# Synchronized Real Time Audio Streaming Over Ethernet in Embedded Systems

Ms. Indumathi Duraipandian
Faculty of Computer Science and Electrical Engineering
Kiel University of Applied Sciences
Kiel, Germany
indu9086@gmail.com

Prof. Dr. Robert Manzke
Faculty of Computer Science and Electrical Engineering
Kiel University of Applied Sciences
Kiel, Germany
robert.manzke@fh-kiel.de

***Abstract* — In any complex audio video networks such as professional audio recording studios, automotive or in-flight infotainment systems, concert avenues or even home entertainment systems, the connection between the various audio/video sources and sinks are mostly analog, point to point and serve a single purpose. This leads to tones of confusing cables, each cable serving a specific data exchange. Even the digital solutions such as I2S, S/PDIF, AES3, MOST, Firewire (IEEE 1394), HDMI or audio over USB etc., still require purpose-built cables and proprietary software to work correctly and yet they still lack interoperability.**

**Ethernet is now ubiquitous and offers high bandwidth for low cost over reasonable distances. But the limitation of Ethernet networks is that, since it is a packet switched network, offering reliable, real time delivery of media is a challenge. But a new set of protocol is being developed for synchronized real time streaming in Ethernet and are collectively called the Audio Video Bridging(AVB). This is a relatively new standard and currently there are only a handful of implementations that make use of this standard.**

**This paper discusses the implementation of AVB stack as a part of the open source Linux kernel and for an open source hardware Beagle Board. And then study the various metrics such as latency, synchronization, throughput etc... between devices like Beagle Bone Black and Beagle Board X15.**

***Keywords — AVB, Embedded Systems, Real time audio, Audio over Ethernet, Beagle board***

## I. INTRODUCTION

In the current "Information Age" we are constantly surrounded by all kinds of information processing systems. A most significant part of the information that we produce and consume is in the form of Audio/Video. Although much can be said about the audio/video information itself, here we concentrate more on the systems that are used in the production, distribution and consumption of these audio/video data. More specifically we are concerned in the embedded systems that are involved inside the several audio/video systems anywhere in the production-distribution-consumption chain. For any audio video system there are several parameters which needs to be considered when designing and building a system. Some important parameters are Bandwidth, Latency and Synchronization. Any of the current technologies for audio/video interlink such as I2S, S/PDIF, AES3, MOST, Firewire (IEEE 1394), HDMI or audio over USB etc., [9] [10] are purpose-built solutions which fit for only some specific problem. For example, HDMI may provide high bandwidth, reliable, resilient media streaming, but it still is not suitable real time requirements and it is not portable since additional purpose-built hardware, connectors and cables are required.

These days there is a rise of digital general-purpose data transfer solutions such as USB, Firewire, Thunderbolt, Ethernet etc., which provide very high bandwidth for a fractional cost and they are portable. The only downside is that since they are general purpose they don't really support the real time requirements of media streaming. Among these several of the digital general-purpose connections, Ethernet has several inherent advantages. Because of this reason there are several protocols developed under the "Audio over Ethernet" model, some noteworthy protocols among them are COBRANET, Ethernet, Ravenna, AES67, Dante and AVB/TSN [8]. And among these protocols Dante is a mature protocol which has been well adopted by the industry and supported by several manufacturers. But Dante is a closed and proprietary system developed by Audinate Inc., which implies royalties, no flexibility in improvements and several other constraints.

Audio Video Bridging is a collection of protocols developed by the Institute of Electrical and Electronics Engineers (IEEE) Audio Video Bridging Task Group of the IEEE 802.1 standards committee, to modify and to ensure that media streaming over Ethernet networks can meet the real time requirements and have reliable and low latency with high level of synchronization. In this study we have developed the complete AVB stack on an open software and hardware combination (Linux on Beagle Board) and use this as a platform to evaluate the various operating parameters of a AVB Ethernet network such as latency, synchronization, throughput etc.

## II. PROTOTYPE IMPLEMENTATION

The choice of hardware for any AVB implementation depends on the support of "hardware timestamping" feature in the Ethernet Physical or Media Access Control layers. This feature is required to precisely timestamp the incoming and outgoing ethernet packets for the precise time delay measurements between two devices.

