GENAVBTSNUG

GenAVB/TSN Stack Evaluation User Guide

Rev. 1.8 — 15 December 2025 **User guide**

Document information

Information	Content
Keywords	GENAVBTSNUG, GenAVB/TSN Stack, Audio Video Bridging (AVB), Time Sensitive Network (TSN), NXP hardware platforms (i.MX 6ULL EVK, i.MX 8M Plus EVK, i.MX 8M Mini EVK, i.MX 8DXL EVK, i.MX 93 EVK, and i.MX 93 14x14 EVK boards), i.MX audio amplifier, TSN endpoint, AVB endpoint
Abstract	This document provides information on how to set up Audio Video Bridging evaluation experiments of the GenAVB/TSN Stack on supported NXP hardware platforms.

1 Introduction

The GenAVB/TSN Stack is a set of software components that provide Audio Video Bridging (AVB) and Time Sensitive Network (TSN) functionality on NXP SoC and hardware platforms. This document provides information on how to set up Audio Video Bridging evaluation experiments of the GenAVB/TSN Stack. It describes the AVB use cases, the hardware and setup requirements, evaluation instructions, and steps to configure the evaluation software, and checking the results.

1.1 Related documentation

For additional information related to Real Time Edge, refer to the URL below: REALTIME EDGE Documentation. The following documents are available:

- *Real-time Edge Yocto Project User Guide (RTEDGEYOCTOUG)* (provides steps using Yocto build environment)
- *Real Time Edge User Guide (REALTIMEEDGEUG)* (provides information on how to use the supported real-time edge features on NXP platforms)
- Real-time Edge Software Release Notes (RN00161) (provides important information about Real-time Edge software contents, supported features, known issues and limitations for the release)
- *Harpoon User's Guide (HRPNUG)* (provides information to build Harpoon Yocto images)
- *i.MX6ULL EVK GenAVB/TSN Rework Application Note* (*AN13678*)
- For details about the graphics feature available in i.MX 8M Plus and i.MX 8M Mini boards, refer to the i.MX Graphics User's Guide (*https://www.nxp.com/docs/en/user-guide/UG10159.pdf*).

Refer to the following guides for detailed instructions on booting up and setting up the relevant boards.

- *i.MX 6ULL EVK Quick Start Guide*
- *i.MX 8M Mini LPDDR4 EVK Quick Start Guide*
- *i.MX 8M Plus LPDDR4 EVK Quick Start Guide*
- *LS1028ARDB Quick Start Guide*
- *LS1043ARDB Getting Started Guide*
- *LS1046ARDB Getting Started Guide*
- *LS1046AFRWY Getting Started Guide*
- *LX2160A/LX2160A-Rev2 RDB Quick Start Guide*
- *IMX93EVK Quick Start Guide*
- *i.MX8M Nano EVK Getting Started Guide*
- *i.MX8DXL Quick Start Guide*

2 Initial preparation

2.1 Evaluation boards description and supported roles

The GenAVB/TSN stack is supported on multiple SoCs (i.MX 6 and i.MX 8) and evaluation boards that differ in capabilities. For different use cases, the required connections differ, depending on the ports available on the evaluation boards. This section provides an overall description of the different evaluation boards and their available hardware ports and connections.

The following sections in this document refer to the i.MX evaluation boards depending on their roles:

- **i.MX Audio Amplifier:** A board that can render (and/or decode) audio samples to a speaker connected to a jack output port or an RCA Output port.
- **i.MX Audio Sampler:** A board that can capture audio samples from an analog device connected to an input jack port, MIC, or an RCA Input port.
- **i.MX Video Renderer:** A board that can (optionally) demux an MPEG2-TS stream and decode then render video frames to a display.
- **i.MX Audio Video Player:** A board that can demux a MPEG2-TS stream and decode both audio and video frames and render them to their correspondent outputs.
- **i.MX Audio Media Server:** A board that can read raw audio samples from a local media file and send them over the network.
- **i.MX Full Media Server:** A board that can read an encoded media file (for example, MP4/MPEG2-TS) and demuxes audio and video. It can also decode audio frames, send separate audio and video AVTP streams, and play the video frames in a local display simultaneously.

Each supported evaluation board can be used in a set of these roles. Refer to the User Manual of the respective board to ensure that your board can be used for the desired use case. Also refer to *AN13678* (*i.MX6ULL EVK GenAVB/TSN Rework Application Note* available on Real Time Edge Documentation) for additional information.

2.1.1 i.MX 6ULL EVK board

This evaluation board supports the following roles in the AVB evaluation uses cases described in this document:

- **i.MX Audio Amplifier:** using the Audio Jack as an output
- **i.MX Audio Sampler:** using the MIC as an input
- **i.MX Audio Media Server**

Figure 1 shows the AVB evaluation use case using i.MX 6ULL EVK board.

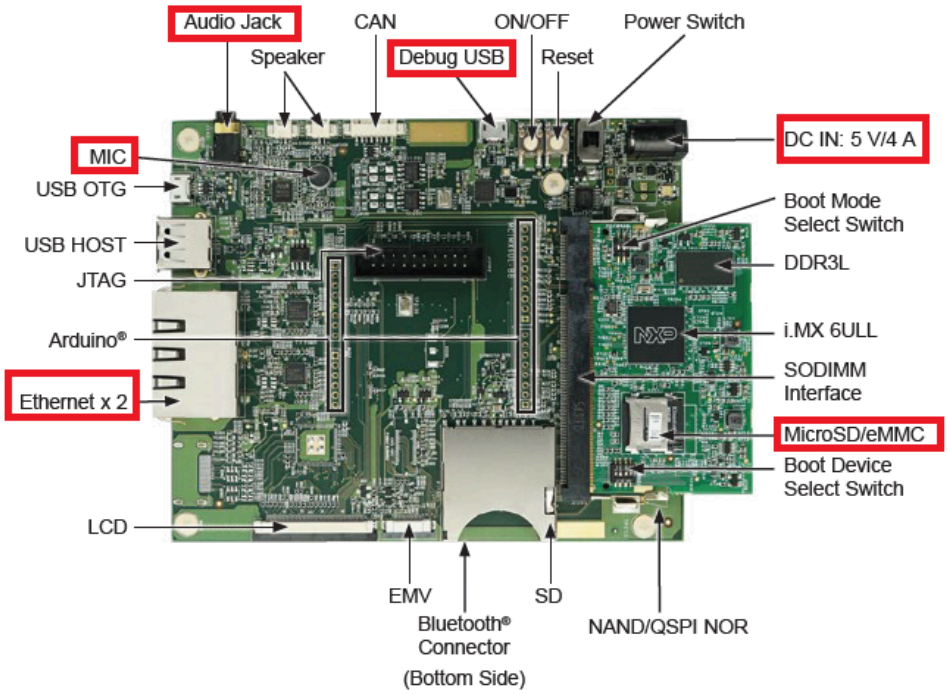

Figure 1. i.MX 6ULL EVK board

2.1.2 i.MX 8M Mini EVK board

This evaluation board supports the following roles in the AVB evaluation uses cases described in this document:

- **i.MX Audio Amplifier**: using the Audio Jack as output, **with software based media clock recovery**
- **i.MX Full Media Server:** using the HDMI port for local display
- **i.MX Audio Media Server**
- **i.MX Video Renderer:** using the HDMI port as output
- **i.MX Audio Video Player:** using the Audio Jack as audio output, **with software based media clock recovery,** and HDMI for video output

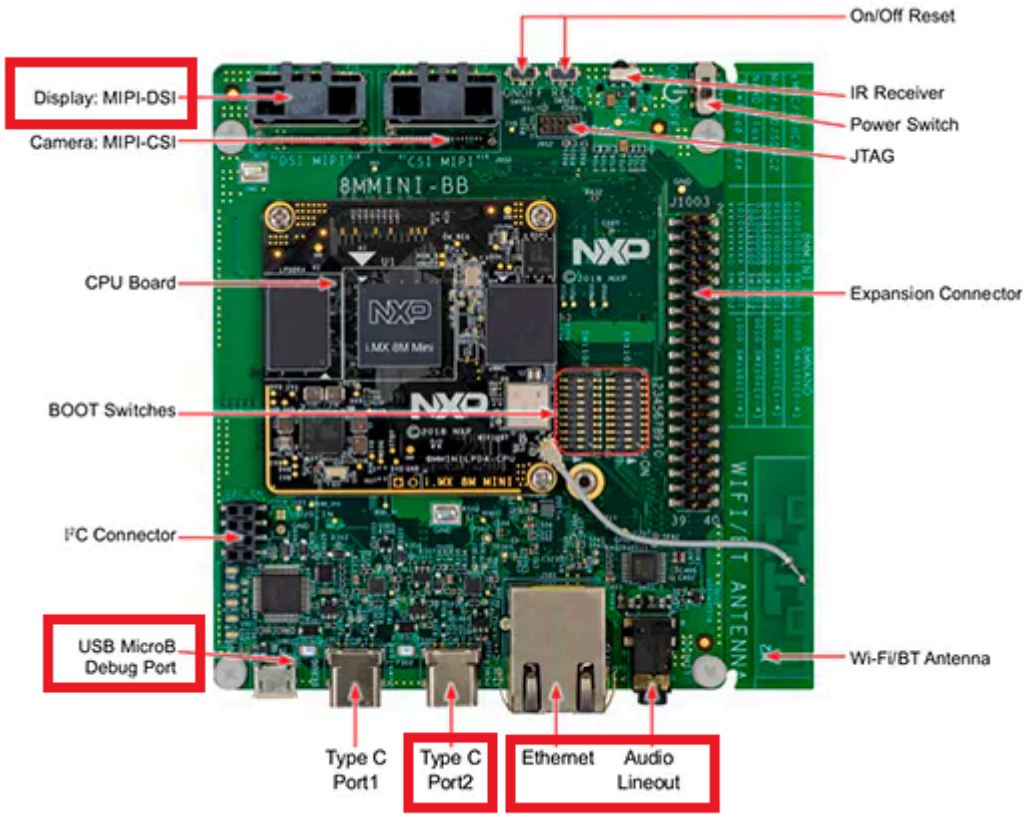

Figure 2. i.MX 8M Mini EVK board

Figure 2 shows the AVB evaluation use case using i.MX 8M Mini EVK board.

2.1.3 i.MX 8M Plus EVK board

This evaluation board supports the following roles in the AVB evaluation uses cases described in this document (refer Figure 3) :

- **i.MX Audio Amplifier**: using the Audio Jack as output with hardware-based media clock recovery.
- **i.MX Audio Sampler**: using an audio jack input with microphone feature (TRRS with four contacts).
- **i.MX Full Media Server**: using the HDMI port for local display.
- **i.MX Audio Media Server**.
- **i.MX Video Renderer**: using the HDMI port as output.
- **i.MX Audio Video Player**: using the Audio Jack as audio output, and HDMI for video output.

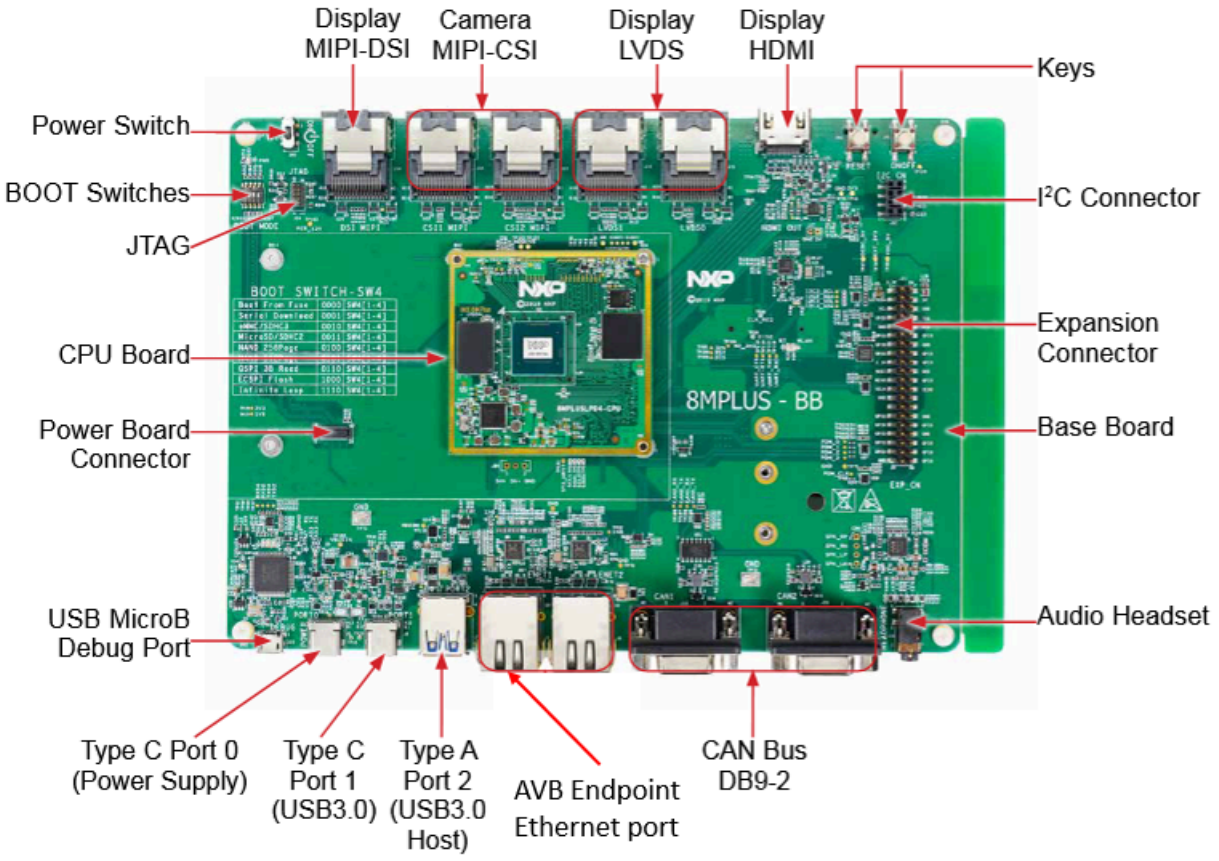

Figure 3. i.MX 8M Plus EVK board

2.1.4 i.MX 93 EVK board

This evaluation board supports the following roles in the AVB evaluation uses cases described in this document:

- **i.MX Audio Amplifier**: using the Audio Jack as output, with software-based media clock recovery.
- **i.MX Audio Sampler:** using an audio jack input with microphone feature (TRRS with four contacts).
- **i.MX Audio Media Server**.

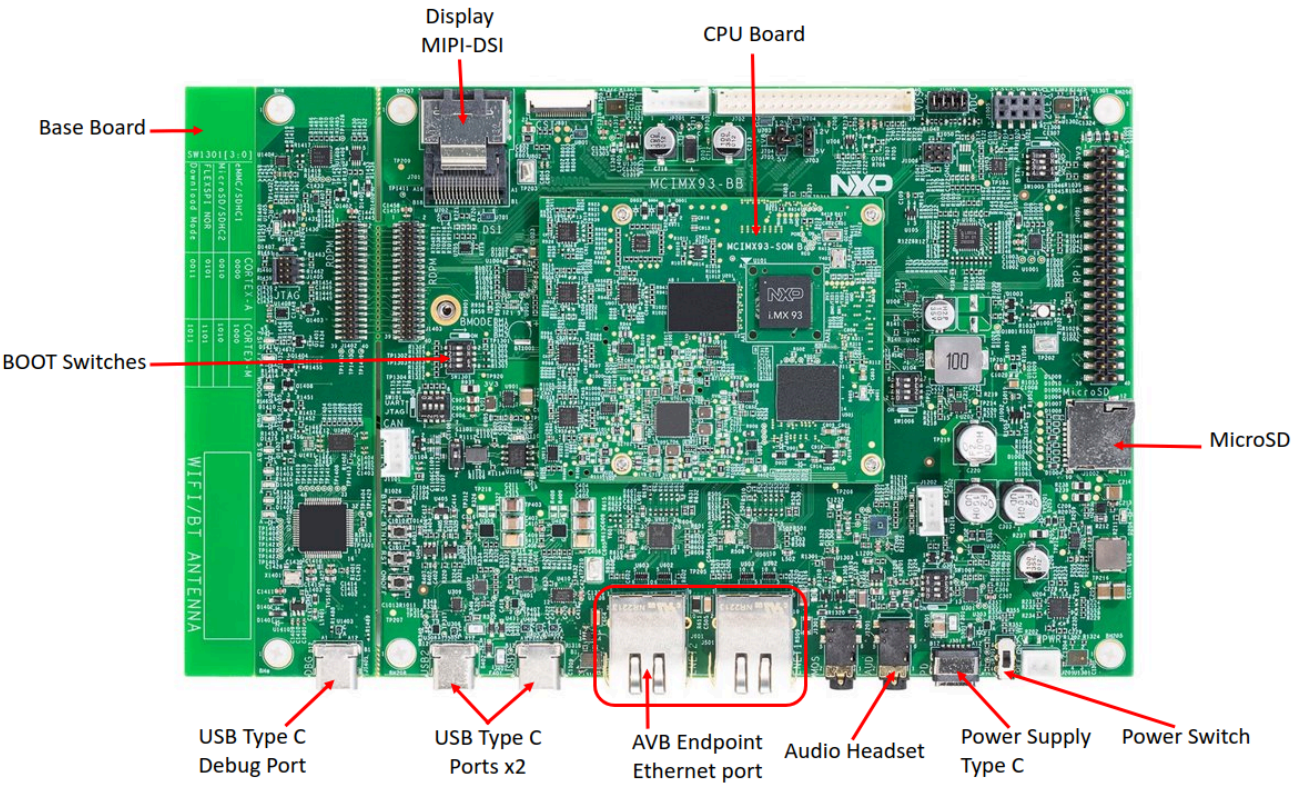

Figure 4. i.MX 93 EVK board

2.1.5 i.MX 93 14x14 EVK board

This evaluation board supports the following roles in the AVB evaluation uses cases described in this document:

- **i.MX Audio Amplifier**: This use case uses the MX93AUD-HAT audio evaluation platform for the AVB endpoint configuration. Two 8 Ω speakers using Medium Quality Sound (MQS) are used for the AVB hybrid configuration, with software-based media clock recovery.
- **i.MX Audio Sampler**: using the MX93AUD-HAT audio evaluation platform (only for the AVB endpoint configuration).
- **i.MX Audio Media Server**.

2.1.6 i.MX 8DXL EVK board

This evaluation board supports the following roles in the AVB evaluation uses cases described in this document:

- **i.MX Audio Amplifier**: using the Audio Jack as output, with hardware-based media clock recovery.
- **i.MX Audio Sampler**: using the Audio Jack as input with a microphone (TRRS with four contacts).
- **i.MX Audio Media Server**.

