data.

Conventional audio signal processing like compression or equalization can be integrated with existing Linux tools, such as JACK [23]. JACK provides a realtime processing environment that is required to execute some DSP al-

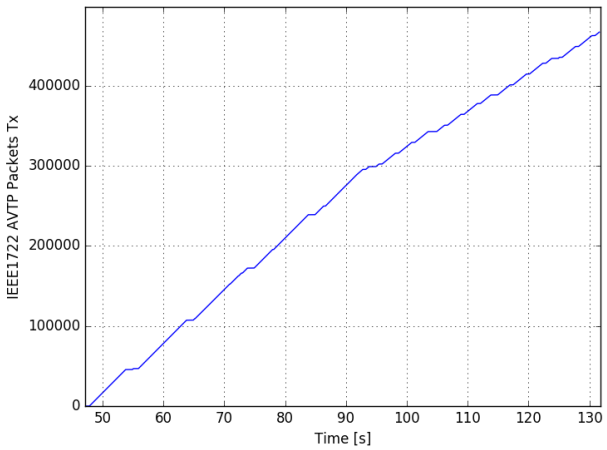

Figure 5: AVTP Stream Packets per Time

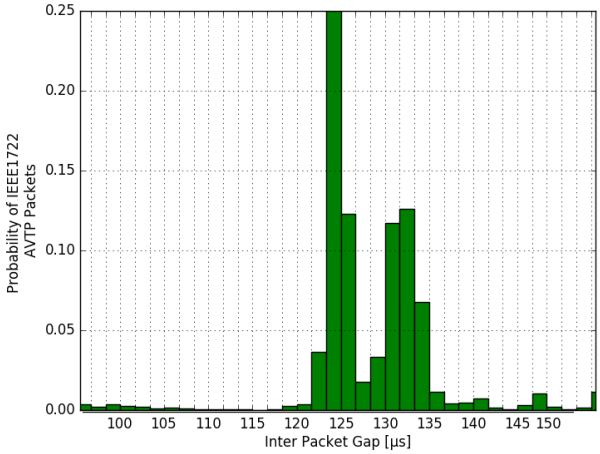

Figure 6: AVTP Stream Inter Packet Gap Probability Distribution

gorithms with LV2 plugins [24] or FAUST [25] applications. The design of the JACK server together with the two required JACK clients is shown in fig. 4. Before the other processes are forked, the main process starts the JACK Server. A Focusrite Scarlett Solo Gen2 [26] is used as audio interface hardware for the JACK server. Until now, JACK is running out of sync with the AVTP streams. After the forking of the listener and the talker processes, each process creates a JACK client. JACK ringbuffers are used by either client to communicate with the de-/packetizer threads respectively.

The JACK clients JACK ports are configured by the AVDECC process by means of the SET_STREAM_FORMAT [10, p.174] AEM command. JACK ringbuffers are created according to the channel count and sample format of the AEM command. This implies that channel count and sample format can only be changed before a AVB server session is established.

The listener AVTP depacketizer threads push audio samples from AVTP packets to the corresponding JACK ringbuffer, while the listener parent process pops the audio samples from the JACK ringbuffer and copies it to the JACK process graph. Audio samples are copied from the JACK process graph to the parent talker process, which in turn pushes the audio samples into the JACK ringbuffers of the talker AVTP packetizer threads.

The signal processing applications are connected to the talker and listener threads via JACK connections to the respective JACK clients.

Realtime audio production environments generally do not use graphics cards, as long as they are not involved in 3D rendering or video production processes. Thus, the graphics card is idle most of the time and can be utilized as an audio co-processor. Graphics card technologies made a lot of progress over the past years, which make modern graphics cards useable as co-processors for realtime signal processing [27]. More complex algorithms for processing audio and video data than the ones mentioned above shall be processed with a graphics card. Further applications such as a Viterbi decoder or virtual soundscapes and environments however, are still under development.

