Application Note: AN00110

MPEG Transport Stream over Ethernet AVB

This application note demonstrates how an MPEG Transport Stream can be streamed over an Ethernet AVB network, with guaranteed Quality of Service and time synchronization, using an AVB endpoint implemented on an XMOS multicore microcontroller.

The firmware associated with this application note is included in an XMOS AVB Reference software release and supports a standard Synchronous Parallel Interface[1] and 61883-4 encapsulation format.

The AVB Audio Endpoint Platform hardware (XR-AVB-LC-BRD) has been adapted to support a Synchronous Parallel Interface operating at TTL levels.

Required tools and libraries

- xTIMEcomposer Tools - Version 13.2
- XMOS AVB Endpoint reference software - Version 6.1.1
- Alitronika DVSStation3 application (Microsoft Windows only)

Required hardware

This application note is designed to run on an XMOS xCORE-L series device.

The example firmware provided in the reference design has been implemented and tested on the existing AVB Audio Endpoint Platform hardware (XR-AVB-LC-BRD). Third-party Transport Stream source/sink hardware from Alitronika (AT40XR2USB) was connected via an LVDS to TTL interposer PCB using a standard 25 contact type D subminiature connector.

There is no dependency on this hardware and the firmware can be modified to run on any xCORE-L series device interfaced to compliant MPEG-TS SPI hardware. LVDS to TTL buffers may be required.

Prerequisites

- This document assumes familiarity with the XMOS xCORE architecture, the IEEE AVB/TSN standards, the XMOS tool chain and the xC language. Documentation related to these aspects which are not specific to this application note are linked to in the references appendix.
- For descriptions of XMOS related terms found in this document please see the *XMOS glossary*[2].
- For the full API listing of the XMOS AVB Audio Endpoint reference software please see the the *AVB Endpoint Design Guide*[3].

[1] https://www.dvb.org/resources/public/standards/En50083-9.2002.pdf
[2] http://www.xmos.com/published/glossary
[3] https://www.xmos.com/published/avb-design-guide

1 Overview

1.1 Introduction

In a satellite, cable or terrestrial digital broadcast head end or set top box, the MPEG-TS data is typically transported between the broadband modulator/demodulator and the baseband stream multiplexors/demultiplexors using an MPEG Synchronous Parallel Interface or Asynchronous Serial Interface. This application note addresses implementation of the Synchronous Parallel Interface only.

1.2 MPEG Synchronous Parallel Interface

The MPEG SPI interface consists of 8 bits of data, a valid, a clock and a start of packet signal. The clock rate on the interface is in the range 0 - 13.5 MHz, giving a maximum data rate of 108 Mbit/s. A 100 Mbit/s AVB Ethernet connection permits 75 Mbit/s of bandwidth to be reserved in each direction.

The interface complies with the Synchronous Parallel Interface specification described in EN 50083-9:2002 4.2.

1.3 1722 transport stream encapsulation

The AVB firmware complies with the IEEE 1722-2011 specification, which describes how to encapsulate 61883-4 packets within the 1722 AVTP. Section 4 of the IEC 61883 specification details how to encapsulate 188 byte transport stream packets into Firewire-type 61883 isochronous packets.

1.4 Network overview

Figure 2 shows a overview of an AVB network consisting of a single Transport Stream Talker and Listener.