The beagle board is an open hardware platform which provides high performance in low cost with several in built peripherals and options to extend the hardware with expansion headers [1]. Thus, the beagle board is a best fit for any embedded real time audio streaming applications. Also, it is based on the Texas Instruments AM335x Sitara line of processors which has an inbuilt Ethernet MAC with the support of Hardware timestamping. The beagle board range of open hardware comes in various configurations from the low-end beagle bones and high-end beagle board x15s making them suitable for several applications.

Similar to the constraint in the choice of the hardware, the choice of the software also depends on the support for hardware timestamping in the Ethernet drivers. Although it is

possible to implement a fully custom solution including custom Ethernet drivers with hardware timestamp support, the Linux operating systems is chosen to make sure of a open software platform which is modular and portable and also has support for the chosen hardware platform.

The AVB protocols are developed as separate software modules in the Linux system. The AVB driver is developed as virtual audio device driver [2] in the Linux ALSA (Advanced Linux Sound Architecture) to make sure the existing audio applications can directly communicate via the AVB interface by just directly using the ALSA APIs for Linux. A block diagram for the AVB software stack is given in the Figure 1.

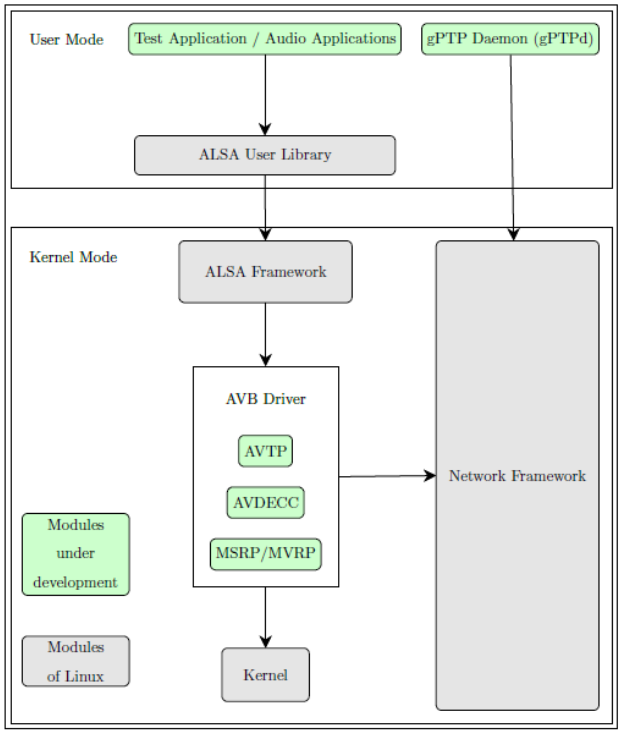


Figure 1 - AVB Software Stack

The Software stack [4] can be subdivided into three broad categories which are listed and described below,

- The gPTP Daemon (gPTPd) is a user level Linux Daemon to handle the Generalized Precision Time Protocol [3] operation to Measure the Peer to Peer delay and Clock Synchronization in the AVB Ethernet Network.
- The AVB Virtual ALSA driver is a ALSA device driver without no associated hardware. It is implemented in the ALSA framework in the Linux kernel. It executes the setup and handling of a AVB audio streaming.
- The test application is a Linux user level program which uses the AVB ALSA virtual driver to stream audio content over the AVB Ethernet network [5].

## III. Evaluation Setup

AVB devices can be tested in two ways. One way is to connect a stream source to a stream sink via a AVB aware network switch. The second way is to directly connect a AVB stream source to a AVB stream sink via a cross Ethernet cable. The evaluation setup used here falls into the second category. Two beagle board devices, one beagle bone and a beagle board X15 are connected via a cross Ethernet cable and the regular audio in and audio out of these devices are connected to a test PC for testing and evaluation. The test setup can be seen in the Figure 2.