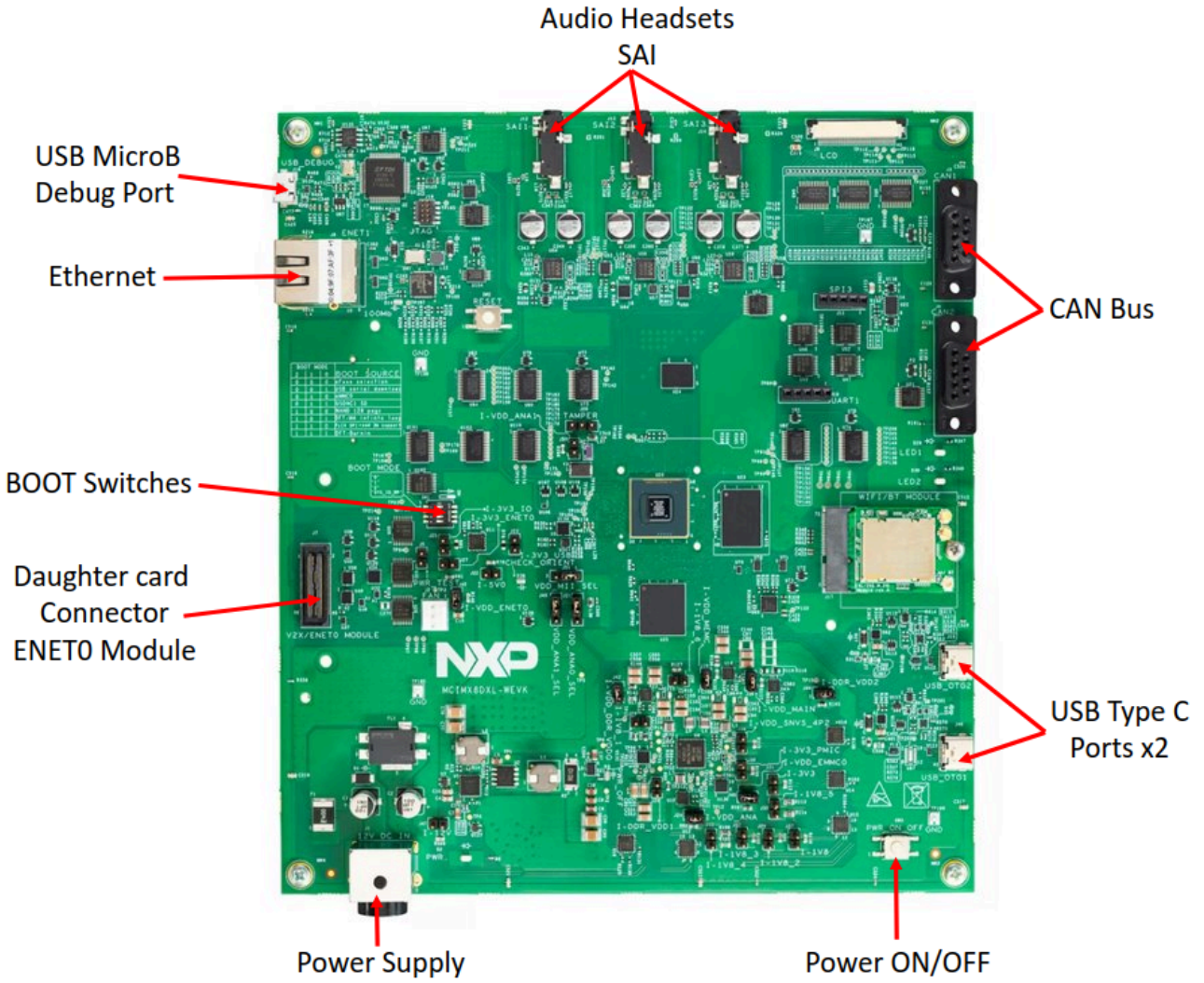

Figure 5. i.MX 8DXL EVK board

2.2 AVB configuration on evaluation boards

The AVB evaluation package can be configured on various reference boards. Certain specific settings must be performed for ensuring correct AVB operations.

2.2.1 i.MX 6ULL EVK board

This section describes AVB configuration on i.MX 6ULL EVK boards with and without media clock recovery.

2.2.1.1 i.MX 6ULL EVK board with media clock recovery

The i.MX 6ULL EVK board features the i.MX 6ULL processor (MCIMX6ULL-EVK) and can be used in the evaluation setup as an audio endpoint. The board must be modified to use it for a listener role and to support the media clock recovery process. Perform the steps below:

- Connect SD1_DATA2 and GPIO1_IO05 pads (can be done by connecting R1728 and TP2120)
- Connect JTAG_MOD and JTAG_TMS pads (can be done by connecting R1023 and JTAG PIN 7)

For details about the required hardware rework, refer to the *i.MX6ULL EVK GenAVB/TSN Rework Application Note* (*AN13678*) available on Real Time Edge Documentation.

The device binary tree includes the hardware description relative to the i.MX 6ULL EVK board. Ensure to update the boot parameters to use the device tree binary, by following the steps listed below:

- Power on the board and stop the automatic boot process by pressing the Space bar on the keyboard.
- Enter the following commands at the U-Boot prompt:

```
U-Boot > setenv fdt_file imx6ull-14x14-evk-avb-mcr.dtb
U-Boot > saveenv
U-Boot > boot
```

The change is saved across reboots.

2.2.1.2 i.MX 6ULL EVK board without media clock recovery

For a proper evaluation setup, it is highly recommended to implement the needed AVB hardware rework for the i.MX 6ULL EVK board.

However, a board without the rework can still run the supported evaluation setups with limited capabilities. In such a case, when used for an audio listener role, the media clock recovery is not available and media clock recovery is not launched.

The device binary tree includes the hardware description relative to the i.MX 6ULL EVK board. Ensure to update the boot parameters to use the device tree binary, by following the steps listed below:

- Power on the board and stop the automatic boot process by pressing the Space bar on the keyboard.
- Enter the following commands at the U-Boot prompt:

```
U-Boot > setenv fdt_file imx6ull-14x14-evk-avb.dtb
U-Boot > saveenv
U-Boot > boot
```

Note: *depending on the Evaluation Board Revision, select the right device tree in U-Boot:*

- `imx6ull-14x14-evk-reve-avb.dtb` : for i.MX 6ULL EVK RevE
- `imx6ull-14x14-evk-avb.dtb` : for previous board revisions

The change is saved across reboots.

2.2.2 i.MX 8M Mini EVK board

The i.MX 8M Mini EVK board (MCIMX8MM-EVK) featuring the i.MX 8M Mini processor can be used in any role of the AVB evaluation setup.

Due to a pin conflict, hardware-based media clock recovery cannot be implemented. Therefore, a fall-back mechanism based on software sampling of PTP and Audio PLL is used to perform media clock recovery.

Ensure to update the boot parameter to use the device tree binary that includes the hardware description relative to the i.MX 8MM EVK board:

- Power on the board and stop the automatic boot process by pressing the Space bar on the keyboard.
- Enter the following commands at the U-Boot prompt:

```
U-Boot > setenv fdtfile imx8mm-evk-avb.dtb
U-Boot > saveenv
U-Boot > boot
```

Note: *depending on the Evaluation Board Revision: REV B or REV C, select the right device tree in U-Boot:*

- `imx8mm-evk-revb-avb.dtb` : for i.MX8MM EVK REVB
- `imx8mm-evk-avb.dtb`: for i.MX8MM EVK REVC

The change is saved across reboots.

2.2.3 i.MX 8MP EVK board

The i.MX 8M Plus EVK board featuring the i.MX 8M Plus application processor can be used in any role of the AVB evaluation setup.

The needed connections for hardware-based media clock recovery are routed internally and no hardware rework is required.

Make sure to update the boot parameter to use the device tree binary that includes the hardware description relative to the i.MX 8M Plus EVK board:

- Power on the board and stop the automatic boot process by pressing the Space bar on the keyboard.
- Enter the following commands at the U-Boot prompt:

```
U-Boot > setenv fdtfile imx8mp-evk-avb.dtb
U-Boot > saveenv
U-Boot > boot
```

The change is saved across reboots.

2.2.4 i.MX 93 EVK board

The i.MX 93 EVK board featuring the i.MX 93 application processor can be used as an audio endpoint.

No hardware rework is yet available for this board. Therefore, the software based media clock recovery mechanism is used.

Make sure to update the boot parameter to use the device tree binary that includes the hardware description relative to the i.MX 93 EVK board:

- Power on the board and stop the automatic boot process by pressing the Space Bar on the keyboard.
- Enter the following commands at the U-Boot prompt:

```
U-Boot > setenv fdtfile imx93-11x11-evk-avb.dtb
U-Boot > saveenv
U-Boot > boot
```

The change is saved across reboots.

2.2.5 i.MX 93 14x14 EVK board

The i.MX 93 14x14 EVK board featuring the i.MX 93 14x14 application processor can be used as:

- An audio endpoint with an audio hat (MX93AUD-HAT) board
- An AVB bridge with SJA1105Q-EVB (Ethernet Switch and PHY Evaluation Board)
- An AVB hybrid with SJA1105Q-EVB and up to two 8 Ohm Speakers

Attention: *SJA1105Q-EVB uses the RGMII interface and the ENET connector on i.MX 93 14x14 EVK board is usually set to RMII by default. A Hardware rework is required to set the interface to RGMII (mount R267, R272, R275 and R278 and unmount R288 and R293). Refer to the board schematics for details.*

2.2.5.1 Using an Audio Hat (MX93AUD-HAT) board

The i.MX 93 14x14 EVK board used as an audio endpoint needs an audio extension board connected to the GPIO connector J16.

Connectors J10, J11, J12, and J13 of the Audio Hat board can be used as audio output and the connectors J8, J9 as audio input.

The MX93AUD-HAT should be used as an extension audio platform.

See Figure 6

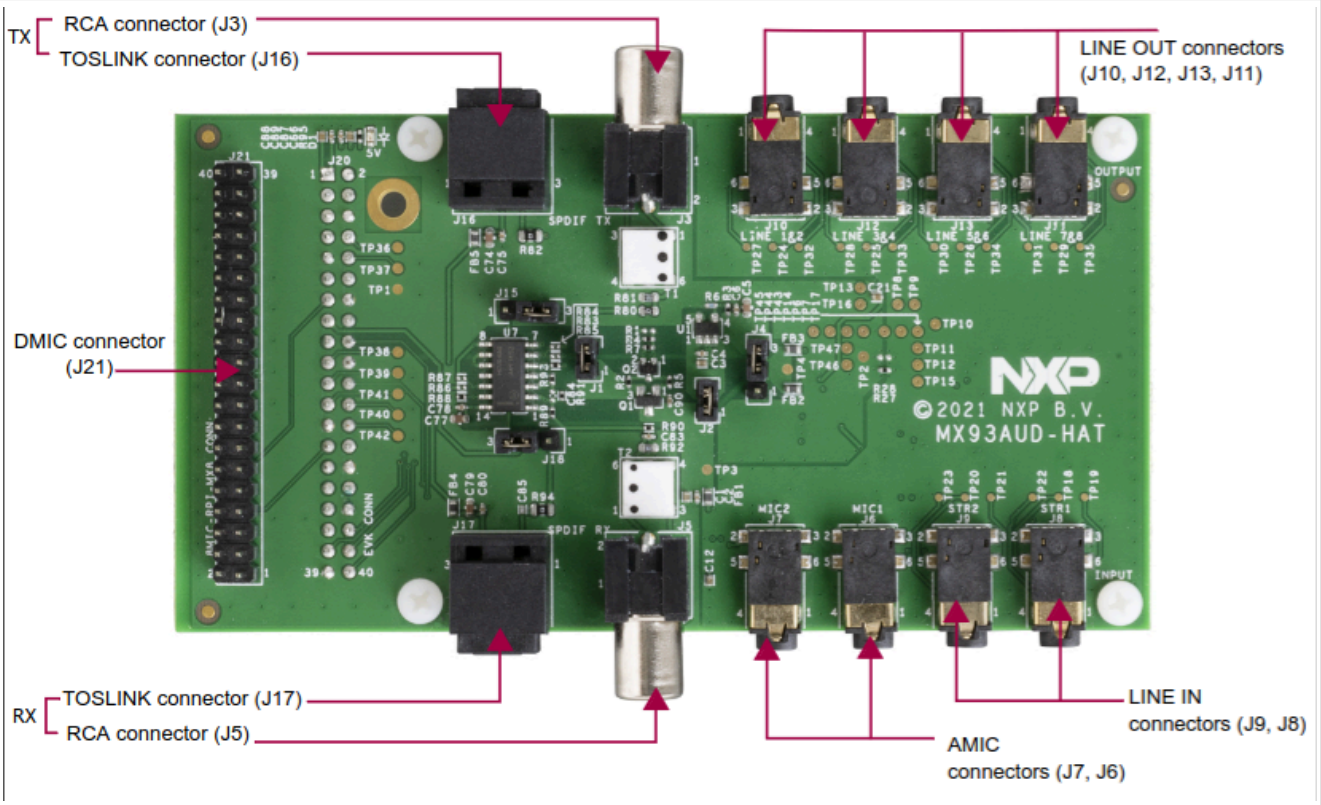

Figure 6. MX93AUD-HAT

Make sure to update the boot parameter to use the device tree binary that includes the hardware description relative to the i.MX 93 14x14 EVK board:

- Power on the board and stop the automatic boot process by pressing the Space Bar on the keyboard.
- Enter the following commands at the U-Boot prompt:

```
U-Boot > setenv fdtfile imx93-14x14-evk-aud-hat-avb.dtb
U-Boot > saveenv
U-Boot > boot
```

The change is saved across reboots.

2.2.5.2 Using SJA1105Q-EVB

The i.MX 93 14x14 EVK can be connected to the SJA1105Q-EVB board through the ENET2 interface. See Figure 7.

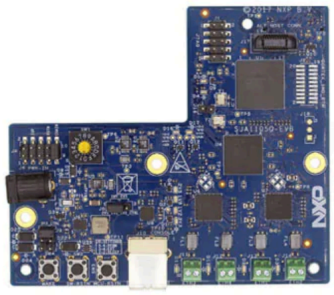

Figure 7. Ethernet Switch and PHY Evaluation Board (SJA1105Q-EVB)

2.2.5.2.1 Configuring i.MX 93 14x14 EVK as AVB Bridge

For the full configuration of the board as an AVB bridge, refer to "AVB Bridge" section in Real Time Edge User Guide .

2.2.5.2.2 Configuring i.MX 93 14x14 EVK as AVB Hybrid

This section describes the steps to configure the i.MX 93 14x14 EVK as AVB Hybrid.

1. Connect the ENET1 interface of the board to one of the SJA1105Q-EVB external ports through an Ethernet to BoardR-reach media converter. See Figure 8. The 8 Ω speakers can be connected to the i.MX 93 14x14 EVK connectors J22 and J23.

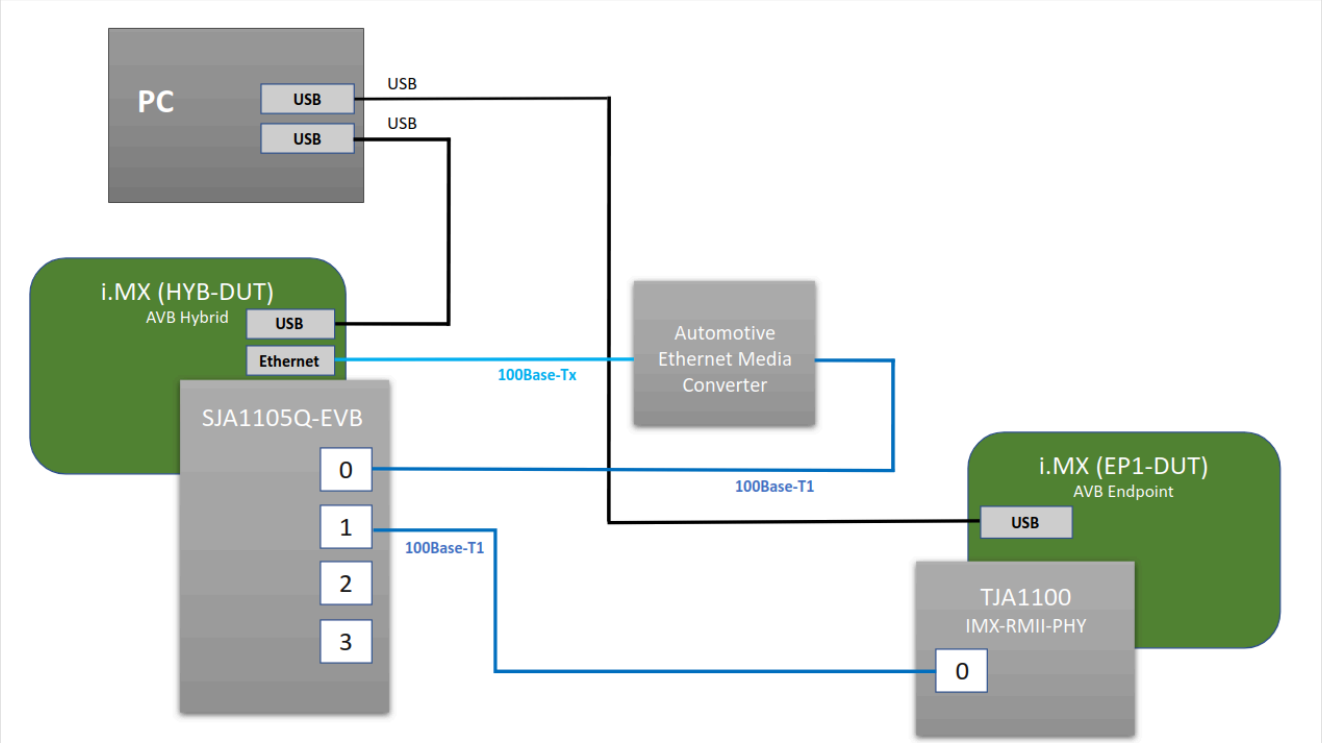

Figure 8. AVB Hybrid setup

2. Users must update the boot parameters to use the device tree binary that includes the hardware description relative to the i.MX 93 14x14 EVK board:
   - Power on the board and stop the automatic boot process by pressing the Space Bar on the keyboard.
   - Enter the following commands at the U-Boot prompt:

```
U-Boot > setenv fdtfile imx93-14x14-evk-sja1105-avb.dtb
U-Boot > saveenv
U-Boot > boot
```

3. Also, update the configuration files to specify the stack mode as "Hybrid AVB" and the port of the SJA1105Q-EVB on which the external link is connected to:
   - Power on the i.MX board and let the boot process complete.
   - Edit the stack configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config
```

   - Set the `GENAVB_TSN_CONFIG` parameter as "Hybrid AVB" using the value below:

```
GENAVB_TSN_CONFIG=4
```

   Note: *AVB endpoint application profiles can be configured in `/etc/genavb/config_avb` and is described in the following sections.*
   - Exit and save the file.
   - Edit the system configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/system.cfg.hybrid_avb
```

   - Set the port of the SJA1105Q-EVB that you have connected to the i.MX 93 14x14 ENET1 interface of the EVK:

```
bridge_hybrid_port = swpX
```

Note: *The mapping of the ports of SJA1105Q-EVB on i.MX93 14x14 EVK is:* `swp1, swp2, swp3, swp4.`

- Exit and save the file.