3 Evaluation and Discussion

For the evaluation of a general concept for a signal processing infrastructure no signal processing applications are applied yet, instead the JACK server creates loopback connections to allow the round trip transportation latencies and jitter to be tested. The current state of the AVB server application has a buffer leak, which leads to buffer overruns after ≈92 sec, as shown in fig. 5. The curve bends and the gradient decreases after this point, i.e. the IPG increases. This event marks the turning point between the two peaks exhibited the probability distribution of the transmitted AVTP packets shown in fig. 6 at 124μsec and 131μsec. Although the inter packet gaps of the AVTP stream are within reasonable bounds, the mean value of the PDF of 129.08μsec obviously cannot meet the defined inter packet gap for a SRP class A domain of 125μsec. The source of the buffer leak could not be determined yet.

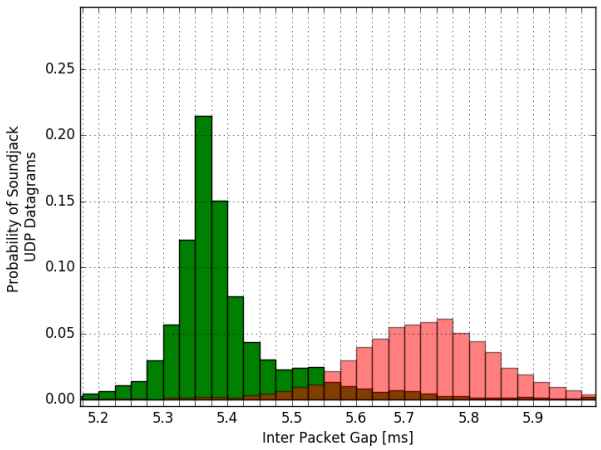

Figure 7: UDP Rx and Tx Inter Packet Gap Probability Distributions

An indicator for the source of the problem is shown in fig. 5. This figure shows the total amount of packets sent over time. The regions with a non-zero gradient show the constant flow of packets, while the gradients with value zero indicate some interrupt of the packet flow. This means that the talkers do not transmit for some period of time, which corresponds to the observation that the SRP states of the used switch ports toggle between listener ready and ask fail states.

Hence, no significant jitter and latency measurements could be done. Nonetheless, the magnitude of the observed end-to-end jitter and latency could be determined. The latency under this circumstances changes drastically when the buffers overrun from below 10msec to $300-600$msec. Figure 7 shows the jitter of the Soundjack clients transmit UDP stream (green distribution) has a mean value of 5.35msec, which corresponds to 256 audio samples per UDP datagram. When leaving the Soundjack cloud, the Soundjack clients receive stream (red distribution) has a mean value of 5.75msec and a higher standard deviation.

The JACK server was running with 64 samples per period at 48kHz without causing xruns during the measurements.

4 Conclusions

The integration of the JACK audio server alongside two JACK clients into the multiprocessing software architecture of the AVB server went very well, although an already known but undocumented bug with libjackserver and libjack [28] required resolving. Only the feature to dynamically change the channel count of a Soundjack stream during the transmission had to be deactivated.
A buffer leak that could not be resolved yet, is accountable for an increasing round trip latency. The jitter and latency of the end-to-end UDP streams do not provide significant measurements yet, but the observed transmission behaviour is very close to the bounds defined in the AVB standards.

5 Future Work

The ongoing work is related to the localization of the buffer leak. Latencies and jitter can only then be evaluated in scenarios, where actual signal processing applications are applied to the audio streams. For the operation under heavy load with all AVB endpoints registered, the EDF scheduling has to be configured properly. It also might be neccessary to upgrade hardware components such as the CPU, since realtime computing by itself requires a lot of CPU utilization and leads to overhead by process context switching. Another item to be handled in the furture is the synchronization of the JACK server to the mediaclock stream.

6 Acknowledgements

fast-music is part of the fast-project cluster (fast actuators sensors & transceivers), which is funded by the BMBF (Bundesministerium für Bildung und Forschung).

References

[1] (2018, Apr. 23) Soundjack - a realtime communication solution. [Online]. Available: http://http://www.soundjack.eu

[2] A. Carôt, "Musical telepresence - a comprehensive analysis towards new cognitive and technical approaches," Ph.D. dissertation, University of Lübeck, Germany, May 2009.

[3] (2018, Apr. 23) Genuin classics gbr, genuin recording group gbr. 04105 Leipzig, Germany. [Online]. Available: http://genuin.de

[4] (2018, Apr. 23) Symonics gmbh. 72144 Dusslingen, Germany. [Online]. Available: http://symonics.de

[5] H. Zimmermann, "Osi reference model - the iso model of architecture for open systems interconnection," in *IEEE Transac-*

tions on Communications, Vol. 28, No. 4, Apr. 1980, pp. 425–432.