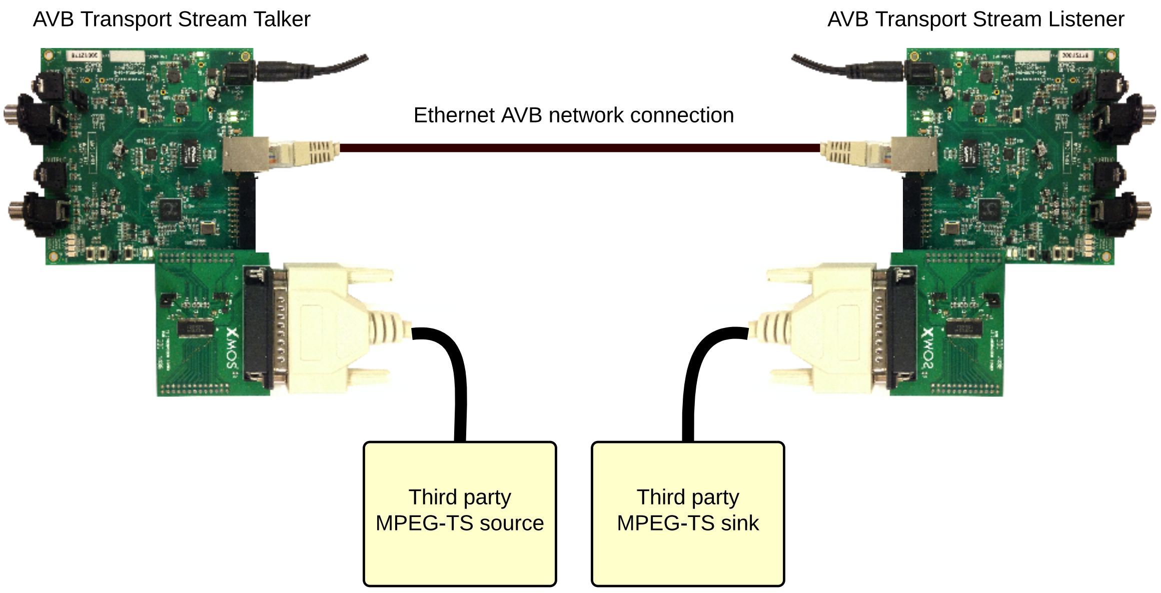

AVB Transport Stream Talker
AVB Transport Stream Listener
Ethernet AVB network connection
Third party MPEG-TS source
Third party MPEG-TS sink

Figure 1: XMOS AVB Talker and Listener network overview

2 Transport Stream Talker and Listener examples

The examples in this application note are contained within the AVB reference design software and show the configuration necessary to enable MPEG Transport Stream support within an application.

Two example applications are provided, `app_avb_tsi_talker` for Talker functionality, and `app_avb_tsi_listener` for Listener functionality.

2.1 Enabling MPEG Transport Stream format

The 1722 format must be defined globally as 61883-4 via the following #define in `avb_conf.h` within the application `src/` sub-directory:

```
/** Use 61883-4 MPEG-TS format for 1722 streams */
#define AVB_1722_FORMAT_61883_4 1
```

2.2 Declaring resources for Transport Stream SPI

Transport Stream SPI input on the AVB Talker requires a number of port resources and a clock block to be declared. For an example, see `app_avb_tsi_talker/src/main.xc`:

```
//***** Transport Stream SPI input resources ****
on tile[0]: in port p_ts_clk = XS1_PORT_1I;
on tile[0]: in port p_ts_valid = XS1_PORT_1H;
on tile[0]: in buffered port:4 p_ts_sync = XS1_PORT_1J;
on tile[0]: in buffered port:32 p_ts_data = XS1_PORT_8B;
on tile[0]: clock clk_ts = XS1_CLKBLK_1;
```

Ports can be changed like-for-like to suit a particular hardware/pin mapping. The ports in the example are connected to the expansion header on the XMOS AVB audio endpoint board (XR-AVB-LC-BRD). See Appendix A for further detail on the pinout of this connector.

Media FIFOs that interface between the SPI pins and the 1722 Talker are also declared at the top level `main.xc`:

```
media_input_fifo_data_t ififo_data[AVB_NUM_MEDIA_INPUTS];
media_input_fifo_t ififos[AVB_NUM_MEDIA_INPUTS];
```

Transport Stream SPI output on the AVB Listener requires a similar set of resources that are declared in `app_avb_tsi_listener/src/main.xc`:

```
//***** Transport Stream SPI output resources ****
on tile[0]: in port p_ts_clk = XS1_PORT_1I;
on tile[0]: out port p_ts_valid = XS1_PORT_1H;
on tile[0]: out buffered port:4 p_ts_sync = XS1_PORT_1J;
on tile[0]: out buffered port:32 p_ts_data = XS1_PORT_8B;
on tile[0]: clock clk_ts = XS1_CLKBLK_1;
```

```
media_output_fifo_data_t ofifo_data[AVB_NUM_MEDIA_OUTPUTS];
media_output_fifo_t ofifos[AVB_NUM_MEDIA_OUTPUTS];
```

The Transport Stream SPI input and output interfaces are available as part of the `module_avb_video` module. This module must be included in the USED_MODULES list in the application Makefile. See `app_avb_tsi_talker/Makefile` for an example.