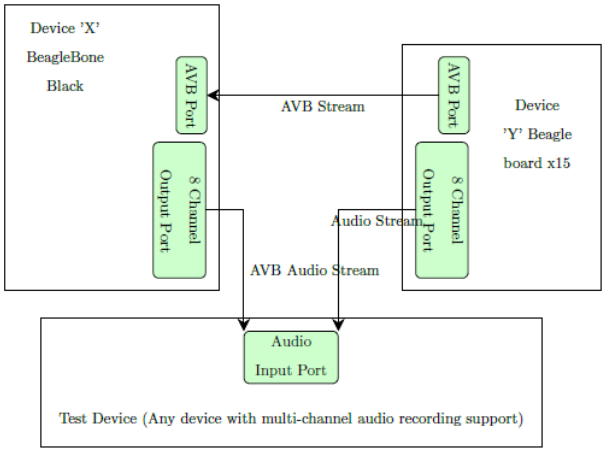


Figure 2 - Evaluation Setup

In such test setup the synchronization of the AVB streaming can be tested. The device Y streams some audio content over AVB for playback at a future time and then streams the same audio content at the predefined future time also over its analog audio output. The device X, upon reception of the AVB audio stream, caches it until the specified stream time is reached and then playbacks the audio content over its analog audio output. The test device can measure the synchronization by recording both streams and measuring the delay between them.

This test setup is also used to measure the other parameters such as the gPTP delay variation, Clock drift, synchronization accuracy and Latency.

## IV. Results

### A. *gPTP Delay Variation*

The delay variation is defined as the variance the successive peer to peer delay measurement values in the gPTP protocol. Although the delay measurement process follows the same procedure every time, random variations in the time stamping (since the normal crystal clocks used are not perfect) leads to slightly different delay values for every measurement. In ideal case the variance should be zero and the delay should be the same for every measurement, but in normal working conditions this will not be the case. The aim here is that the delay variance should be as small as possible. Since the clock synchronization between the two devices depends on the measured delay, when the delay variation is small the clock synchronization accuracy be high.

From the graph in Figure 3 the average value for the measured delay value is calculated as 841.88 ns and the average standard deviation is calculated as 8.89 ns. Also,

from the normal distribution 6.2 of the measured delay values we can see that it approximates a bell curve (i.e. normal distribution) indicating that the measured delay values follow a natural normal distribution with no other external influences.

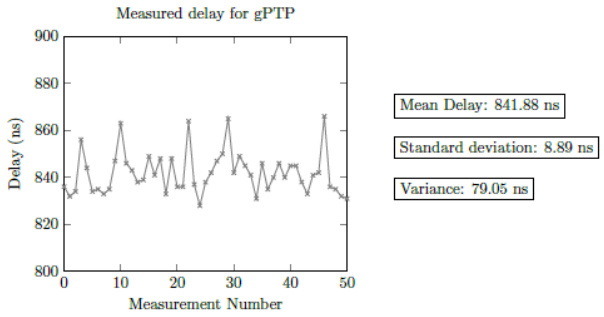


Figure 3 - Delay Variance

From the above discussions it can be decided that the measured delay values do not influence the clock synchronization too much as the variance is only in the order of 10 ns which is negligible for most audio applications. Also, this delay variance is much smaller than other parameters which influence the synchronization.

### B. *Clock Drift*

Clock drift is defined as the amount a clock drifts away from another clock, from a defined, point in time when the both clocks have the same time value. Since the frequency of the crystal oscillators used in the embedded systems as clock sources are highly unstable, it increases the clock drift. Crystal oscillators are highly influenced by environmental factors such as heat, stress, moisture and inherent factors such as age. Since clock drift affects the clock synchronization it has to be kept low for synchronized audio streaming.

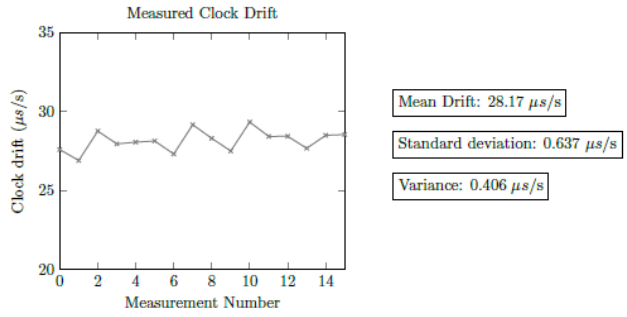


Figure 4 - Clock Drift

As in real embedded system crystal oscillator-based clocks the clock drift cannot be avoided, to has to be mitigated by some other means. This is covered by the gPTP protocol synchronization process. By periodically synchronizing the slave devices to the grand master device, the clock drift is reset periodically, and the synchronization is re-established. So, the maximum drift the slave clock encounters are limited to the amount of drift that is accumulated during the time before the next synchronization.