4. These changes are saved across reboots.
5. Then, configure the leader/follower role of the PHYs:
   - On the i.MX 93 14x14 EVK with SJA1105Q-EVB, check the default PHY role for the connected ports:

     ```
     ethtool swpX
     ```

     Note: *The default role is Leader (ethtool reports the forced-master under master-slave configuration).*
   - On the endpoint with TJA1100 PHY expansion card, set the PHY role accordingly (leader if SJA1105 port reports being follower and vice versa):

     ```
     ethtool -s eth0 master-slave forced-slave
     ```
   - On the automotive media converter, set the role using the same method.
6. Finally, before starting the GenAVB stack, always run the bridge configuration script using the following command at the Linux prompt:

   ```
   # avb-bridge.sh
   ```

2.2.6 i.MX 8DXL EVK board

The i.MX 8DXL EVK board featuring the i.MX 8DXL application processor can be used as:

- An audio endpoint with standard Ethernet PHY daughter card.
- An audio endpoint with automotive Ethernet PHY daughter card.
- An AVB bridge with SJA1105Q-EVB (Ethernet Switch and PHY Evaluation Board).

The evaluation board has a single built-in PHY connected to ENET1. As AVB endpoint is supported only on ENET0, an additional Ethernet PHY daughter card is required. Moreover, when using the PHY daughter card or an SJA1105Q-EVB board with ENET0, the SD card slot cannot be used to boot the image. Therefore, ensure to flash the image and boot from eMMC.

For details on how to flash images on eMMC and boot them, refer to the *Section 4.2 Universal Update Utility*" of i.MX Linux User's Guide (https://www.nxp.com/docs/en/user-guide/UG10163.pdf).

2.2.6.1 Using a standard Ethernet PHY expansion card

To support AVB endpoint use case with standard Ethernet, an additional Ethernet PHY expansion card (IMXAI2ETH-ATH) is required. See Figure 9.

Figure 9. Atheros Ethernet add-on card (IMXAI2ETH-ATH)

Make sure to update the boot parameter to use the device tree binary that includes the hardware description relative to the i.MX 8DXL EVK board (Using ENET0):

- Power on the board and stop the automatic boot process by pressing the Space bar on the keyboard.
- Enter the following commands at the U-Boot prompt:

```
U-Boot > setenv fdt_file imx8dxl-evk-enet0-avb.dtb
U-Boot > saveenv
U-Boot > boot
```

The change is saved across reboots.

2.2.6.2 Using automotive Ethernet PHY expansion card

To support the AVB endpoint use case with automotive Ethernet, an additional Ethernet PHY expansion card (IMX-RMII-BRPHY) is required. See IMX-RMII-BRPHY TJA1100

Update the boot parameters to use the device tree binary that includes the hardware description relative to the i.MX 8DXL EVK board (Using ENET0 and TJA1100 PHY):

- Power on the board and stop the automatic boot process by pressing the Space bar on the keyboard.
- Enter the following commands at the U-Boot prompt:

```
U-Boot > setenv fdt_file imx8dxl-evk-enet0-tja1100-avb.dtb
U-Boot > saveenv
U-Boot > boot
```

The change is saved across reboots.

2.2.6.3 Using SJA1105Q-EVB as AVB Bridge

The i.MX 8DXL EVK can be connected to the SJA1105Q-EVB board through the ENET0 interface. See Figure 7.

2.3 Configuring GenAVB/TSN stack and demo applications

This section describes steps to configure GenAVB/TSN stack and demo applications.

2.3.1 Profiles supported by GenAVB/TSN stack

For some hardware platforms, the GenAVB/TSN stack supports both modes: endpoint TSN and Endpoint AVB.

To configure the stack to Endpoint AVB mode, use the file `/etc/genavb/config` to set the `GENAVB_TSN_CONFIG` parameter to the right configuration:

```
# avb.sh stop_all
# vi /etc/genavb/config
```

For platforms supporting only Endpoint AVB (such as i.MX 8M Mini and i.MX 6ULL), use the value below:

```
GENAVB_TSN_CONFIG=1
```

For platforms supporting both Endpoint AVB and Endpoint TSN (such as i.MX 8M Plus, i.MX 8DXL, and i.MX 93), use the value below:

```
GENAVB_TSN_CONFIG=2
```

2.3.2 Configuring the GenAVB/TSN stack to start at system boot

By default, the stack does not start automatically on boot. Enable a systemd service to ensure automatic stack startup (on the next reboot) by using the following commands:

```
# systemctl enable genavb-tsn
# systemctl daemon-reload
```

2.3.3 Profiles supported by GenAVB/TSN stack

The following sections describe the various profiles supported by the GenAVB/TSN stack as AVB endpoint. The change from one profile to another is made by modifying the `/etc/genavb/config_avb` file.

This file specifies a pair of configuration files:

- The `APPS_CFG_FILE` (`apps-*.cfg`) points to a file containing a demo configuration (media application to use, controller option, or so on). The startup script `avb.sh` parses this file.
- The `GENAVB_CFG_FILE` (`genavb-*.cfg`) points to a file containing the configuration of the AVB stack. The AVB application parses this file.

A demo profile comprises a pair of `cfg` files.

The file `/etc/genavb/config_avb` already groups the `cfg` files by pairs. Therefore, uncomment the two lines corresponding to the desired demo profile.

3 AVB Milan Audio Sampler/Amplifier with CRF

This section describes AVB Milan Audio Sampler and Amplifier with CRF support, implemented on i.MX evaluation boards, connected through an AVB switch. The setup is shown in Figure 10.

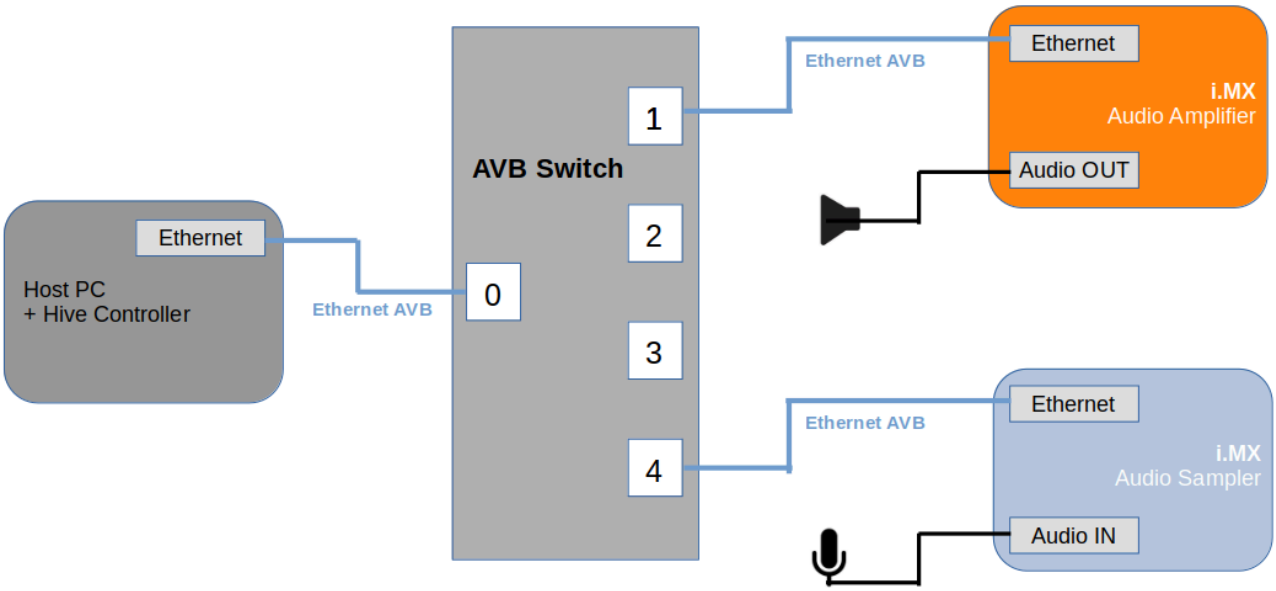

Figure 10. AVB Milan Audio Sampler/Amplifier with CRF setup

3.1 Requirements

The setup requirement is as follows:

- An **i.MX Audio Sampler** capable board **with media clock recovery support**.
- An **i.MX Audio Amplifier** capable board **with media clock recovery support**.
- An AVB switch is required to interconnect the talker and listeners endpoints.
- A host PC with a Hive controller installed.
- Headphones/speakers with male Jack.

3.2 Setup preparation

The hardware preparation of the i.MX platforms is similar to the AVB Audio Sampler/Amplifier setup. All the endpoints must be connected to an AVB switch. The configuration of the AVB stack on talker and listeners is specific to this evaluation setup and must be completed as described in the next sections.

3.2.1 Preparing the AVB configuration for the Talker

The default AVB script must be modified to configure operations of the Talker entity as using a custom Media Application. The AVB Stack is provided with a Media application example, interfaced to the AVB stack through the GenAVB/TSN API, and supports audio sampling from a speaker. To enable using this media application, the `GenAVB/TSN avb` configuration file must be modified as follows:

1. Power ON the i.MX board and let the boot process complete.

2. Edit the `GenAVB/TSN avb` configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to profile 22:

```
PROFILE=22
```

4. Exit and save the file.
5. Reboot the board. The change is saved across reboots, so perform this step only once.

3.2.2 Preparing the AVB configuration for the Listener

The default AVB script must be modified to configure operations of the Listener entity as using a custom Media Application. The AVB Stack is provided with a Media application example, interfaced to the AVB stack through the GenAVB/TSN API, and supports audio playout from a headphone. To enable using this media application, the `GenAVB/TSN avb` configuration file must be modified as follows:

1. Power ON the i.MX board and let the boot process complete.
2. Edit the `GenAVB/TSN avb` configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to profile 22:

```
PROFILE=22
```

4. Exit and save the file.
5. Reboot the board. The change is saved across reboots, so perform this step only once.

3.2.3 Preparing the Hive Controller on Host PC

On the Host PC, you must retrieve Hive Controller's binaries and follow the instructions to compile and install it depending on your platform.

Then, follow the instructions provided in the above URL to execute the Hive Controller.

Figure 11 shows an example of the interface displayed once both endpoints have started with the AVB stack running.

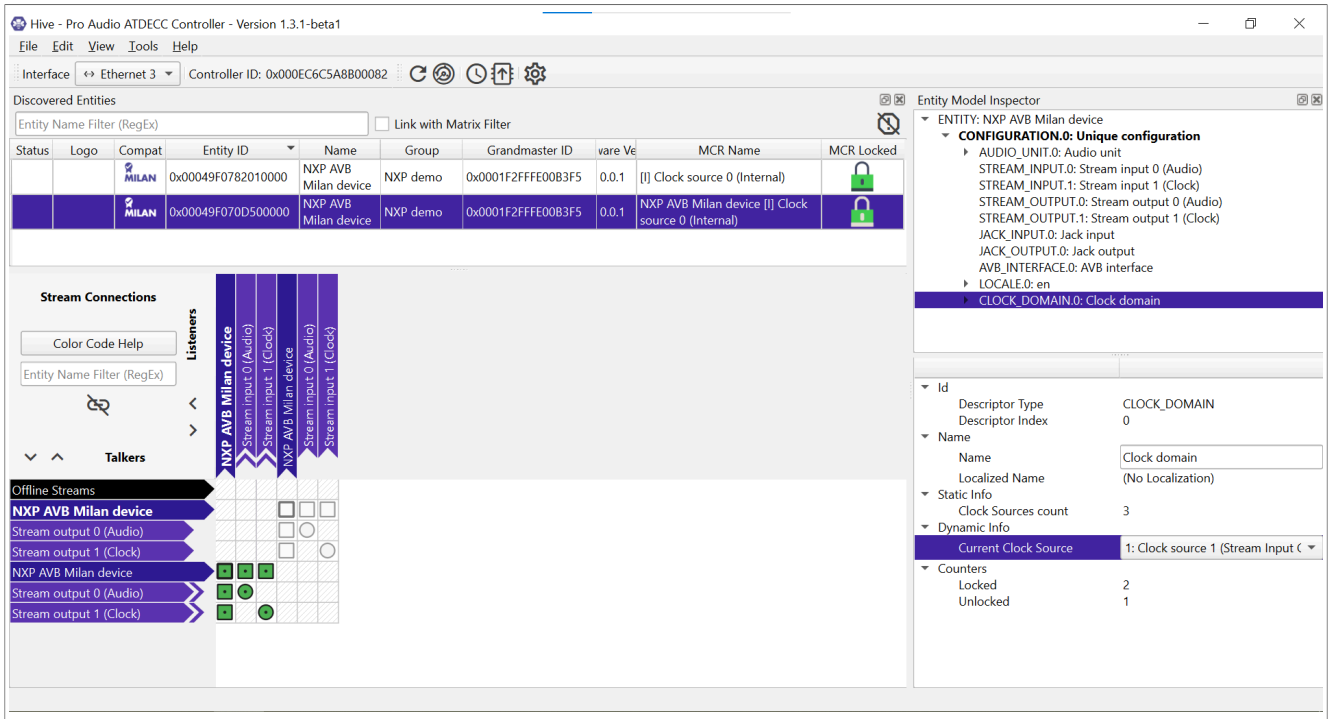

Figure 11. Hive Controller interface with CRF streams

3.3 Evaluation instructions

On both endpoints, power and boot the boards, and log in as root (no password).

The AVB stack starts automatically on each endpoint (per `genavb-tsn` systemd service configuration described in Section 2.3.2).

Once the stack is started on each endpoint, you can start the Hive Controller on the Host PC.

Two Milan compatible entities would be detected and appear on the interface.

3.3.1 Setup media clock domain

Both endpoints use the same AVDECC entity model with a default clock source set to the "Internal clock source".

Use the Hive controller to set the clock domain sources on both endpoints to a valid configuration and connect the needed clock source streams.

For this use case:

- Endpoint acting as Talker for both AAF (audio) and CRF (clock): Keep and verify that the clock source is "Internal clock source". See Section 3.3.1.1.
- Endpoint acting as Listener for both AAF (audio) and CRF (clock): Set the clock source to "Stream Input CRF". See Section 3.3.1.1.

3.3.1.1 Setup clock sources

To change the default clock source, click on the discovered entity, then on its clock domain descriptor and select the clock source to be used. The options for the clock source are: Internal or Stream Input (for both AAF and CRF format). The list of clock sources available in the Entity Model Inspector is shown in Figure 12.

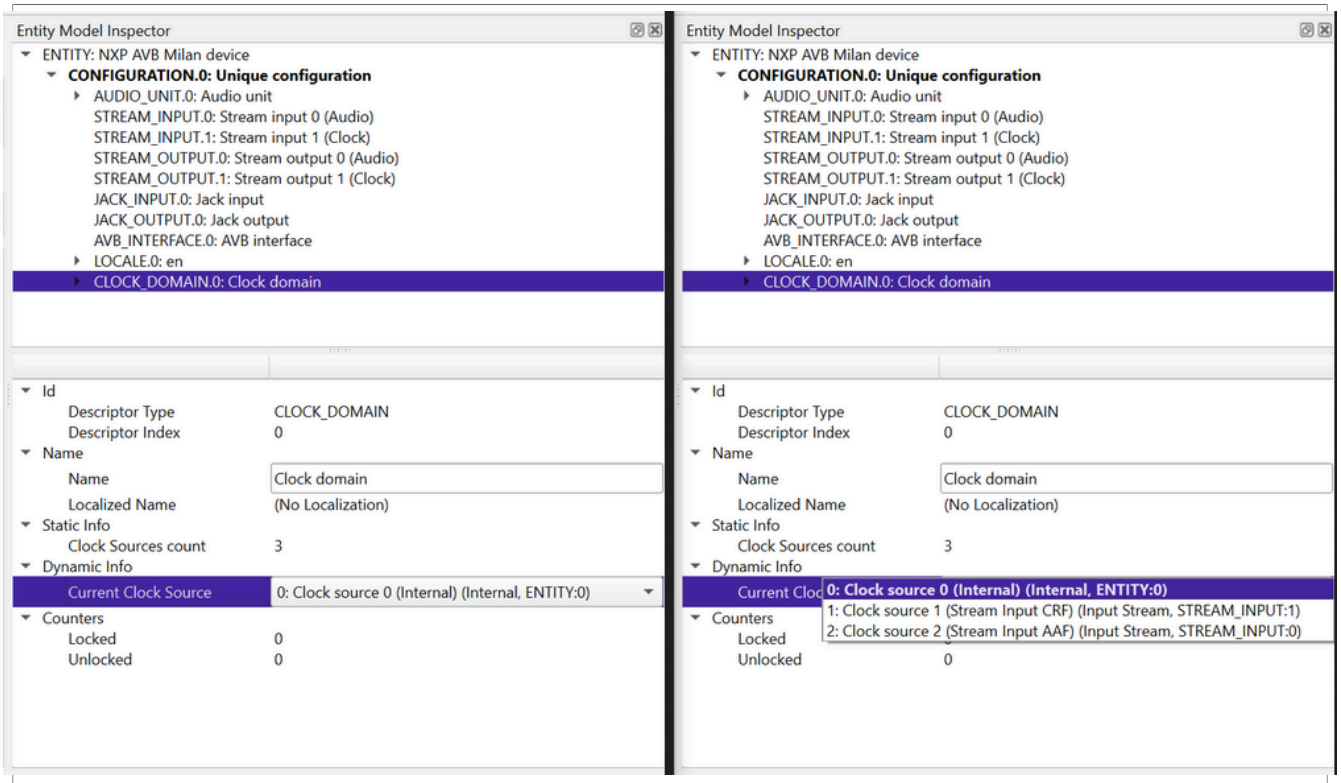

Figure 12. List of clock sources in the clock domain descriptors

For this use case:

- Set the clock source of the endpoint acting as the CRF Listener, to "Clock source 1 (Stream Input CRF)".
- Set the clock source of the endpoint acting as the CRF Talker, to "Clock source 0 (Internal)".

Note:

- *If an endpoint's clock domain uses a stream input for its clock source, the stream input of that endpoint must be active for the media clock to be properly set up.*
- *The endpoint acting as the CRF Listener must support media clock recovery.*

3.3.1.2 Connect clock streams

To connect a stream, click the corresponding box in the matrix of the ‘Stream Based’ section. (See Section 3.2.3). Once the streams are successfully connected, the selected box turns green.