SPI input is implemented as a task running on a separate logical core within the Talker application. It is prototyped in the the tsi_input.h header which must be included:

```
#include "tsi_input.h"
```

The interface is registered with the AVB manager and called in app_avb_tsi_talker/src/main.xc as follows:

```
on tile[0]:
{
  init_media_input_fifos(ififos, ififo_data, AVB_NUM_MEDIA_INPUTS);
  media_ctl_register(c_media_ctl[0], 1, ififos, 0, null, 0);
  tsi_input(clk_ts, p_ts_data, p_ts_clk, p_ts_sync, p_ts_valid, ififo_data[0]);
}
```

Similarly, the SPI output is prototyped in the tsi_output.h header and is called in app_avb_tsi_listener/src/main.xc within the main() function:

```
on tile[0]:
{
  init_media_output_fifos(ofifos, ofifo_data, AVB_NUM_MEDIA_OUTPUTS);
  media_ctl_register(c_media_ctl[0], 0, null, 1, ofifos, 0);
  tsi_output(clk_ts, p_ts_data, p_ts_clk, p_ts_sync, p_ts_valid, ofifo_data[0]);
}
```

2.3 Enabling the built-in 1722.1 Controller

An example 1722.1 Controller is enabled by default in the Transport Stream example applications. This functionality allows the Talker to automatically connect to the Transport Stream Listener without an external 1722.1 Controller present on the network.

The following #define should be set in avb_conf.h within the application directory to enable the Controller functionality:

```
/** Enable 1722.1 Controller functionality on the entity. */
#define AVB_1722_1_CONTROLLER_ENABLED 1
```

The #define can be set to 0 to disable this functionality.

APPENDIX A - Compiling and running the example firmware

A.1 Obtaining the latest firmware

1. Log into xmos.com and access *My XMOS* ▶ *Reference Designs*
2. Request access to the *AVB Endpoint Software* by clicking the *Request Access* link under *AVB Audio Endpoint*. An email will be sent to your registered email address when access is granted.
3. A *Download* link will appear where the *Request Access* link previously appeared. Click and download the firmware zip.

A.2 Installing xTIMEcomposer Studio

The AVB-LC software requires xTIMEcomposer version 13.2, available for download at xmos.com[4].

A.3 Importing and building the firmware

To import and build the firmware, open xTIMEcomposer Studio and follow these steps:

1. Choose *File* ▶ *Import*.
2. Choose *General* ▶ *Existing Projects into Workspace* and click **Next**.
3. Click **Browse** next to **'Select archive file'** and select the firmware .zip file downloaded in section 1.
4. Make sure that all projects are ticked in the *Projects* list.
5. Click **Finish**.
6. Select the `app_avb_tsi_talker` project in the Project Explorer and click the **Build** icon in the main toolbar.
7. Repeat step 6 for the `app_avb_tsi_listener` project.

A.4 Installing the application onto flash memory

1. Connect the xTAG-2 debug adapter (XA-SK-XTAG2) to the first AVB endpoint board.
2. Plug the xTAG-2 into your development system via USB.
3. Plug in the 5V power adapter and connect it to the AVB endpoint board.
4. In xTIMEcomposer, right-click on the binary within the *app_avb_tsi_talker/bin* folder of the project.
5. Choose *Flash As* ▶ *Flash Configurations*.
6. Double click *xCORE Application* in the left panel.
7. Choose *hardware* in *Device options* and select the relevant xTAG-2 adapter.
8. Click on **Apply** if configuration has changed.
9. Click on **Flash**. Once completed, reset the AVB endpoint board using the reset button.
10. Repeat steps 1 through 8 for the second endpoint and the `app_avb_tsi_listener` binary.

A.5 Setting up the hardware

Refer to Appendix B for details on an example setup with third party Transport Stream SPI hardware. To make an AVB connection between the Talker and Listener endpoints:

1. Connect the two development boards together via the provided Ethernet cable.
2. If not already powered, connect the power supplies to the input power jacks of the boards and power them on.
3. The Talker will automatically make a connection to the Listener via a built-in 1722.1 controller running on the endpoints.

[4] `http://www.xmos.com/support/xtools`

APPENDIX B - Example hardware setup

An interface PCB connects the XMOS AVB audio endpoint board (XR-AVB-LC-BRD) to third-party transport stream hardware. The interface PCB provides differential signal to single ended conversion, a 13.5 MHz clock for use on DVB-SPI output, and a standard 25 contact type D subminiature connector.

The interface attaches to the expansion header of the audio endpoint board. The port/pin assignments used in the example firmware are:

Header Pin	xCORE Port	xCORE Pin	DVB-SPI Signal	Comment
16	XS1_PORT_1H	X0D24	Valid	1-bit port
17	XS1_PORT_1I	X0D23	Clock	1-bit port
18	XS1_PORT_1J	X0D25	Sync	1-bit port
21-28	XS1_PORT_8B	X0D14 - X0D21	Data	8-bit port

Figure 2 shows the hardware setup for a single endpoint.