[6] *Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks*, IEEE Std. 802.1AS, Mar. 2011.

[7] *Virtual Bridged Local Area Networks - Amendment 14: Stream Reservation Protocol (SRP)*, IEEE Std. 802.1Qat-2010, Sep. 2010.

[8] *Virtual Bridged Local Area Networks - Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams*, IEEE Std. 802.1Qav-2009, Jan. 2010.

[9] *Layer 2 Transport Protocol for Time-Sensitive Applications in Bridged Local Area Networks*, IEEE Std. 1722, May 2011.

[10] *Device Discovery, Connection Management, and Control Protocol for IEEE 1722 Based Devices*, IEEE Std. 17 221, Aug. 2013.

[11] A. Alliance. (2018, Apr. 23) Openavnu - an avnu sponsored repository for time sensitive network (tsn and avb) technology. [Online]. Available: https://github.com/AVnu/OpenAvnu/

[12] (2018, Apr. 23) Xmos ltd. / attero tech inc. [Online]. Available: http://www.atterodesign.com/cobranet-oem-products/xmos-avb-module/

[13] W. R. Stevens, B. Fenner, and A. M. Rudoff, *UNIX Network Programming, Vol. 1*, 3rd ed. Pearson Education, 2003.

[14] J. Kacur, "Realtime kernel for audio and visual applications," in *Proceedings of the Linux Audio Conference 2010*. Wittenburg, DE: Red Hat, Apr. 2010.

[15] (2018, Apr. 23) Howto: Build an rt-application. [Online]. Available: https://rt.wiki.kernel.org/index.php/HOWTO:_Build_an_RT-application

[16] (2018, Apr. 23) Linux programmer's manual sched(7). [Online]. Available: http://man7.org/linux/man-pages/man7/sched.7.html

[17] K. J. e. a. Ion Stoica, Hussein Abdel-Wahab, "A proportional share resource allocation algorithm for real-time, time-shared systems," in *Proceedings of the IEEE*, 1996.

[18] G. B. Luca Abeni, "Integrating multimedia applications in hard real-time systems," in *Proceedings of the 19th Real-Time System Symposium (RTSS 1998)*. Madrid, ESP: Scuola Superiore S. Anna, Pisa, Dec. 4–13, 1998.

[19] J. Corbet, A. Rubini, and G. Kroah-Hartman, *Linux Device Drivers, 3rd Edition*. O'Reilly Media, Inc., 2005.

[20] J. JONGEPIER, "Configuring your system for realtime low latency audio processing," in *Proceedings of the Linux Audio Conference 2011*. ICTE department Faculty of Humanities, University of Amsterdam, 2011.

[21] S. McCanne and V. Jacobson, "The bsd packet filter: A new architecture for user-level packet capture," in *Presented at the 1993 Winter USENIX conference*. San Diego, CA: Lawrence Berkeley Laboratory, One Cyclotron Road, Berkeley, CA, Jan. 25–29, 1993.

[22] (2018, Apr. 23) Dynamic memory allocation example. [Online]. Available: https://rt.wiki.kernel.org/index.php/Dynamic_memory_allocation_example

[23] (2018, Apr. 23) Jack audio connection kit. [Online]. Available: https://jackaudio.org

[24] (2018, Apr. 23) Lv2 - open standard for audio plugins. [Online]. Available: http://www.lv2plug.in

[25] (2018, Apr. 23) Faust programming language. [Online]. Available: http://faust.grame.fr/

[26] (2018, Apr. 23) Focusrite audio engineering ltd. United Kingdom. [Online]. Available: https://us.focusrite.com/usb-audio-interfaces/scarlett-solo

[27] C. Kuhr and A. Carôt, "Evaluation of data transfer methods for block-based realtime audio processing with cuda," in *Proceedings of the 10th Forum Media Technology and 3rd All Around Audio Symposium*. St. Pölten, Austria: St Pölten University of Applied Sciences, Nov. 71–76, 2017.

[28] C. Kuhr. (2018, Mar. 06) Undocumented crash when using libjack and libjackserver #331. [Online]. Available: https://github.com/jackaudio/jack2/issues/331