- Connect the CRF stream input ‘Stream input 1 (Clock)’ of the endpoint using its CRF stream input as its clock source (The endpoint must support media clock recovery) and acts as the CRF Listener to the CRF stream output ‘Stream output 1 (Clock)’ of the endpoint that acts as the CRF Talker.

A successful connection is reported with a green dot.

3.3.2 Connect audio streams with the default (stereo) format

The default stream format used in the avdecc entity model is the stereo format (audio with 2 channels)

To connect the default format: no additional change is required on stream inputs/outputs descriptors.

- Connect the Audio stream input ‘Stream input 0 (Audio)’of the endpoint you want to use as a Listener, to the Audio stream output ‘Stream output 0 (Audio)’ of the endpoint you want to use as a Talker.

(See Hive Controller interface with CRF streams of Section 3.2.3)

Once the connection is reported successful (green dot), you can hear on the Listener's headphone, any sound passed into the Talker's audio input port.

3.3.3 Connect audio streams with different channel count

The avdecc entity model supports multiple stream formats with different channel counts (1, 2, 4, 6 and 8 channels) for the stream inputs/outputs. To change the stream inputs/outputs format, use the Hive controller to set the desired stream format on both talker and listener. The list of formats available in the Entity Model Inspector is shown in Figure 13.

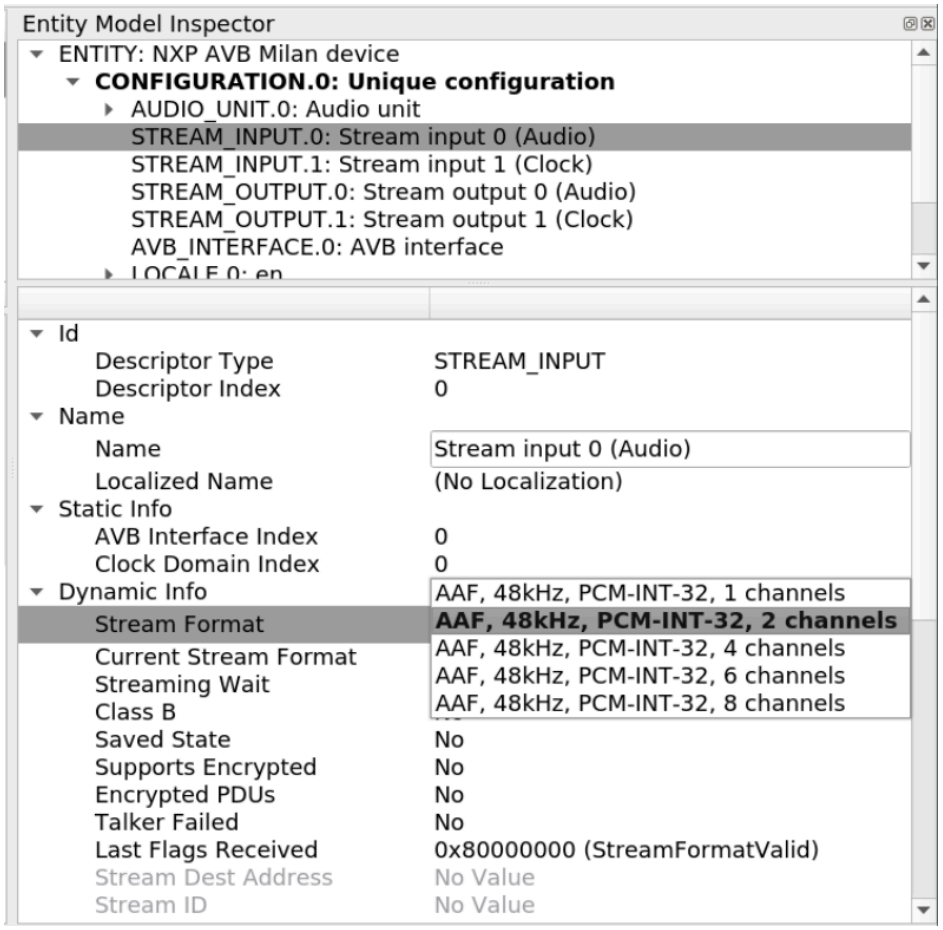

Figure 13. List of formats with different channel counts in the stream descriptors

Then:

- Connect the Audio stream input 'Stream input 0 (Audio)'of the endpoint you want to use as a Listener, to the Audio stream output 'Stream output 0 (Audio)' of the endpoint you want to use as a Talker.

Once the connection is reported successful (green dot), you can hear on the Listener's headphone, any sound passed into the Talker's audio input port.

Note:

- *Stream format changes are not permitted while stream is running.*

4 AVB audio sampler/amplifier back-to-back

This section describes the i.MX platform supporting AVB Audio Talker and Listener roles on two i.MX evaluation boards connected back-to-back as shown in Figure 14.

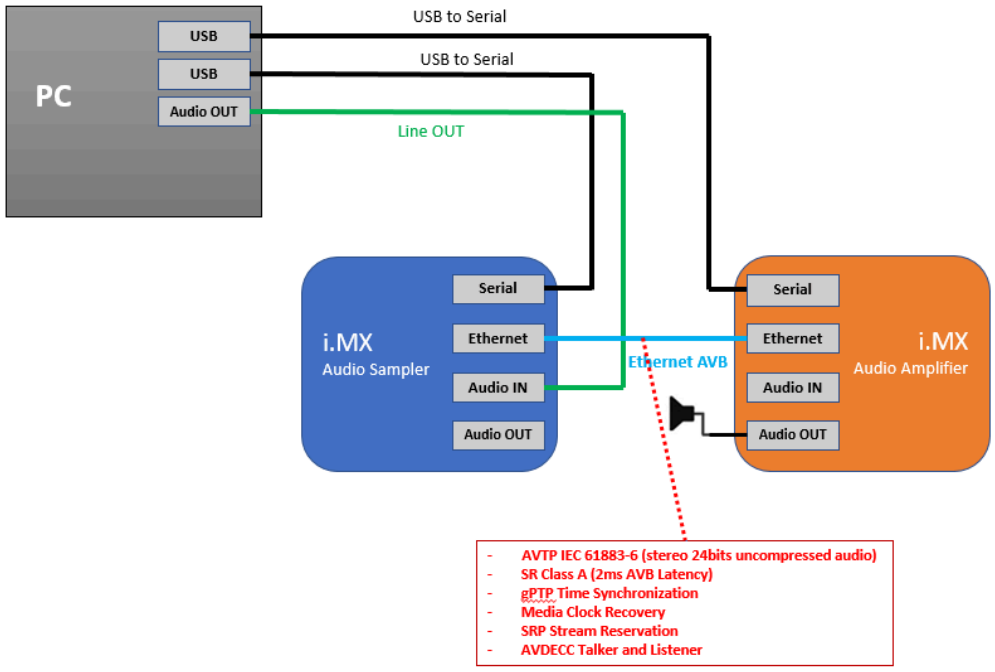

Figure 14. AVB i.MX back-to-back setup

4.1 Requirements

1. One **i.MX Audio Amplifier** capable evaluation board
2. One **i.MX Audio Sampler** capable evaluation board
3. Headphones/speakers with male Jack as shown in Figure 15

Figure 15. Headphones used as speakers

4. Two USB/Serial cables
5. Windows OS laptop (see Figure 16)

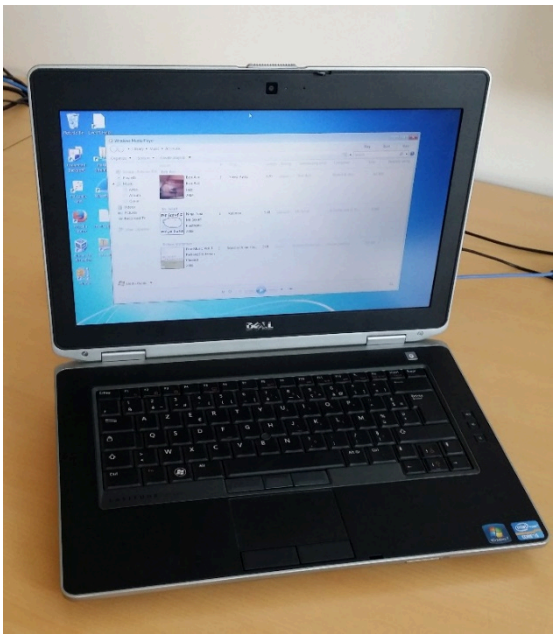

Figure 16. Laptop running Windows OS

4.2 Setup preparation

1. Connect the headphones/speakers to the **i.MX Audio Amplifier** audio Jack port.
2. Connect the Line OUT of the laptop to the available audio input (audio jack port with microphone feature or on board MIC) of the i.MX Audio Sampler.
3. Connect both the i.MX boards with an Ethernet RJ45 cable.
4. Connect a Serial/USB cable to each i.MX board and to some USB ports of the Laptop.
5. Install a Terminal emulator on the Laptop for enabling console display of the i.MX boards through the serial/ USB ports, as shown in the Figure 17.

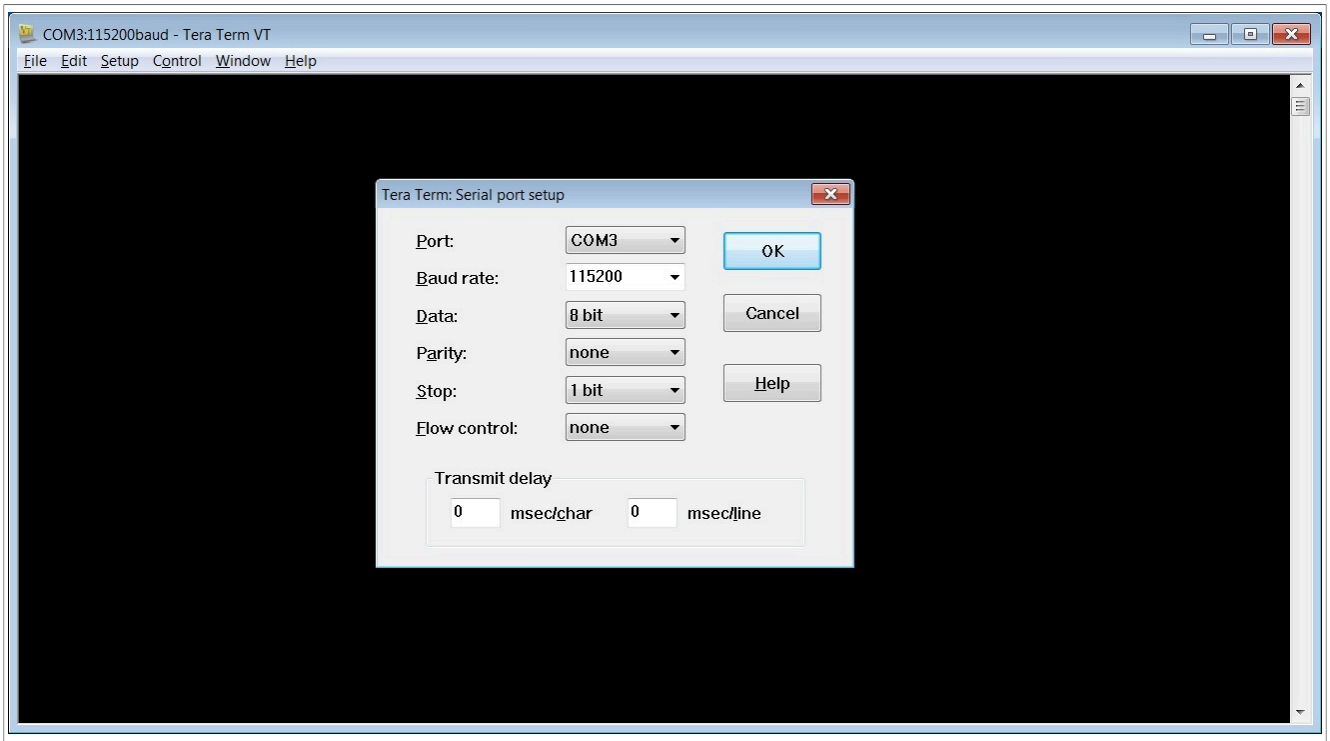

Figure 17. Connecting to the i.MX console via a terminal

Now, configure the i.MX boards for supporting media clock functions. Configure the board acting as talker (sampler) to support media clock generation. Further, configure the board acting as listener (amplifier) to support media clock recovery. Refer to the section Initial Preparation for completing this preparation, depending on the board type used for evaluation.

4.2.1 Preparing the AVB configuration for the Talker

The default AVB script must be modified to configure operations of the Talker entity as using a custom Media Application. The AVB Stack is provided with an ALSA application, supporting audio sampling from an ALSA interface. To enable this media application, the GenAVB/TSN avb configuration file must be modified as follows:

1. Power on the i.MX board and let the boot process complete.
2. Edit the GenAVB/TSN avb configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to PROFILE 14:

```
PROFILE=14
```

4. Exit and save the file. Then, reboot the board. The change is saved across reboots, so perform this step only once.

The setup is then ready for evaluation.

4.2.2 Preparing the AVB configuration for the listener

The default AVB script must be modified to configure operations of the Listener entity as using a custom Media Application. The AVB Stack is provided with an ALSA application example, interfaced to the AVB stack through the GenAVB/TSN API, and supporting audio playout to an ALSA interface. To enable this media application, the GenAVB/TSN avb configuration file must be modified as follows:

1. Power on the i.MX board and let the boot process complete.
2. Edit the GenAVB/TSN avb configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to PROFILE 15:

```
PROFILE=15
```

4. Exit and save the file and then reboot the board.
5. The change is saved across reboots, so this is required to be done only once.

4.3 Evaluation instructions

On both talker and listener boards, power on and boot the boards, and then login as root (no password). The AVB stack is automatically started (per `genavb-tsn` systemd service configuration described in Section 2.3.2).

The AVB audio stream connection is established automatically. When PTP is synchronized between the two systems, a media player on the laptop can be used to play audio to its line out interface. Audio is rendered on the headset of the listener board.

5 AVB audio media server/amplifier Back-to-back

This section describes the i.MX platform supporting AVB Audio Media Server and Amplifier roles on two i.MX boards connected back-to-back. This setup uses custom media applications interfaced to the AVB Stack through the GenAVB/TSN API. See Figure 18.

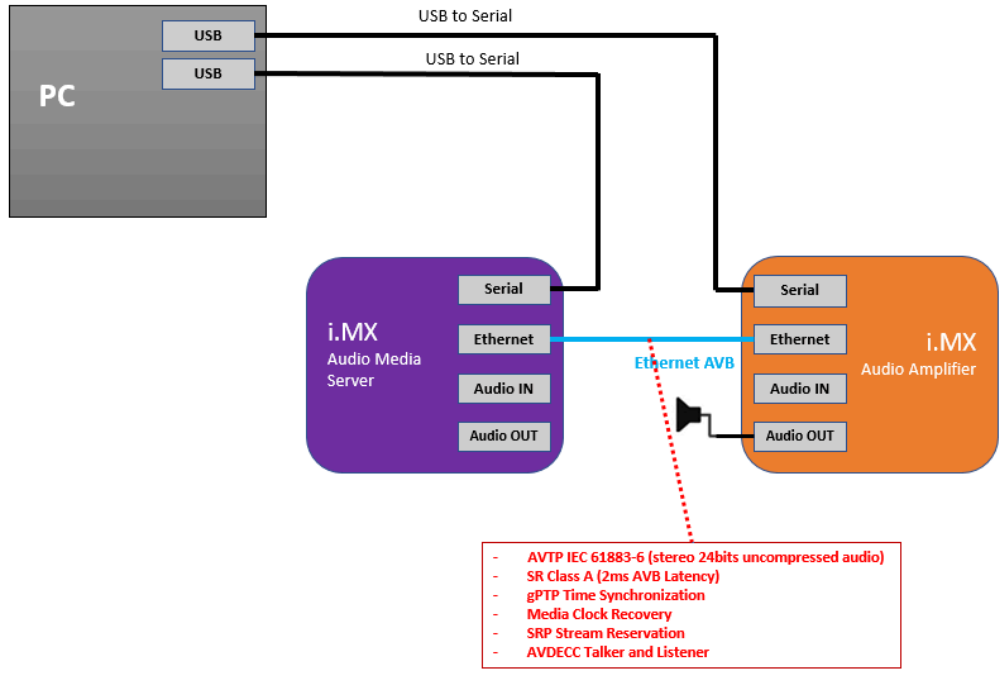

Figure 18. AVB Audio Media Server setup

5.1 Requirements

The section requirement is similar to the AVB Audio Sampler/Amplifier Back-to-Back setup, with the following variations:

- The i.MX talker is a Media Server (**i.MX Audio Media Server** capable board) reading samples from a file stored on the SD flash memory.
- The talker does not need to support media clock functions.

5.2 Setup preparation

The setup preparation is similar to the AVB Audio Sampler/Amplifier Back-to-Back setup. The variation is that there is no need to connect an audio analog device to the audio input of the Talker board, as the samples to be played out are stored in a file of the SD flash memory. For this reason, configuring the talker board for supporting media clock generation is not required for this use case.

5.2.1 Preparing AVB configuration for the Talker

To configure operations of the Talker entity for using a custom Media Application, you must modify the default AVB script. The AVB Stack supports reading audio samples from a media file. To enable this, it is provided with a simple Media Server application example interfaced to the AVB stack through the GenAVB/TSN API. To enable using this media application, modify the GenAVB/TSN avb configuration file as follows:

1. Power ON the i.MX board and let the boot process complete.
2. Edit the GenAVB/TSN avb configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to PROFILE 9:

```
PROFILE=9
```

4. Exit and save the file. Then, reboot the board.

The change is saved across reboots, so perform this step only once. The setup is then ready for evaluation.

5.2.2 Preparing the AVB configuration for the Listener

The default AVB script must be modified to configure operations of the Listener entity as using a custom Media Application. The AVB Stack is provided with an ALSA application example, interfaced to the AVB stack through the GenAVB/TSN API, and supporting audio playout to the ALSA interface. To enable using this media application, the GenAVB/TSN avb configuration file must be modified as follows:

1. Power on the i.MX board and let the boot process complete.
2. Edit the GenAVB/TSN avb configuration file using the following command in the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to PROFILE 11:

```
PROFILE=11
```

4. Exit and save the file then reboot the board.

The change is saved across reboots, so this has only to be done once.

5.3 Evaluation instructions

On both Talker and Listener boards, power on and boot the boards, and log in to the boards as root (no password). The AVB stack automatically starts (as per `genavb-tsn` systemd service configuration described in Section 2.3.2).

The AVB audio stream connection is automatically established between Talker and Listener. At that point, the Media Server application of the Talker starts streaming the audio samples from the source audio file to the AVB Stack. The audio file is read indefinitely in a loop. Samples are transported as an AVB stream to the Listener and delivered to the ALSA application for playout to the audio device. Audio is rendered on the headset or speakers connected to the listener board.