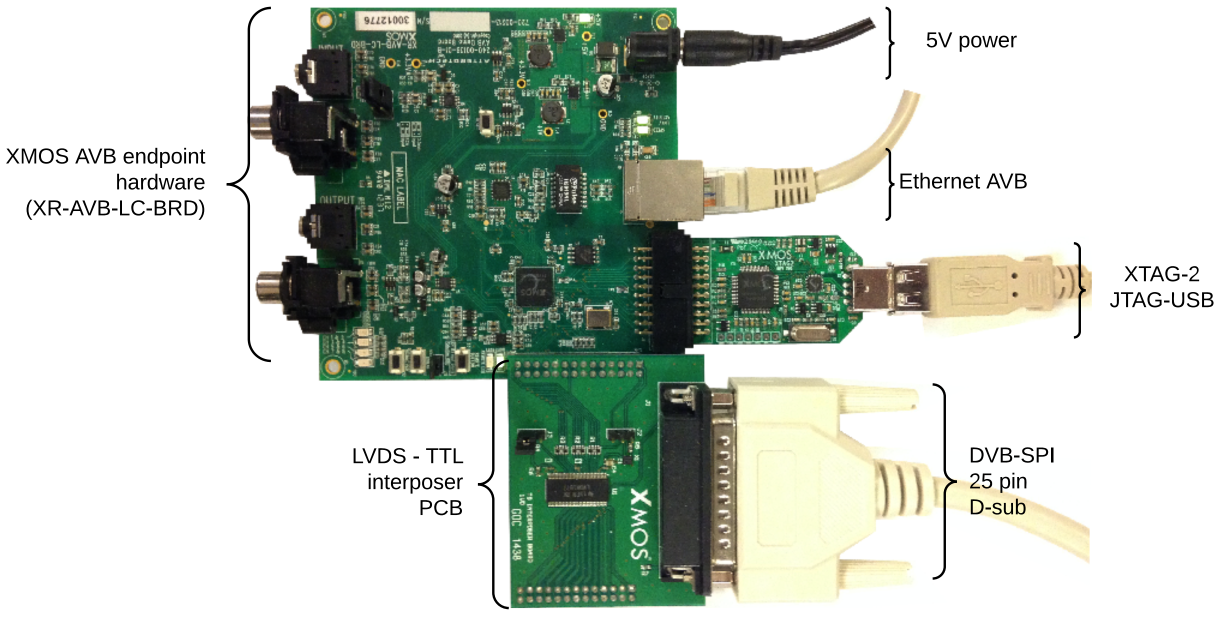
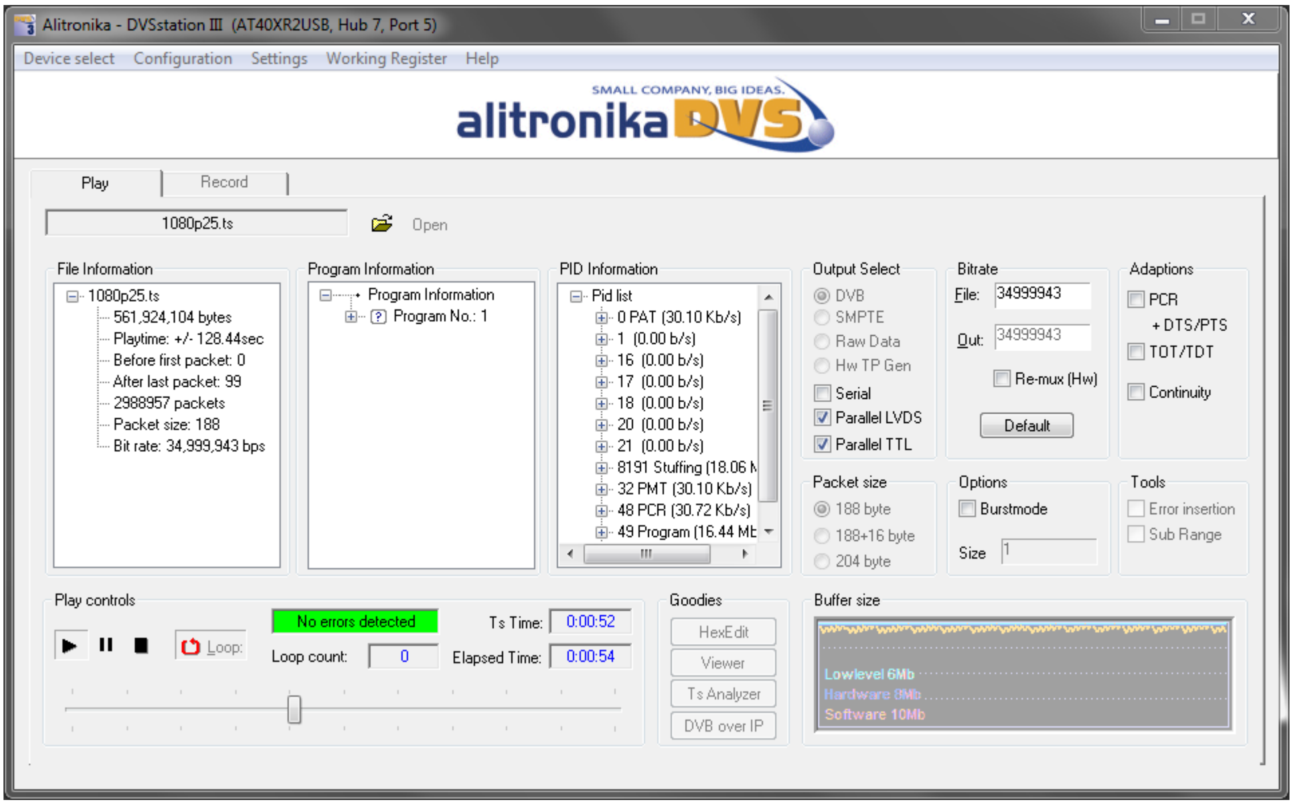