Based on the current default sync repetition time of 32 seconds, the maximum clock drift for the slave device can be up-to a maximum value of 900 µs. Choosing a smaller sync repetition time will limit this clock drift value to a lower value. Although this value can affect the media synchronization it is limited to the sub milli-second range which is still low enough to be perceived in real world scenario.

### C. *Synchronization Accuracy*

The synchronization accuracy is defined as the time delta between the streams output by two devices. It is measured by the test setup described in the previous section. The test PC measures the difference in the position of the recorded stream samples from both the devices. From such a measurement a delay of 15 samples was measured. So, for a 48kHz sampling rate of the stream this translates into 312.5 µs. As it is generally accepted that any audio delay less than 10 ms is not perceivable to the human ear [6] [7], the synchronization accuracy of less than 1 ms is acceptable for most streaming applications.

### D. *Latency*

The latency is a measure of the time it takes for the samples to reach the destination from the time they are transmitted. For a real time, audio streaming system, the latency should be as low as possible. The test application is modified accordingly to measure the latency between the devices when the AVB streaming is active. A latency of 18.46 ms was measured.

Although the measured latency of 18.46 ms is not sufficient for real time audio streaming in live low latency applications, it is just almost sufficient for most of the real time audio streaming for media applications. But if more lower latencies are required the size of the hardware buffer has to be decreased. But decreasing the size of the hardware buffer can result in frequent under-flows in the streaming which can lead to audio breaks. So, a fine balance is required to trade-off between low latency and reliable and continuous audio streaming.

## V. CONCLUSION

To summarize, AVB was proposed as a solution for a synchronized, real time audio streaming in embedded systems. The proposed solution is researched, it's technical details are studied, and a software solution is designed and implemented. Beagle bone devices, BeagleBone black and BeagleBoard x15 are chosen as a hardware platform for evaluating the solution. The implemented AVB solution is evaluated on these beagle board devices. By analyzing the results of the evaluations, it is concluded that synchronized, real time audio streaming is possible in embedded systems using the proposed AVB software solution and the beagle board devices are sufficient in providing an environment for a stable and synchronized hardware clock through gPTP and sufficient bandwidth through the integrated Ethernet ports. And because of their small form factor, power and support for a range of Capes that extend their functionality, the combination of the beagle board devices plus the proposed AVB solution can provide a platform for synchronized real time audio streaming applications.

## VI. References


[1] beagleboard.org, “Beagle Bone” [Online]. Available: https://beagleboard.org/

[2] A. Rubini and J. Corbet, Linux Device Drivers, 2nd Edition. O'Reilly&Associates Inc, June 2001..

[3] IEEE, “IEEE standard for a precision clock synchronization protocol for networked measurement and control systems," IEEE Std 1588-2008, 2008..

[4] Beagleboard, “GSoC - Beaglebone AVB Stack,” 2017. [Online]. Available at: https://elinux.org/BeagleBoard/GSoC/2017 Projects#Project: BeagleBone AVB Stack.

[5] ----, “GSoC - Beaglebone AVB Stack Wiki,” 2017. [Online]. Available: https://elinux.org/BeagleBoard/GSoC/BeagleBoneAVB.

[6] C. E. Ralf Steinmetz, “Human perception of media synchronization,” Technical Report 43.9310, IBM European Networking Center Heidelberg, Germany, 1993..

[7] J. Deber, R. Jota, C. Forlines, and D. Wigdor, “How much faster is fast enough?” 33rd Annual ACM Conference, pp. 1827-1836, 04 2015..

[8] A. Inc., “Dante Overview,” 2018. [Online]. Available: https://www.audinate.com/solutions/dante-overview

[9] B. McCarthy, Sound Systems: Design and Optimization: Modern Techniques and Tools for Sound System Design and Alignment. Taylor & Francis, 2016. [Online]. Available: https://books.google.de/books?id=FMejCwAAQBAJ

[10] M. Walker, “Choosing an audio interface,” 2008. [Online]. Available: https://www.soundonsound.com/sound-advice/choosing-audio-interface