The evaluation can be stopped by using the following command on either the Talker or the Listener side:

```
# systemctl stop genavb-tsn
```

It can be re-started using the following command on either the Talker or the Listener side:

```
# systemctl start genavb-tsn
```

5.4 Audio file format and generation

The current Media Server application uses a raw audio format:

- Two channels
- 48 kHz

- 24 bits
- Big Endian numbering format

An **i.MX Audio Sampler** capable board can be used to generate such a file, by connecting an analog source (external player) to a Jack audio input port.

The following ALSA command allows capturing and storing the file in the expected format:

```
# arecord-D <device> –t raw –c 2 –r 48000 –f S24_BE <file name>
```

To verify and listen to the captured file, connect a hearing device (headphones/speakers) to audio output Jack port or MIC. (In case of the SABRE-AI board, use the RCA/Jack cable connected to Audio Line OUT port.)

Use the following ALSA command:

```
# aplay -D <device> –c 2 –r 48000 –f S24_BE <file name>
```

Assign the ALSA device to the audio codec connected to the jack interface (wm8960-audio).

6 AVB Audio Multi-Stream

This section describes i.MX platform supporting AVB Audio Media Server and Listener roles on three i.MX boards connected through an AVB switch. Figure 19 shows the AVB Audio Multi-Stream setup.

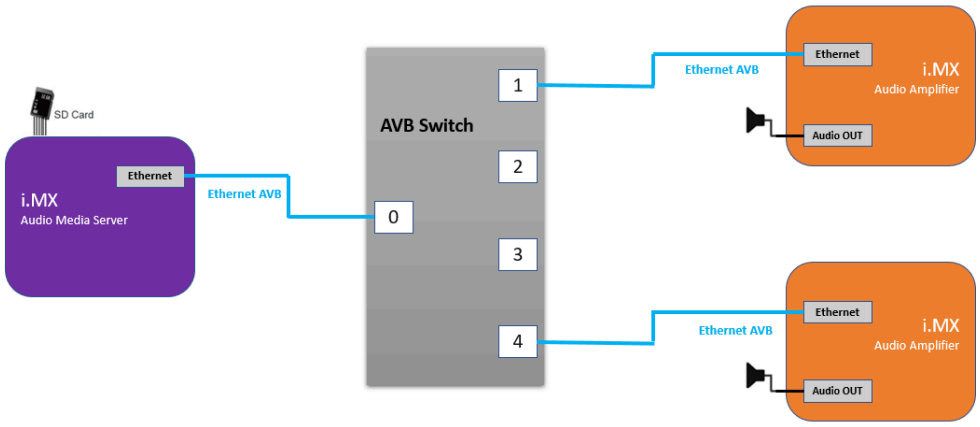

Figure 19. AVB Audio Multi-Stream setup

6.1 Requirements

The setup requirement is as follows:

- The i.MX talker is an Audio Media Server reading samples from some files stored on the SD flash memory. The platform is similar to the talker used in the AVB Audio Media Server setup.
- The i.MX listeners are Audio Amplifiers similar to the listener used in the AVB Audio Sampler/Amplifier Back-to-Back setup. Up to four listeners can be part of the evaluation setup.
- An AVB switch is required to interconnect the talker and listeners nodes.

6.2 Setup preparation

The hardware preparation of the i.MX platforms is similar to the AVB Audio Media Server setup, with the variation that several listeners can be prepared. All such endpoints must be connected to an AVB switch. The configuration of the AVB stack on talker and listeners is specific to this evaluation. Perform the configuration as described in the following sections.

6.2.1 Preparing the AVB configuration for the Talker

The default AVB script must be modified to configure operations of the Talker entity as using a custom Media Application. The AVB Stack is provided with a multi-stream Media Server application example, interfaced to the AVB stack through the GenAVB/TSN API, and supports reading audio samples from media files. The multi-stream application uses an AEM profile supporting streaming of up to 8 audio streams.

To enable this media application, modify the GenAVB/TSN avb configuration file as follows:

1. Power on the i.MX board and let the boot process complete.
2. Edit the GenAVB/TSN avb configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to PROFILE 7:

```
PROFILE=7
```

4. Exit and save the file then reboot the board.
5. A raw audio file `sample1.raw` is available in the `/home/media` repository. The multi-stream application example looks for audio files named `talker_mediaX.raw` in the `/home/media` repository, with `X` being the stream number (i.e. stream #0 will read `media0.raw` file…). Hence, before executing the multi-stream application, some symbolic links must be created in the `/home/media` directory for associating the `talker_mediaX.raw` names to the `sample1.raw` file, or any other raw audio file that would be present in the file system:

```
# ln -s sample1.raw talker_media0.raw
# ln -s sample1.raw talker_media1.raw
# ln -s sample1.raw talker_media2.raw
# ln -s sample1.raw talker_media3.raw
```

6. Check the audio files links:

```
# ls -l
total 10708
lrwxrwxrwx  1 root root 11 Jan  1 00:18 talker_media0.raw -> sample1.raw
lrwxrwxrwx  1 root root 11 Jan  1 00:19 talker_media1.raw -> sample1.raw
lrwxrwxrwx  1 root root 11 Jan  1 00:20 talker_media2.raw -> sample1.raw
lrwxrwxrwx  1 root root 11 Jan  1 00:20 talker_media3.raw -> sample1.raw
-rwxr--r--  1 root root 6000000 Feb  6  2015 sample1.raw
```

Reboot the board. The change is saved across reboots, so perform this step only once.

7. The setup is now ready for evaluation.

6.2.2 Preparing the AVB configuration for the Listeners

This section describes steps for configuring operations of the listener entity to use the ALSA application, supporting audio playout to the ALSA interface. The default AVB script must be modified to enable the media application. To achieve this, modify the GenAVB/TSN configuration file using the steps described below:

1. Power on the i.MX board and let the boot process complete.
2. Edit the `GenAVB/TSN avb` configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to `PROFILE 11`:

```
PROFILE=11
```

4. Then, edit the `/etc/genavb/genavb-listener-btb.cfg` configuration file to set the talker's AVDECC Entity ID and Unique ID information. This step is mandatory to configure the stream output of the talker, the current listener being configured would connect to.
In this use case, each listener connects to a different AVB stream. Therefore, each listener in the setup must connect to a different talker's Unique ID, with audio transported in separate AVB streams. For this, uncomment and edit the following fields in the AVDECC entity 1 section with X being the entity , and Y being the unique ID:

```
[AVB_AVDECC_ENTITY_1]
…
talker_entity_id_list = X
…
talker_unique_id_list = Y
```

Note: *1) The Talker entity ID is the unique EUI-64 identifier of the Talker AVDECC entity. It is derived from the EUI-48 MAC address of the Talker padded with the entity index value. For instance, assuming a Talker has a MAC address set as 00:11:22:33:44:55 and its entity index within the AVDECC configuration is 0x0000, then its corresponding entity ID would be:* `talker_entity_id_list = 0x0011223344550000` *The Talker Unique ID identifies the stream source in the talker's entity model, usually an index starting from 0. For instance, to connect to the stream output 0 the corresponding talker unique ID would be:*

```
talker_unique_id_list = 0
```

Note: *2)The talker entity information can be displayed by using the AVDECC controller application available on the talker endpoint:*

```
root@imx6qsabreauto-avb:~# genavb-controller-app -l
NXP's GenAVB AVDECC controller demo application
Number of discovered entities: 2
Entity ID = 0x49f04433f0001     Model ID = 0x49f04433f0001     MAC address:
 00:04:9F:04:43:3F     Capabilities = 0x8 Association ID = 0x2
Controller

Controls:
        None
Entity ID = 0x49f04433f0000     Model ID = 0x49f04433f0001     MAC address:
 00:04:9F:04:43:3F     Capabilities = 0x708 Association ID = 0x2
     Talker:     sources = 8     capabilities = 0x4801

Stream 0: name = Stream output 0 number of formats = 1 flags = 0x6
 current_format = 0x00a0020240000200 ( 61883-6 AM824 2chans 48000kHz )
Stream 1: name = Stream output 1 number of formats = 1 flags = 0x6
 current_format = 0x00a0020240000200 ( 61883-6 AM824 2chans 48000kHz )
Stream 2: name = Stream output 2 number of formats = 1 flags = 0x6
 current_format = 0x00a0020240000200 ( 61883-6 AM824 2chans 48000kHz )
Stream 3: name = Stream output 3 number of formats = 1 flags = 0x6
 current_format = 0x00a0020240000200 ( 61883-6 AM824 2chans 48000kHz )
Stream 4: name = Stream output 4 number of formats = 1 flags = 0x6
 current_format = 0x00a0020240000200 ( 61883-6 AM824 2chans 48000kHz )
Stream 5: name = Stream output 5 number of formats = 1 flags = 0x6
 current_format = 0x00a0020240000200 ( 61883-6 AM824 2chans 48000kHz )
Stream 6: name = Stream output 6 number of formats = 1 flags = 0x6
 current_format = 0x00a0020240000200 ( 61883-6 AM824 2chans 48000kHz )
Stream 7: name = Stream output 7 number of formats = 1 flags = 0x6
 current_format = 0x00a0020240000200 ( 61883-6 AM824 2chans 48000kHz )
Controls:
        None
```

Enter this command while AVB is running on the endpoints part of the AVB setup.

5. Exit and save the file. Then, reboot the board. Repeat on all the listeners. The changes are saved across reboots, so perform this step only once.

6.3 Evaluation instructions

On both talker and listener boards, power on and boot the boards, and log in to the boards as root (no password). The AVB stack is automatically started (as per `genavb-tsn` systemd service configuration described in Section 2.3.2).

The listeners must connect automatically to the different streams advertised by the talker.

On the talker side, the media application log can be monitored by using the following command:

```
# tail -f /var/log/avb_media_app
```

The log indicates that several streams are configured in parallel with different stream IDs:

```
AVB_MSG_MEDIA_STACK_CONNECT
stream ID: 00049f0254af0003
find_free_thread: thread(0x13bf8) found
find_free_stream: stream(0x13c58) found
mclock_gen_ptp_config: wake period : 72/48000 s = 1500000 ns
Configured AVB batch size (bytes): 1008
stream ID: 00049f0254af0003
media file name: /home/media/talker_media3.raw
mode: TALKER
msg_send(0x13bf8, 1)
msg_receive(0x13bf8, 1)
thread_add_stream: thread(0x13bf8) added stream(0x13c58) fd(7)
AVB_MSG_MEDIA_STACK_CONNECT
stream ID: 00049f0254af0001
find_free_thread: thread(0x13d38) found
find_free_stream: stream(0x13d98) found
Configured AVB batch size (bytes): 1008
stream ID: 00049f0254af0001
media file name: /home/media/talker_media1.raw
mode: TALKER
msg_send(0x13d38, 1)
msg_receive(0x13d38, 1)
thread_add_stream: thread(0x13d38) added stream(0x13d98) fd(10)
```

Each listener plays out music on the regular audio output.

The evaluation can be stopped using the following command on either talker or listeners side:

```
# systemctl stop genavb-tsn
```

It can be restarted using the following command on either talker or listeners side:

```
# systemctl start genavb-tsn
```

6.4 Audio file format and generation

The audio file format and generation is similar to the AVB Audio Media Server setup.

7 AVB Audio Multi-Format

This section describes i.MX platform supporting AVB Audio Multi-format on several i.MX boards connected in back-to-back or through an AVB switch.

7.1 Requirements

The setup requirement is as follows:

- The i.MX talker is an Audio Sampler reading samples from a live input source. The platform is similar to the talker used in the AVB Audio Sampler/Amplifier Back-to-Back setup.
- The i.MX listeners are Audio Amplifiers similar to the listener used in the AVB Audio Sampler/Amplifier Back-to-Back setup. Up to four listeners can be part of the evaluation setup.
- An AVB switch is required to interconnect the talker and listeners nodes, unless there is only one listener.

Note: *The i.MX 6ULL EVK and i.MX 8MP EVK boards do not support 96 kHz (and above) formats.*

7.2 Setup preparation

The hardware preparation of the i.MX platforms is similar to the AVB Audio Sampler/Amplifier Back-to-Back setup, with evolution that several listeners may be prepared. All those endpoints must be connected to an AVB switch. The configuration of the AVB stack on talker and listeners is specific to this evaluation and must be completed as described in the next sections.

7.2.1 Preparing the AVB configuration for the Talker

The default AVB script must be modified to configure operations of the Talker entity as using a custom Media Application. The AVB Stack is provided with a multi-stream Media Server application example, interfaced to the AVB stack through the GenAVB/TSN API, and supporting reading audio samples from media files. The GenAVB/TSN media application uses an AEM profile supporting streaming of 8 different audio streams. Please note that only one of those streams can run at a time as they all have different, non-compatible settings.

To enable this media application, the `GenAVB/TSN avb` configuration file needs to be modified as follows:

1. Power on the i.MX board and let the boot process complete.
2. Edit the `GenAVB/TSN avb` configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to PROFILE 19:

```
PROFILE=19
```

4. Then it is possible to edit the `/etc/genavb/srp.cfg` configuration file to set the talker's enabled SR class. Two classes are needed to run correctly. The configuration file associated to the profile can be modified at any moment, but changes would only be effective after AVB is restarted. SR classes can be enabled by setting "`sr_class_enabled`", which can take any combination of two existing classes, separated by a comma. For example:
   `"A,B"` => enables `SR_CLASS_A` and `SR_CLASS_B`
   `"E,B"` => enables `SR_CLASS_B` and `SR_CLASS_E`
   For any combination, class "HIGH" is the first enabled class in an alphabetical order and class "LOW" the second one. So "A,B" is identical to "B,A".
   It is not allowed to:
   - Enable less/more than 2 classes (for example: "A";"B,C,D")
   - Enable a class twice (for example: "B,B")
   - Enable an unknown class

If those conditions are not followed, an error message is displayed in the logs and the AVB starting process stops.
Note: *Backup the original* `/etc/genavb/srp.cfg` *before changing it so that the original configuration can be restored at the end of the current evaluation.*

The setup is then ready for evaluation.

7.2.2 Preparing the AVB configuration for the Listeners

The default AVB script must be modified to configure operations of the listener entity as using the ALSA application, supporting audio playout to the ALSA interface. To enable this media application, the GenAVB/TSN avb configuration file must be modified as follows:

1. Power on the i.MX board and let the boot process complete.
2. Edit the GenAVB/TSN avb configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to PROFILE 19:

```
PROFILE=19
```

4. Edit the `/etc/genavb/srp.cfg` configuration file to set the listener's enabled SR class. Refer to the talker configuration in the previous section for details.
5. Edit the `/etc/genavb/apps-listener-talker-multi-format.cfg` configuration file to set the listener as media clock receiver as follows:

```
CFG_MEDIA_CLOCK="-L -S 0 -c 0"
```

6. Exit and save the file. Then, reboot the board.
Note: The talker entity information can be displayed out by using the AVDECC controller application available on the talker endpoint:

```
root@imx6qsabreauto-avb:~# genavb-controller-app -l
NXP's GenAVB AVDECC controller demo application
Number of discovered entities: 2
Entity ID = 0x49f039dc00001     Model ID = 0x49f0000080001     MAC address:
 00:04:9F:00:00:07     Capabilities = 0x8 Association ID = 0x0
     Controller
     Controls:
         None
Entity ID = 0x49f039dc00000     Model ID = 0x49f0000070001     MAC address:
 00:04:9F:00:00:07     Capabilities = 0x708 Association ID = 0x0
    Talker:  sources = 8  capabilities = 0x4801
         Stream   0: name =      Stream output 0    number of formats = 1
  flags = 0x6    current_format = 0x00a0020240000200 ( 61883-6 AM824 2chans
 48000Hz)
         Stream   1: name =      Stream output 1    number of formats = 1
  flags = 0x6    current_format = 0x00a0040240000200 ( 61883-6 AM824 2chans
 96000Hz)
         Stream   2: name =      Stream output 2    number of formats = 1
  flags = 0x6    current_format = 0x0205021800806000 ( AAF 2chans 24/32bits
 48000Hz 6samples/packet)
         Stream   3: name =      Stream output 3    number of formats = 1
  flags = 0x6    current_format = 0x020502180080c000 ( AAF 2chans 24/32bits
 48000Hz 12samples/packet)
         Stream   4: name =      Stream output 4    number of formats = 1
  flags = 0x6    current_format = 0x0205021800840000 ( AAF 2chans 24/32bits
 48000Hz 64samples/packet)
```

```
        Stream   5: name =      Stream output 5    number of formats = 1
 flags = 0x6    current_format = 0x0205021800830000 ( AAF 2chans 24/32bits
48000Hz 48samples/packet)
        Stream   6: name =      Stream output 6    number of formats = 1
 flags = 0x6    current_format = 0x020702180080c000 ( AAF 2chans 24/32bits
96000Hz 12samples/packet)
        Stream   7: name =      Stream output 7    number of formats = 1
 flags = 0x6    current_format = 0x0209021800818000 ( AAF 2chans 24/32bits
192000Hz 24samples/packet)
   Listener:   sinks    = 8     capabilities = 0x4801
        Stream   0: name =       Stream input 0    number of formats = 1
 flags = 0x6    current_format = 0x00a0020240000200 ( 61883-6 AM824 2chans
48000Hz)
        Stream   1: name =       Stream input 1    number of formats = 1
 flags = 0x6    current_format = 0x00a0040240000200 ( 61883-6 AM824 2chans
96000Hz)
        Stream   2: name =       Stream input 2    number of formats = 1
 flags = 0x6    current_format = 0x0205021800806000 ( AAF 2chans 24/32bits
48000Hz 6samples/packet)
        Stream   3: name =       Stream input 3    number of formats = 1
 flags = 0x6    current_format = 0x020502180080c000 ( AAF 2chans 24/32bits
48000Hz 12samples/packet)
        Stream   4: name =       Stream input 4    number of formats = 1
 flags = 0x6    current_format = 0x0205021800840000 ( AAF 2chans 24/32bits
48000Hz 64samples/packet)
        Stream   5: name =       Stream input 5    number of formats = 1
 flags = 0x6    current_format = 0x0205021800830000 ( AAF 2chans 24/32bits
48000Hz 48samples/packet)
        Stream   6: name =       Stream input 6    number of formats = 1
 flags = 0x6    current_format = 0x020702180080c000 ( AAF 2chans 24/32bits
96000Hz 12samples/packet)
        Stream   7: name =       Stream input 7    number of formats = 1
 flags = 0x6    current_format = 0x0209021800818000 ( AAF 2chans 24/32bits
192000Hz 24samples/packet)
    Controls:
        Control  0: name =     Volume Control 0    type = 0x90e0f00000000004
   read-only =  No    value_type = 1  min = 0 current = 100 max = 100 step =
1
```

7. This command must be entered while AVB is running on the endpoints part of the AVB setup. Repeat on all the listeners.
   The changes are saved across reboots, so perform the process only once.

7.3 Evaluation instructions

On both talker and listener boards, power on and boot the boards, and log in to the boards as root (no password). The AVB stack is automatically started (as per `genavb-tsn` systemd service configuration described in Section 2.3.2).

To connect streams, use the following command:

```
genavb-controller-app -c <talker_entity_id> <talker_unique_id>
 <listener_entity_id> <listener_unique_id> <flag>
```

`<flag>` allows to choose between the high- and low-class priority.

For example, if *CLASS_A* and *CLASS_C* are enabled, `flag=0` connects a stream using CLASS_A and `flag=1` connects a stream using CLASS_C.

It is important to have **only one stream connected at a time** as all streams are mapped to the same ALSA device (playback or capture) as specified in above configurations.

Also, the talker and listener stream ID must match in order to have the same stream formats connected.

To disconnect a stream, use the command:

```
genavb-controller-app -d < talker_entity_id > <talker_unique_id>
 <listener_entity_id> <listener_unique_id>
```

These AVDECC stream profile definitions are optimized for specific SR classes. Here is a summary of the stream profile classification:

Table 1. AVDECC stream profile definitions and SR classes

Stream number	Rate	Format	Class supported
0	48000 Hz	61883-6 AM824	A,B,C,E
1	96000 Hz	61883-6 AM824	A,B,C,E
2	48000 Hz	AAF	A
3	48000 Hz	AAF	B
4	48000 Hz	AAF	C
5	48000 Hz	AAF	E
6	96000 Hz	AAF	A
7	192000 Hz	AAF	A

The evaluation can be stopped using the following command on either the talker or listener's side:

```
# systemctl stop genavb-tsn
```

It can be restarted using the following command on either the talker or listener's side:

```
# systemctl start genavb-tsn
```

Note that after any change in the enabled SR class configuration, you must manually restart the TSN process alongside the AVB process using the command below:

```
# tsn.sh restart
```

7.4 Audio file format and generation

The audio file format and generation is similar to the AVB Audio Media Server setup.

8 AVB Audio/Video Media Server

This section describes AVB Audio/Video Media Server and Players, implemented on i.MX evaluation boards, connected through an AVB switch as shown in Figure 20.

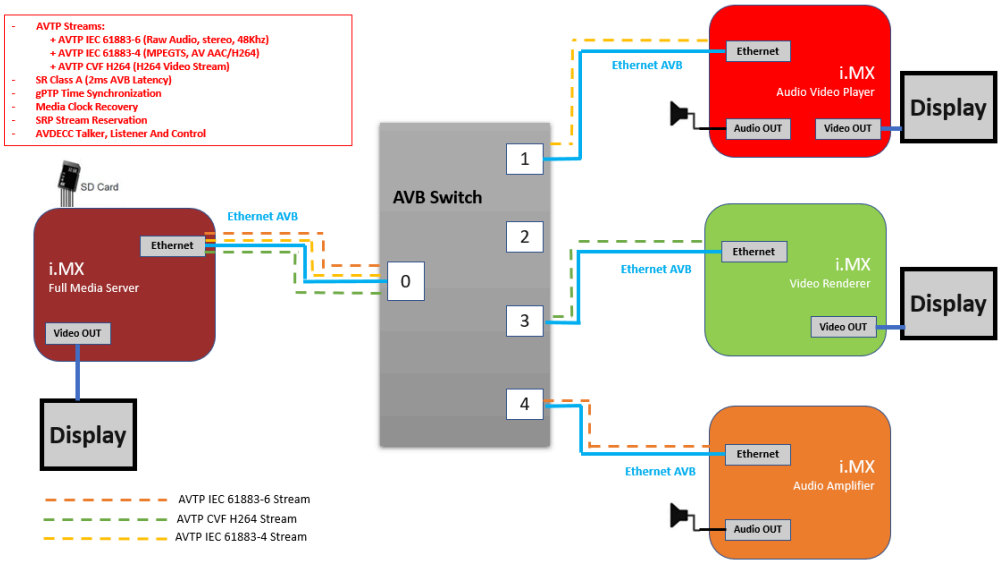

Figure 20. AVB Audio/Video Media Server setup

8.1 Requirements

The setup requirement is as follows:

- The i.MX talker is an **i.MX Full Media Server** capable board reading samples from files stored on the SD flash memory.
- An **i.MX Audio Video Player** capable board. Note that a listener can be configured to play only audio or only video. In this latter case (no audio playout), an i.MX Video Renderer can be used.

Up to 10 listeners can be part of the evaluation setup.

- An **i.MX Video Renderer** capable board can be implemented as listener. The AVB configuration is described in the following section.
- An **i.MX Audio Amplifier** capable board can be implemented as listener. The AVB configuration is described in the following section.
- An AVB switch is required to interconnect the talker and listeners endpoints.

8.2 Setup preparation

The hardware preparation of the i.MX platforms is similar to the AVB Audio Media Server setup, with the variation that several listeners may be prepared, with or without screen display. For the video endpoints the HDMI port can be used for connecting a display.

All the endpoints must be connected to an AVB switch. The configuration of the AVB stack on talker and listeners is specific to this evaluation setup, and must be completed as described in the next sections.

8.2.1 Preparing the AVB configuration for the Talker

The default AVB script must be modified to configure operations of the Talker entity as using a custom Media Application. A media server application example is provided with the AVB Stack, which interfaces the AVB stack through the GenAVB/TSN API. This example supports the reception of A/V samples from GStreamer by reading from media files. To enable this media application, the GenAVB/TSN avb configuration file must be modified as follows:

1. Power on the i.MX board and let the boot process complete.
2. Edit the `GenAVB/TSN avb` configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to PROFILE 4:

```
PROFILE=4
```

4. Make sure that the `GenAVB/TSN systemd` service is enabled to start at boot:

```
# systemctl enable genavb-tsn
# systemctl daemon-reload
```

Enable the system clock to be gPTP-based:

```
CFG_USE_PHC2SYS=1
```

Exit and save the file.

5. The A/V file to be streamed must be stored in the `/home/media` repository. Some A/V files are available in the /home/media repository. They are named as follows:

```
# ls /home/media
sintel_trailer_720p_TOVERLAY.mp4
```

The full path to the A/V file, or at least a directory containing a list of mp4 files, must be mentioned as input parameter to the Media Application. For this purpose, the GenAVB/TSN profile must be modified as follows:

6. Edit the `GenAVB/TSN` video talker profile using the following command at the Linux prompt:

```
# vi /etc/genavb/apps-talker-video.cfg
```

The media application to be used is specified as follows:

```
CFG_EXTERNAL_MEDIA_APP="genavb-media-app"
```

The media application option string contains the below application options:

```
CFG_EXTERNAL_MEDIA_APP_OPT='${CFG_MEDIA_CLOCK} -T -A 0 -m 0:1 -T -A 1 -m 0:0
 -T -A 2 -m 0:2 -f /home/media/'
```

The option string indicates that the media application is configured for multitalker streams pipeline instance **0** (-m **0**:x) and each AVDECC stream index (-A <X>) is mapped to the right pipeline sink (-m 0:**y**). The list of all the multitalker streams pipeline instances can be shown by visualizing the application help (`$genavb-media-app -h`). Each talker stream configuration is separated and preceded by the `-T` option.

7. To modify or extend the media application option string, use the following arguments preferably at the end of the string (not be overridden):
   - '`-f`' indicates the path to the A/V file to be read or the path to an `mp4` media files directory
   - '`-l`' enables video local preview (on the talker)
   - '`-d`' indicates the display type to be used (`lvds` or `hdmi`)
   - '`-p`' <delay> sets the latency (in ns) added to the local preview rendering
8. Exit and save the file. Reboot the board.
   The change is saved across reboots so this has only to be done once.

Note: *While configuring the Talker with local preview rendering, a screen must be connected to the board. Otherwise, the GStreamer pipeline does not perform video looping.*

8.2.2 Preparing the AVB configuration for the Listeners

The default AVB script must be modified to configure operations of the Listener entity as using a custom Media Application. A video application is provided with the AVB Stack, which interfaces with the AVB stack through the GenAVB/TSN API, and supports A/V playout to GStreamer. To enable this media application, the GenAVB/TSN AVB configuration file must be modified as follows:

1. Power on the i.MX board and let the boot process complete.
2. Edit the GenAVB/TSN avb configuration file using the following command at the Linux prompt:
   ```
   # vi /etc/genavb/config_avb
   ```
3. For an Audio/Video listener, set the configuration profile to PROFILE 5:
   ```
   PROFILE=5
   ```
4. For an Audio only listener, set the configuration profile to PROFILE 6:
   ```
   PROFILE=6
   ```
5. For a Video only listener, set the configuration profile to PROFILE 16:
   ```
   PROFILE=16
   ```
6. Make sure that the `GenAVB/TSN systemd` service is enabled to start at boot:
   ```
   # systemctl enable genavb-tsn
   # systemctl daemon-reload
   ```
7. Enable the system clock to be gPTP-based:
   ```
   CFG_USE_PHC2SYS=1
   ```
8. Exit and save the file. The GStreamer options are controlled through the parameters passed to the Media Application.
   To change those parameters, modify the `GenAVB/TSN` profile as described in the next step.
9. For an Audio/Video listener, edit the video listener profile by using the following command at the Linux prompt:
   ```
   # vi /etc/genavb/apps-listener-video.cfg
   ```
   - The Media application to be used is specified as follows:
     ```
     CFG_EXTERNAL_MEDIA_APP="genavb-media-app"
     ```
   - The Media Application option string contains GStreamer options:
     ```
     CFG_EXTERNAL_MEDIA_APP_OPT='${CFG_MEDIA_CLOCK} -L -A 0 -g 0 -p 305000000
      -a -v -d ${CFG_PRIMARY_VIDEO_DEVICE} -P ${CFG_ALSA_PLAYBACK_DEVICE}
     -L -A 1 -g 1 -p 305000000 -v -d ${CFG_PRIMARY_VIDEO_DEVICE}'
     ```
     The *options* string indicates that the media application is configured for two AVDECC (-A <X>) streams. Each of the streams goes to a single GStreamer pipeline (-g <Y>) on connection time.
     - Stream index `0` is configured for an audio video stream that displays on the configured primary video device (HDMI or LVDS).
     - Stream index `1` is configured for a video stream that displays on the configured primary video device (HDMI or LVDS).

Only the first connected AVDECC stream is displayed.

- To modify or extend the Media Application option string, use the following arguments for each stream:
  - '-a' indicates to playout audio only from the IEC_61883_CIP_FMT_4 Stream.
  - '-v' indicates to playout video only from the IEC_61883_CIP_FMT_4 Stream.
  - '-d' indicates the display type to be used (lvds or hdmi).
  - '-p' <delay> sets the decoding delay (ns) for the local rendering.

10. The AVB options are specified in the second `cfg` file:

```
# vi /etc/genavb/genavb-listener-video-btb.cfg
```

The `talker_unique_id_list` indicates the AVDECC `talker_unique_ID` stream source. Its value is:

- `0` for the RAW audio stream
- `1` for the MPEG2-TS A/V stream
- `2` for the CVF H264 Video stream

The `listener_unique_id_list` indicates the AVDECC `listener_unique_ID` stream sink (0 for the first stream).

11. For an Audio only listener, edit the audio listener profile by using the following command at the Linux prompt:

```
# vi /etc/genavb/apps-listener-audio.cfg
```

The Media Application to be used is specified as follows:

```
CFG_EXTERNAL_MEDIA_APP="genavb-media-app"
```

The Media Application option string contains GStreamer options:

```
CFG_EXTERNAL_MEDIA_APP_OPT='${CFG_MEDIA_CLOCK} -L -A 0 -g 0 -p 305000000 -P
 ${CFG_ALSA_PLAYBACK_DEVICE}'
```

The options string indicates that the media application is configured for one AVDECC (-A 0) stream with index 0 that plays through a single GStreamer pipeline.

- '-p' <delay> sets the decoding delay (ns) for the local rendering.
- The AVB options are specified inside the second cfg file:

```
# vi /etc/genavb/genavb-listener-btb.cfg
```

- The '`talker_unique_id_list`' indicates the AVDECC `talker_unique_ID` stream sources:
  - `0` for the RAW audio stream.
  - `1` for the MPEG2-TS A/V stream.
  - `2` for the CVF H264 Video stream.
- '`listener_unique_id_list`' indicates the AVDECC `listener_unique_ID` stream sink (`0` for the first stream).
- For a Video only listener, edit the video listener profile by using the following command at the Linux prompt:

```
# vi /etc/genavb/apps-listener-video.cfg
It uses the same configuration as for the Audio/Video Listener, except that
 the second pipeline will be launched.
```

The AVB options specified inside the second cfg file are:

```
# vi /etc/genavb/genavb-listener-video-h264-btb.cfg
```

- '`talker_unique_id_list`' indicates the AVDECC `talker_unique_ID` stream source:
  - '`0`' for the RAW audio stream.
  - '`1`' for the MPEG2-TS A/V stream.
  - '`2`' for the CVF H264 Video stream.

- 'listener_unique_id_list' indicates the AVDECC listener_unique_ID stream sink (0 for the first stream).

12. Exit and save the file, and then reboot the board. The change is saved across reboots so it must be done only once.
    The setup is then ready for evaluation.

8.3 Evaluation instructions

On both talker and listener boards, power and boot the boards, and log in to the boards as root (no password).

The AVB stack starts automatically on each endpoint (as per the genavb-tsn systemd service configuration described in Section 2.3.2).

The endpoints configuration is using the "*Fast Connect"* option, so the listeners automatically connect their input stream to the talker output streams once detected.

The Media Server runs a video talker application using GStreamer to read an A/V MP4 file from the SD flash storage encoded in H.264/AAC. The A/V source file is selected according to the filename given as an input parameter to the talker application. (Refer to the previous section for preparation of the talker configuration). The A/V file is read indefinitely in a loop.

From this audio/video muxed source, GStreamer pipelines are configured to generate an MPEG2-TS A/V stream, a RAW audio stream, and another CVF H264 Stream. When a stream is connected, the talker application starts reading the source media file and streaming samples to the AVB stack for transport to listeners. If any or all streams are connected, the corresponding samples are passed to the AVB stack. Therefore, the talker can generate three AVB class A streams:

- a RAW audio 2 channel stereo 48 kHz in an IEC 61883-6 AVTP format.
- an MPEG2-TS audio/video AAC/H.264 in an IEC 61883-4 AVTP format.
- a H264 video stream in a CVF H264 AVTP format.

Several listeners can listen to the same stream. Those streams are multicast to the connected listeners that are running the GStreamer application for A/V decoding and playout to the A/V devices. A/V is rendered on the display and/or speakers connected to the listener board, depending on the parameters passed to the listener application. Refer to Figure 21 and Section 8.2 for details about preparation of the listener configuration.

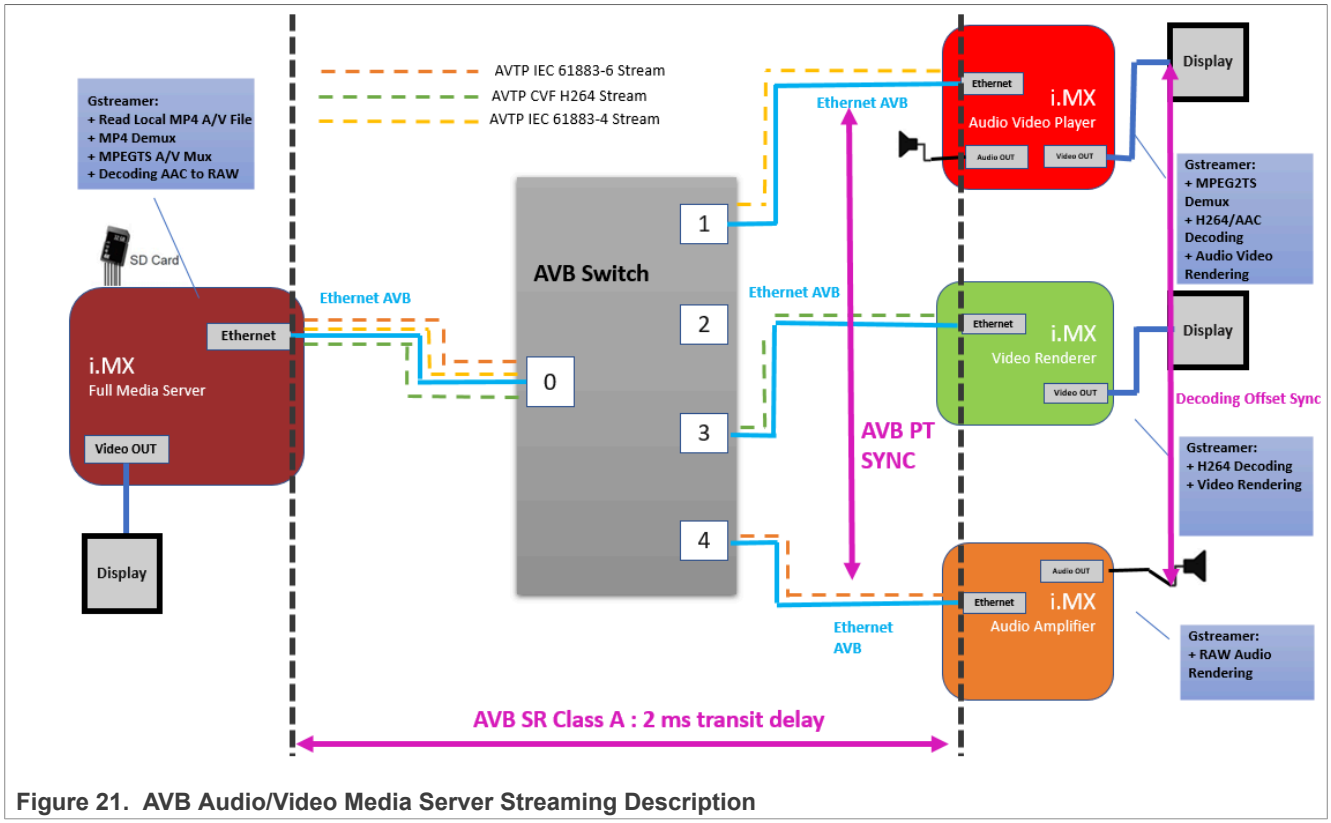

Figure 21. AVB Audio/Video Media Server Streaming Description

The evaluation can be stopped using the following command on either Talker or Listeners side:

```
# systemctl stop genavb-tsn
```

It can be restarted using the following command:

```
# systemctl start genavb-tsn
```

8.4 Audio/Video file format

The A/V file format is expected to be AAC/H.264 encoded, in an MP4 file container. The audio original format must be two channels with 48 KHz sampling rate.

9 AVB Media Synchronization Use Case

This section describes a use case for measuring media synchronization performance. It relies on using an AVB Audio live source stream from a talker transmitted to several listeners and running on three i.MX/Linux evaluation boards connected through an AVB switch.

9.1 Requirements

The setup requirement is as follows:

- The i.MX talker is reading samples from a live source (i.MX Audio Sampler role). The platform is similar to the talker used in the AVB Audio Sampler/Amplifier Back-to-back setup.
- The i.MX listeners are Audio Amplifiers similar to the listener used in the AVB Audio Sampler/Amplifier Back-to-Back setup. Up to four listeners can be part of the evaluation setup. (i.MX Audio Amplifiers role)
- An AVB switch is required to interconnect the talker and listeners nodes.

9.2 Setup preparation

- The hardware preparation of the i.MX evaluation platforms is similar to the AVB Audio Sampler/Amplifier Back-to-Back setup, with the variation that several listeners may be prepared. All those endpoints must be connected to an AVB switch. The configuration of the AVB stack on talker and listeners is specific to this evaluation and must be completed as described in the next sections.

9.2.1 Preparing the AVB configuration for the Talker

The default AVB script must be modified to configure operations of the Talker entity as using a custom Media Application. The AVB Stack is provided with an ALSA application, supporting audio sampling from an ALSA interface. To enable this media application, modify the GenAVB/TSN avb configuration file as follows:

1. Power on the i.MX evaluation board and let the boot process complete.
2. Edit the GenAVB/TSN avb configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to PROFILE 14:

```
PROFILE=14
```

4. Exit and save the file, then reboot the board.
5. The change is saved across reboots, so this process is required to be done only once.

The setup is then ready for evaluation.

9.2.2 Preparing the AVB configuration for the Listeners

The default AVB script must be modified to configure operations of Listeners entity as using a custom Media Application. The AVB Stack is provided with an ALSA application example, interfaced to the AVB stack through the GenAVB/TSN API, and supporting audio playout to an ALSA interface. To enable this media application, the GenAVB/TSN avb configuration file must be modified as follows:

1. Power on the i.MX evaluation boards and let the boot process complete.
2. Edit the GenAVB/TSN avb configuration file on each board using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to PROFILE 15:

```
PROFILE=15
```

4. Edit `/etc/genavb/genavb-audio-multi-btb-aaf.cfg` to set:

```
fast_connect = 0
btb_demo_mode = 0
```

5. Exit and save the file, then reboot the board. The change is saved across reboots, so this has only to be done once per board.

9.3 Evaluation instructions

On both talker and listeners boards, power on and boot the boards, and then log in it as root (no password). The AVB stack is automatically started on each endpoint (per `genavb-tsn` systemd service configuration described in Section 2.3.2).

To connect streams, use the command:

```
genavb-controller-app -c <talker_entity_id> 0 <listener_entity_id> 0 0
```

To disconnect streams, use the command:

```
genavb-controller-app -d <talker_entity_id> 0 <listener_entity_id> 0
```

The AVB audio stream connection is NOT automatically established. It is the controller's role to establish connections and request disconnections (for example, a talker's reboot does not automatically disconnect listeners from their stream). When PTP is synchronized (see section 10.5) between the two systems, a media player on the laptop can be used to play audio to its line out interface. Audio is rendered on the headset of all listener boards.

9.4 Measuring performance

To perform measurement between two endpoints, use a PC with a stereo LINE-IN and an audio editing software. The software enables capturing audio on two different channels and measuring the time or sample the difference between them, as described below:

- For synchronization measurements, one channel must come from each listener.
- For end-to-end latency measurement, one channel must come from the live audio source feeding the talker and the other must come from the listener.

Multiple audio adapters and cables are required for measurements on the evaluation boards. Figure 22 and Figure 23 show example setups with boards having audio jack input or output.

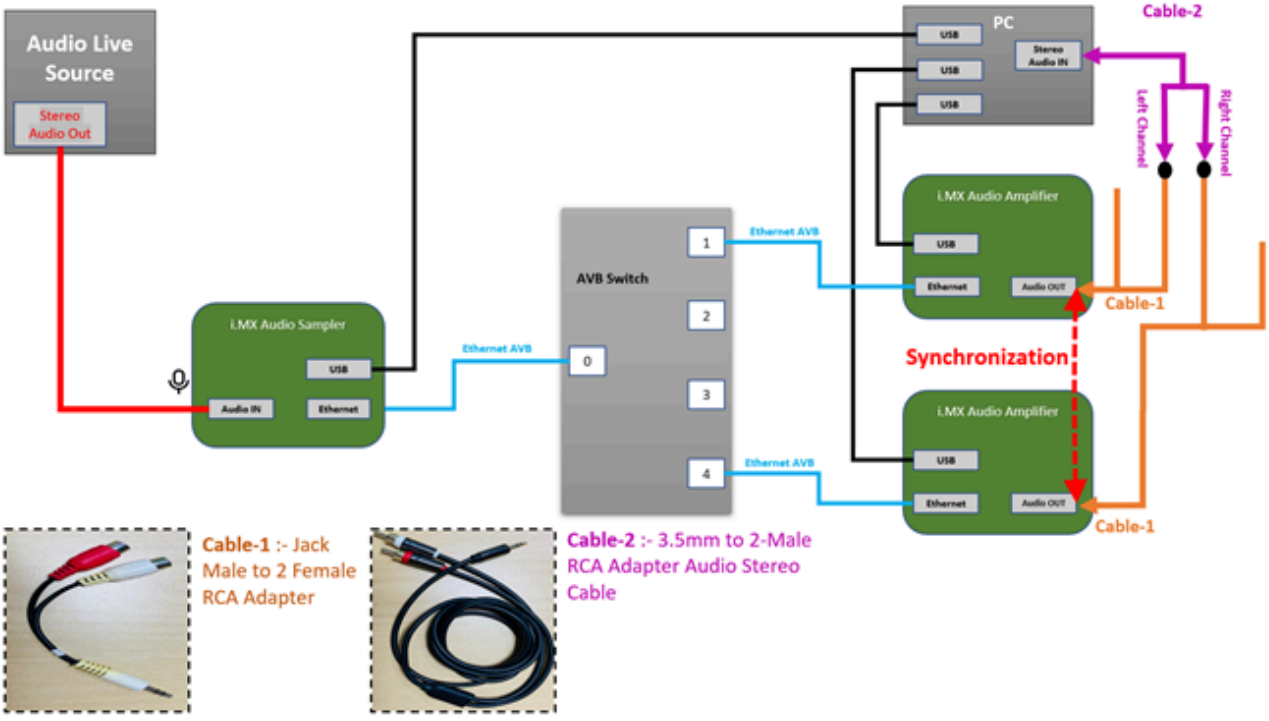

Figure 22. AVB Media Synchronization setup: Synchronization measurement

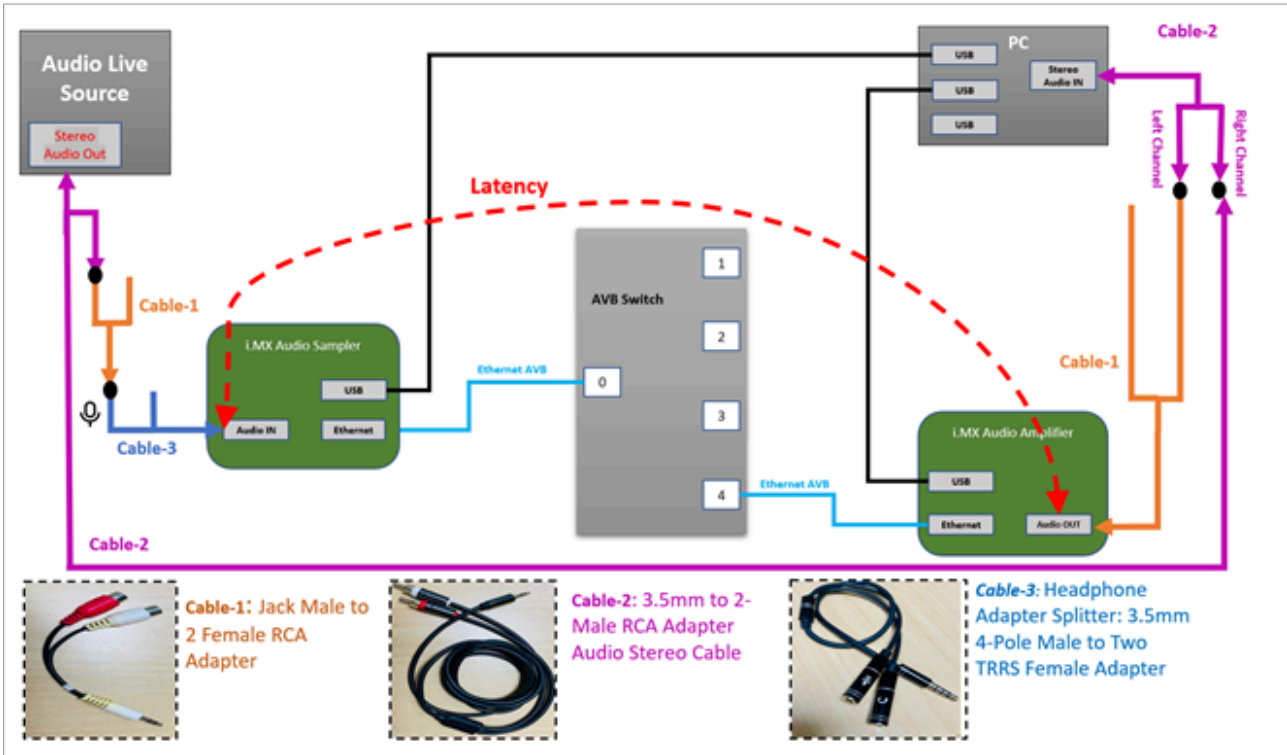

Figure 23. AVB Media Synchronization setup: Latency Measurement

The granularity of the measures is 20 µs for 1 sample. The synchronization performances must be below 50 µs between two listener boards (measured stable at 40 µs). The latency between the talker and one listener (end-to-end latency including application buffering) must be around 8.2 ms.

10 AVB Milan Audio Media Server/Amplifier

This section describes AVB Milan Audio Media Server and Amplifier implemented on i.MX evaluation boards. The boards are connected through an AVB switch as shown in Figure 24.

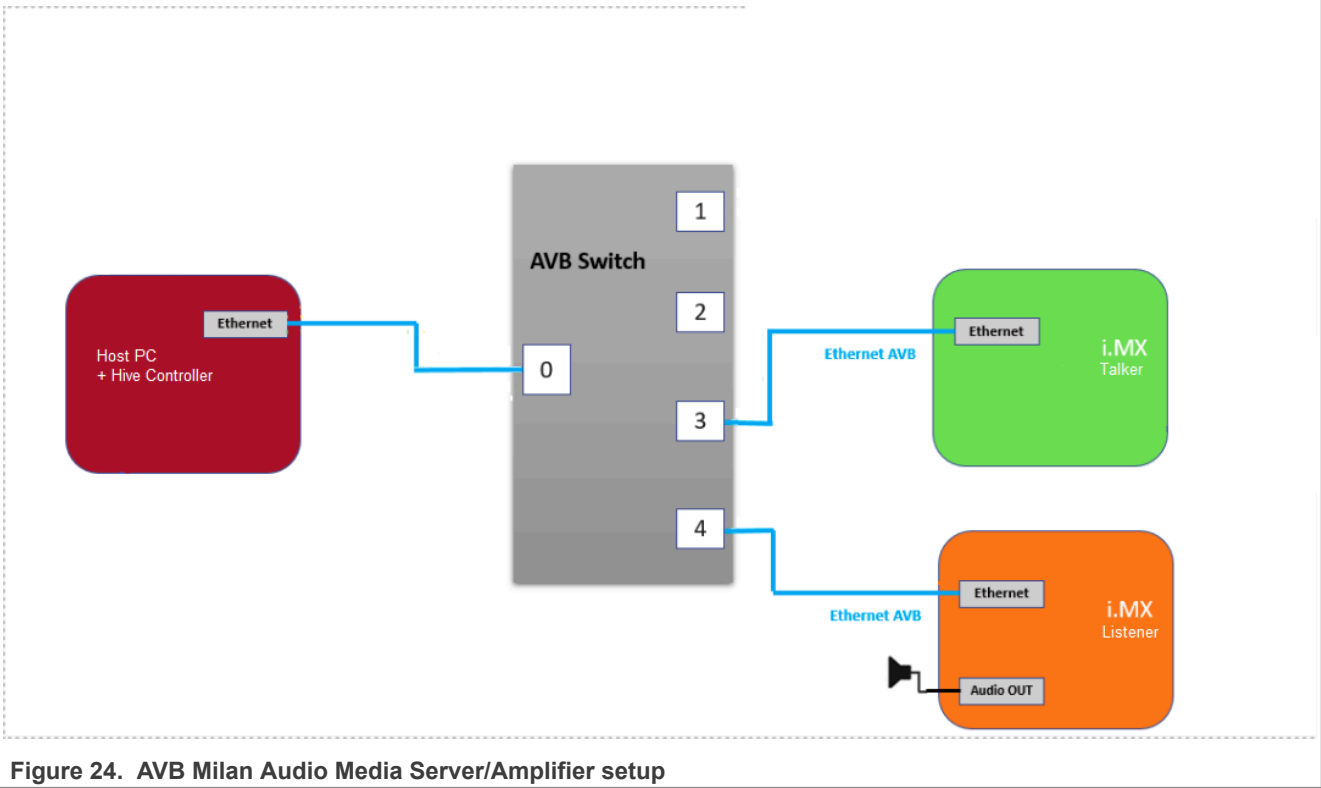

Figure 24. AVB Milan Audio Media Server/Amplifier setup

10.1 Requirements

The setup requirement is as follows:

- An **i.MX Full Media Server** capable board reading samples from files stored on the SD flash memory.
- An **i.MX Audio Amplifier** capable board can be implemented as listener. The AVB configuration is described in the following section.
- An AVB switch is required to interconnect the talker and listeners endpoints.
- A host PC with a Hive controller installed.

10.2 Setup preparation

The hardware preparation of the i.MX platforms is similar to the AVB Audio Media Server setup.

All the endpoints must be connected to an AVB switch. Configuration of the AVB stack on talker and listeners is specific to this evaluation setup and must be completed as described in the next sections.

10.2.1 Preparing the AVB configuration for the Talker

The default AVB script must be modified to configure operations of the Talker entity to use a custom Media Application. The AVB Stack is provided with a Media Server application example, which is interfaced to the AVB stack through the GenAVB/TSN API. The application supports reading audio samples from a media file. To enable using this media application, the GenAVB/TSN configuration file must be modified as follows:

1. Power on the i.MX board and let the boot process complete.
2. Edit the `GenAVB/TSN avb` configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. Set the configuration profile to `PROFILE 20`:

```
PROFILE=20
```

4. Exit and save the file. Make sure that the GenAVB/TSN systemd service is enabled to start at boot:

```
# systemctl enable genavb-tsn
# systemctl daemon-reload
```

The raw audio file to be streamed must be stored in the `/home/media` repository. A sample Audio file is already available in the `/home/media/` directory. It is named as follows:

```
# ls /home/media
sample1_for_aaf.raw
```

The full path to the raw audio file, must be mentioned as input parameter to the Media Application. For this purpose, the `GenAVB/TSN` profile must be modified as follows:

5. Edit the GenAVB/TSN video talker profile using the following command at the Linux prompt:

```
# vi /etc/genavb/apps-talker-simple-aaf.cfg
```

6. The Media Application to be used is specified as follows:

```
CFG_EXTERNAL_MEDIA_APP="simple-audio-app"
```

- The Media Application option string contains application options:

```
CFG_EXTERNAL_MEDIA_APP_OPT="-f /home/media/sample1_for_aaf.raw "
```

- '-f' indicates the path to the raw Audio file (must be in the format: `Raw Audio, S32BE, 2 channels, 48 KHz`).

7. For more information on the options, please refer to the application help (`$ simple-audio-app -h`).
8. Exit and save the file.
9. Reboot the board. The change is saved across reboots so this has to be done only once.

10.2.2 Preparing the AVB configuration for the Listener

The default AVB script must be modified to configure operations of the Listener entity as using a custom Media Application. The AVB Stack is provided with a video application, interfaced to the AVB stack through the GenAVB/TSN API, and supporting audio playout to an ALSA interface. Details of the GenAVB/TSN avb configuration file are as follows:

1. Power on the i.MX board and let the boot process complete.
2. Edit the `GenAVB/TSN avb` configuration file using the following command at the Linux prompt:

```
# vi /etc/genavb/config_avb
```

3. For an Audio/Video listener, set the configuration profile to PROFILE 21:

```
PROFILE=21
```

4. Exit and save the file.
5. Make sure that the `GenAVB/TSN systemd` service is enabled to start at boot:

```
# systemctl enable genavb-tsn
```

```
# systemctl daemon-reload
```

6. The alsa options are controlled through the parameters passed to the Media Application. To change those parameters, the GenAVB/TSN profile must be modified as follows:
   - For an Audio/Video listener, edit the video listener profile by using the following command at the Linux prompt:

     ```
     # vi /etc/genavb/apps-listener-alsa-milan.cfg
     ```

   - The Media application to be used is specified as follows:

     ```
     CFG_EXTERNAL_MEDIA_APP="alsa-audio-app"
     ```

   - The Media application option string contains the following:

     ```
     CFG_EXTERNAL_MEDIA_APP_OPT='-d ${CFG_ALSA_PLAYBACK_DEVICE} -b /etc/genavb/
     milan_binding_params.nvram'
     ```

   - '-b' indicates the file location where binding parameters must be saved in non-volatile memory (as per Milan specification), so that streaming can be recovered without controller intervention in case of a power cycle or network disruption or a similar situation.
7. Exit and save the file, and then reboot the board. The change is saved across reboots so it must be done only once. The setup is then ready for evaluation.

10.2.3 Preparing the Hive controller on the Host PC

On the Host PC, you must retrieve Hive Controller's binaries. Preferably use a Host PC under Windows/MacOS. Otherwise, you must compile the Hive controller yourself.

Follow the instructions provided in the above link and execute the Hive controller. Figure 25 shows an example of the interface displayed, when both endpoints are started with the AVB stack running.

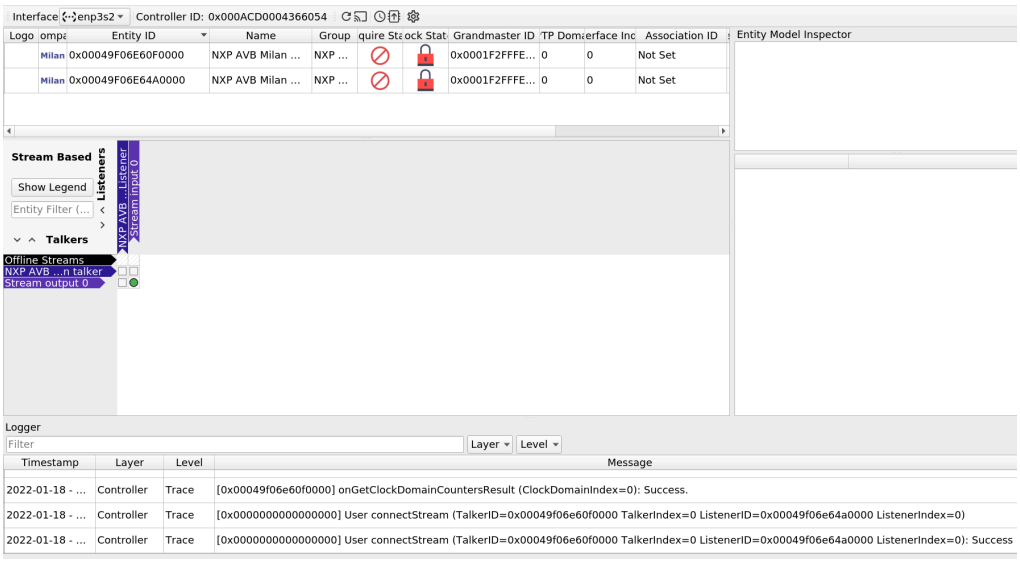

Figure 25. Hive controller interface

10.3 Evaluation instructions

On both talker and listener boards, power and boot the boards, and log in to the boards as root (no password).

The AVB stack starts automatically on each endpoint (per `genavb-tsn` systemd service configuration described in Section 2.3).

Once the stack is started on each endpoint, you can open the **Hive controller**. You must then see two Milan compatible entities detected.

Connect the stream outputs of the talker to the stream inputs of the listener. This step can be performed by clicking the corresponding box in the matrix of the ‘Stream Based’ section.

Once the streams are successfully connected, the selected box turns green.

The binding information including the talker, listener streams, and controller id is saved in the nonvolatile memory on the listener side. Therefore, the streams already bound are recovered automatically after a board reboot. To remove these parameters, an explicit unbind must be done on the controller (or an explicit removal of the binding file on board).

10.4 Audio file format and generation

The current Media Server application uses a raw audio format:

- Two channels
- 48 kHz
- 32 bits
- Big Endian numbering format

An i.MX Audio Sampler capable board can be used to generate such a file, by connecting an analog source (external player) to a Jack audio input port. The following ALSA command allows capturing and storing the file in the expected format:

```
# arecord-D <device> –t raw –c 2 –r 48000 –f S32_BE <filename>
```

To verify and listen to the captured file, connect a hearing device (headphones/speakers) to audio output Jack port or MIC. (In case of the SABRE-AI board, use the RCA/Jack cable connected to Audio Line OUT port.)

Use the following ALSA command:

```
# aplay -D <device> –c 2 –r 48000 –f S32_BE <file name>
```

The alsa device must be assigned to audio codec connected to jack interface (wm8960-audio).

11 More on evaluation usage

This section provides more information on the GenAVB/TSN evaluation package usage.

11.1 AVB stack start/stop

By default, the GenAVB/TSN stack does not start on boot. However, a `systemd` service is available to enable that option. The service file can be found at `/lib/systemd/system/genavb-tsn.service`. It can be disabled or re-enabled by using the following commands:

```
# systemctl <enable|disable> genavb-tsn
```

The AVB stack can be manually started and stopped at any time using the following command:

```
# systemctl start genavb-tsn
```

It loads the kernel modules, starts the TSN daemon if it is not running, sets up device nodes and processes priorities, and starts both the AVB daemon and demo application.

```
# systemctl stop genavb-tsn
```

It stops the AVB application but keeps the TSN daemon (gPTP and SRP stacks) running (so that PTP synchronization can be kept even if the AVB stack is restarted).

Note:

1. *To stop or start the gPTP and SRP stacks independently, use the following command:*

   ```
   # tsn.sh <start|stop>
   ```

2. *To stop or restart all the daemons and apps (AVB and TSN) of the stack, use the following command:*

   ```
   # avb.sh <stop_all|restart_all>
   ```

3. *Executing `avb.sh <start|stop>` can still be used to start/stop the stack with the same manner. However, it is recommended to use `systemctl` to keep `systemd` updated about the status of the service, as shown below:*

   ```
   # systemctl status genavb-tsn
   ```

4. *The logs from the GenAVB/TSN service can be found in the `systemd` journal by executing the command below:*

   ```
   # journalctl
   ```

11.2 AVB stack logs

The AVB stack logs contain additional information about the stack behavior, in particular about the playback delays. Logs are stored in `/var/log/avb`. This is a `tmpfs` that is lost on reboot.

- Linux command:

```
# tail -f /var/log/avb
```

- Focus on AVB statistics displayed every 10 seconds in the log file:

```
# tail -f /var/log/avb | grep "stats_print"
```

- Meaning of media stack statistics:

```
avtp stream_listener_stats_print : now-rx_ts   37/  58/  95
```

- In the above message, min/average/max in µs signifies the difference between the time the media stack received the packet and the time the ENET timestamped the packet on the receiver.

```
avtp stream_listener_stats_print : avtp_ts-now 1772/1809/1836
```

- In the above message, min/average/max in µs signifies the difference between the AVTP timestamp inside the packet and the time the packet was received by the media stack.

11.3 SRP stack logs

SRP stack runs inside the TSN process. Logs for SRP stack are stored in `/var/log/tsn` . This is a `tmpfs` that is lost on reboot.

- Linux command:

```
# tail -f /var/log/tsn | grep srp
```

11.4 AVB applications logs

AVB applications logs provide more information about the running media application. Logs are stored in `/var/log/avb_media_app`. This is a `tmpfs` that is lost on reboot.

- Linux command:

```
# tail -f /var/log/avb_media_app
```

Displayed statistics depend on the application.

- Meaning of media stack statistics:

```
alsa latency  895/1020/1187
```

The above command displays min/average/max in µs for the playback delay of frames sent to the ALSA stack.

11.5 gPTP endpoint statistics

The gPTP stack runs inside the TSN process. Logs for gPTP are stored in `/var/log/tsn`, but a gPTP filtered log file is available in `/var/log/fgptp`. This is a `tmpfs` that is lost on reboot.

- Linux command:

```
# tail -f /var/log/fgptp
```

- If the stack is configured in automotive mode, then the log contains:

```
Running fgptp in automotive profile on interface eth0
```

- *Port Role*, *Port AS-capability*, *link Status*, *neighbor capability* and *delay mechanism* are reported each time there is a change in the link state (link is 802.1AS capable or not) or upon Grandmaster change. This information is also displayed regularly along with current synchronization and pdelay statistics:

```
Port(0): domain(0,0): role changed from DISABLED to SLAVE
…
Port(0): domain(0,0): Slave – Link: Up – asCapable: Yes neighborGptpCapable:
 Yes delayMechanism: P2P
```

- The selected Grandmaster capabilities are reported upon new Grandmaster selection.
  - *Root Identity* represents the clock ID of the currently selected Grandmaster.
  - *Priority1, Priority2, Class* and *Accuracy* describe the clock quality of the selected Grandmaster.
  - *Peer Time Transmitter Identity* identifies the nearer peer time transmitter clock the current device is connected to (for example, a switch placed between the time receiver device and the Grandmaster):

```
Grand master: root identity 00049ffffe039e35
Grand master: priority1 245 priority2
Grand master: class 248 accuracy 248
Grand master: variance 17258
Grand master: source port identity 0001f2fffe0025fe, port number 2
```

  - *Synchronization State* is reported upon Grandmaster selection (SYNCHRONIZED) or when no Grandmaster are detected (NOT SYNCHRONIZED). Synchronization *Time* expressed in ms represents the time it took to the local clock to reach synchronization threshold starting from the first SYNC message received.

```
Port(0) SYNCHRONIZED – synchronization time (ms): 250
```

  - *Pdelay* (propagation delay) and local clock adjustments are displayed every 5 seconds.
    - *PDelay* is expressed in ns units and represents the one-way delay from the endpoint and its peer time transmitter clock.
    - *Correction* is expressed in parts per billion and represents the frequency adjustment performed to the local clock.
    - *Offset* is expressed in ns. It represents the resulting difference between the locally adjusted clock and the reference gPTP Grandmaster's clock. (Min/Max/Avg and Variance are computed for both Correction and Offset statistics)

```
Port(0): domain(0,0): Propagation delay (ns): 37.60  min 34 avg 36 max 45
 variance 17
Port(0): domain(0,0): Correction applied to local clock (ppb): min -5603 avg
 5572 max 5538 variance 148
Port(0): domain(0,0): Offset between GM and local clock (ns) min -12 avg 4
 max 22 variance 111
```

  - The Table 2 lists the port statistics (32 bits counters) displayed every 15 seconds on time receiver and time transmitter entities:

Table 2. Port statistics (for 32 bits counters)

Counter type	Statistics displayed
Receive counters	
`PortStatRxPkts`	Number of gPTP packets received (ether type 0x88F7)
`PortStatRxSyncCount`	Number of SYNC packets received
`PortStatRxSyncReceiptTimeouts`	Number of FOLLOW-UP packets timeout
`PortStatRxFollowUpCount`	Number of FOLLOW-UP packets received
`PortStatRxAnnounce`	Number of ANNOUNCE packets received

Table 2. Port statistics (for 32 bits counters)*...continued*

Counter type	Statistics displayed
PortStatAnnounceReceiptTimeouts	Number of ANNOUNCE packets timeout
PortStatAnnounceReceiptDropped	Number of ANNOUNCE packets dropped by the entity
PortStatRxSignaling	Number of SIGNALING packets received
PortStatRxPdelayRequest	Number of PDELAY REQUEST packets received
PortStatRxPdelayResponse	Number of PDELAY RESPONSE packets received
PortStatPdelayAllowedLostResponsesExceeded	Number of excess of allowed lost responses to PDELAY requests
PortStatRxPdelayResponseFollowUp	Number of PDELAY FOLLOW-UP packets received
PortStatRxErrEtype	Number of ether type errors (not 0x88F7)
PortStatRxErrPortId	Number or port ID errors
Transmit counters	
PortStatTxPkts	Number of gPTP packets transmitted
PortStatTxSyncCount	Number of SYNC packets transmitted
PortStatTxFollowUpCount	Number of FOLLOW-UP packets transmitted
PortStatTxAnnounce	Number of ANNOUNCE packets transmitted
PortStatTxSignaling	Number of SIGNALING packets transmitted
PortStatTxPdelayReques	Number of PDELAY REQUEST packets transmitted
PortStatTxPdelayResponse	Number of PDELAY RESPONSE packets transmitted
PortStatTxPdelayResponseFollowUp	Number of PDELAY FOLLOW-UP packets transmitted
PortStatTxErr	Number of transmit errors
PortStatTxErrAlloc	Number of transmit packets allocation errors
Miscellaneous counters	
PortStatAdjustOnSync	Number of adjustments performed upon SYNC received
PortStatMdPdelayReqSmReset	Number of resets of the PDELAY REQUEST state machine
PortStatMdSyncRcvSmReset	Number of resets of the SYNC RECEIVE state machine
PortStatHwTsRequest	Number of egress timestamp requests
PortStatHwTsHandler	Number of egress timestamp notifications
PortStatNumSynchronizationLoss	Number or synchronization loss on the time receiver endpoint (for example, Grandmaster change, Grandmaster reference clock discontinuity, and so on)
PortStatNumNotAsCapable	Number of transitions from AS_Capable=TRUE to AS_Capable=FALSE

11.6 gPTP bridge statistics

gPTP stack is running inside the TSN process. Logs for gPTP are stored in /var/log/tsn-br, but a gPTP filtered log file is available in /var/log/fgptp-br.

This is a tmpfs that is lost on reboot.

- Linux command:

```
# tail -f /var/log/fgptp-br
```

- The bridge stack statistics is similar to the endpoint stack ones except that the parameters are reported for each of the external ports of the switch (Port 0 to 3) and also for the internal port connected to the endpoint stack (Port 4) in case of Hybrid setup.
- *Pdelay* (propagation delay), *Link status*, *AS capability*, *Port Role*, *neighbor capability* and *delay mechanism* are displayed for each port as shown below:

```
Port(0): Role: Disabled    Link: Up asCapable: No neighborGptpCapable: No
 delayMechanism: P2P
```

```
Port(1): Role: Disabled    Link: Up asCapable: No neighborGptpCapable: No
 delayMechanism: P2P
```

```
Port(2): domain(0,0): Role: Disabled    Link: Up asCapable: No
 neighborGptpCapable: Yes delayMechanism: P2P

Port(2): Propagation delay (ns): 433.98 min 425 avg 438 max 457 variance 87
```

```
Port(3): domain(0,0): Role: Disabled    Link: Up asCapable: No
 neighborGptpCapable: No delayMechanism: P2P

Port(4): domain(0,0): Role Master    Link: Up    asCapable: Yes
 neighborGptpCapable: Yes delayMechanism: P2P

Port(4): Propagation delay (ns): 433.98 min 425 avg 438 max 457 variance 87
```

11.7 Using the GenAVB/TSN AVDECC controller

The evaluation package includes an AVDECC controller demo application. This application can be invoked from the Linux command prompt of any endpoint running the GenAVB/TSN stack:

```
# genavb-controller-app -h

NXP's GenAVB AVDECC controller demo application

Usage:
app [options]

Options:
-S, --set-control <control_type> <entity_id> <control_index> <value>
Set a given control to the given value where control_type must be uint8 or utf8
 (For utf8: <value> must be string of max 99 characters)
-G, --get-control <control_type> <entity_id> <control_index>
Get a control value where control_type must be uint8 or utf8
-l, --list-entities
List discovered AVDECC entities
-c, --connect-stream <talker_entity_id> <talker_unique_id> <listener_entity_id>
 <listener_unique_id> <flags>
Connect a stream between a talker and a listener
-d, --disconnect-stream <talker_entity_id> <talker_unique_id>
 <listener_entity_id> <listener_unique_id>
Disconnect a stream between a talker and a listener
-r, --get-rx-state <listener_entity_id> <listener_unique_id>
Get information about a listener sink
-t, --get-tx-state <talker_entity_id> <talker_unique_id>
```

```
Get information about a talker source
-s, --get-tx-connection <talker_entity_id> <talker_unique_id> <index>
Get information from a talker about a given connection/stream
-T, --listener-streaming <talker_entity_id> <talker_unique_id> <start|stop>
Send START_STREAMING or STOP_STREAMING command to a talker
-L, --talker-streaming <listener_entity_id> <listener_unique_id> <start|stop>
Send START_STREAMING or STOP_STREAMING command to a listener
-n, --set-name <entity_id> <configuration_id> <descritptor_type>
 <descriptor_index> <name_index> "<name>"
Send SET_NAME command to a descriptor
--list-clock-domain-sources <entity_id> <clock_domain_index>
List clock sources of a clock domain
--set-clock-domain-source <entity_id> <clock_domain_index> <clock_source_index>
Set the clock source of a clock domain
-h
Print this help text
```

The "`-l`" option lists information and characteristics of all the AVDECC entities declared by the AVB endpoints present on the network. This information is used to control AVB activity among the endpoints using other available options as described in the help output.

Logs from the AVDECC controller application can be displayed by using the following command:

```
# tail -f /var/log/avb_avdecc_controller
```

11.8 Net AVB kernel module statistics

The module exports multiple stats through debugfs under `/sys/kernel/debug/avb/`

- Software FQTSS and TX stats: Stats about the different TX streams/traffic classes and the software FQTSS used in the net AVB kernel module are shown under `/sys/kernel/debug/avb/tx/`
- RX stats: Stats about received packets in the net AVB kernel module with their different protocols are shown under: `/sys/kernel/debug/avb/rx/`
- Hardware timer stats: Stats about the hardware timer used to drive the packets' processing in the net AVB kernel module are shown under: `/sys/kernel/debug/avb/hw_timer/`
- Media clock stats: Stats about the medial clock driver in the net AVB kernel module are shown under: `/sys/kernel/debug/avb/mclock/`

When the AVB endpoint is configured as media clock receiver with enabled media clock recovery, the stats are shown under: `/sys/kernel/debug/avb/mclock/rec_pll_0`

```
# cat /sys/kernel/debug/avb/mclock/rec_pll_0
```

When the recovery mechanism is running, the stats must show the number of adjustment "adjust" increasing on multiple reads and must display the drift between the audio clock (time receiver clock) and the time transmitter clock.

The latest applied adjustment (in ppb) to the audio PLL is shown under the field: "`last applied ppb adjust`"

11.9 Setting a static IP address for the network interface

The system uses *systemd-networkd* service to manage network configurations.

To configure the network interfaces (either as DHCP client or static IP address), a network configuration file can be added at `/etc/systemd/network/70-eth0.network`.

To configure the eth0 interface with a static IP address, set the desired IP address as in the following example:

```
[Match]
Name=eth0
[Network]
Address=192.168.1.5/16
Gateway=192.168.1.1
```

Then, restart the systemd-networkd service for the new configuration to take effect:

```
# systemctl restart systemd-networkd.service
```

The changes are saved across reboots.

12 Known Issues

Table 3. Known issues

Issue ID	Description	Workaround
GENAVB-2570	On the first audio stream connection after a fresh boot, the AVTP timestamps of the i.MX 8DXL are late when set as CRF time transmitter clock and Audio Talker. Thus no audio is played on the Audio Listener.	Perform a stack restart on the i.MX 8DXL with `avb.sh restart` or disconnect and reconnect the audio stream.

13 Note about the source code in the document

Example code shown in this document has the following copyright and BSD-3-Clause license:

Copyright 2017-2025 NXP Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials must be provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

14 Revision history

Table 4 summarizes the revisions to this document.

Table 4. Document revision history

Document ID	Release date	Description
GENAVBTSNUG v.1.8	15 December 2025	Updated for Real Time Edge Software Rev 3.3 release
GENAVBTSNUG v.1.7	29 July 2025	Updated for Real Time Edge Software Rev 3.2 release
GENAVBTSNUG v.1.6	26 March 2025	Updated for Real Time Edge Software Rev 3.1 release
GENAVBTSNUG v.1.5	25 July 2024	Updated for Real Time Edge Software Rev 2.9 release
GENAVBTSNUG v.1.4	29 March 2024	Updated for Real Time Edge Software Rev 2.8 release
GENAVBTSNUG v.1.3	18 December 2023	Updated for Real Time Edge Software Rev 2.7 release
GENAVBTSNUG v.1.2	28 July 2023	Updated for Real Time Edge Software Rev 2.6 release
GENAVBTSNUG v.1.1	30 March 2023	Updated for Real Time Edge Software Rev 2.5 release
GENAVBTSNUG v.1	16 December 2022	Updated for Real Time Edge Software Rev 2.4 release
GENAVBTSNUG v.0	28 July 2022	Initial version for Real Time Edge Software Rev 2.3 release

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

HTML publications — An HTML version, if available, of this document is provided as a courtesy. Definitive information is contained in the applicable document in PDF format. If there is a discrepancy between the HTML document and the PDF document, the PDF document has priority.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

Contents