5V power
XMOS AVB endpoint hardware (XR-AVB-LC-BRD)
Ethernet AVB
XTAG-2 JTAG-USB
LVDS - TTL interposer PCB
DVB-SPI 25 pin D-sub

Figure 2: Hardware setup for DVB-SPI connection

B.1 Third-party Transport Stream hardware

Two *iDo it All AT40XR2USB* devices from Alitronika can be used, one as a Transport Stream SPI source and the other as a sink. The DVB–SP1 LVDS input and output can be connected to a 25 pin D-sub connector on the interface PCB described above.

The Alitronika devices are configured using the *Alitronika DVSStation3* software running on a Microsoft Windows PC host. An example configuration for Transport Stream playback is show in Figure 3. *Output Select* should be set to *DVB - Parallel LVDS* and the *Re-mux (Hw)* setting under *Bitrate* should be disabled. *Packet size* should be set to *188 byte*.



Alitronika - DVSstation III (AT40XR2USB, Hub 7, Port 5)

1080p25.ts

File Information
- 1080p25.ts
  - 561,924,104 bytes
  - Playtime: +/- 128.44sec
  - Before first packet: 0
  - After last packet: 99
  - 2988957 packets
  - Packet size: 188
  - Bit rate: 34,999,943 bps

Program Information
- Program Information
  - Program No.: 1

PID Information
- Pid list
  - 0 PAT (30.10 Kb/s)
  - 1 (0.00 b/s)
  - 16 (0.00 b/s)
  - 17 (0.00 b/s)
  - 18 (0.00 b/s)
  - 20 (0.00 b/s)
  - 21 (0.00 b/s)
  - 8191 Stuffing (18.06 M
  - 32 PMT (30.10 Kb/s)
  - 48 PCR (30.72 Kb/s)
  - 49 Program (16.44 Mb

Output Select: [✓ DVB] SMPTE Raw Data Hw TP Gen; Serial ☐, Parallel LVDS ✓, Parallel TTL ✓

Bitrate: File: 34999943; Out: 34999943; Re-mux (Hw) ☐

Adaptions: PCR ☐ + DTS/PTS; TOT/TDT ☐; Continuity ☐

Packet size: [✓ 188 byte] 188+16 byte 204 byte

Options: Burstmode ☐; Size: 1

Tools: Error insertion ☐; Sub Range ☐

Play controls: No errors detected; Loop count: 0; Ts Time: 0:00:52; Elapsed Time: 0:00:54

Buffer size: Lowlevel 6Mb; Hardware 8Mb; Software 10Mb

Figure 3: DVSStation3 Playback configuration

Similarly, an example of Transport Stream record is shown in Figure 4. *Input Select* should be set to *DVB - Parallel LVDS*. The *Signal Info* should show green lights for *Carrier Detect*, *Lock* and *Sync* when the Playback is active and *Record controls* is set to *Start recording* or *Monitor input stream*.

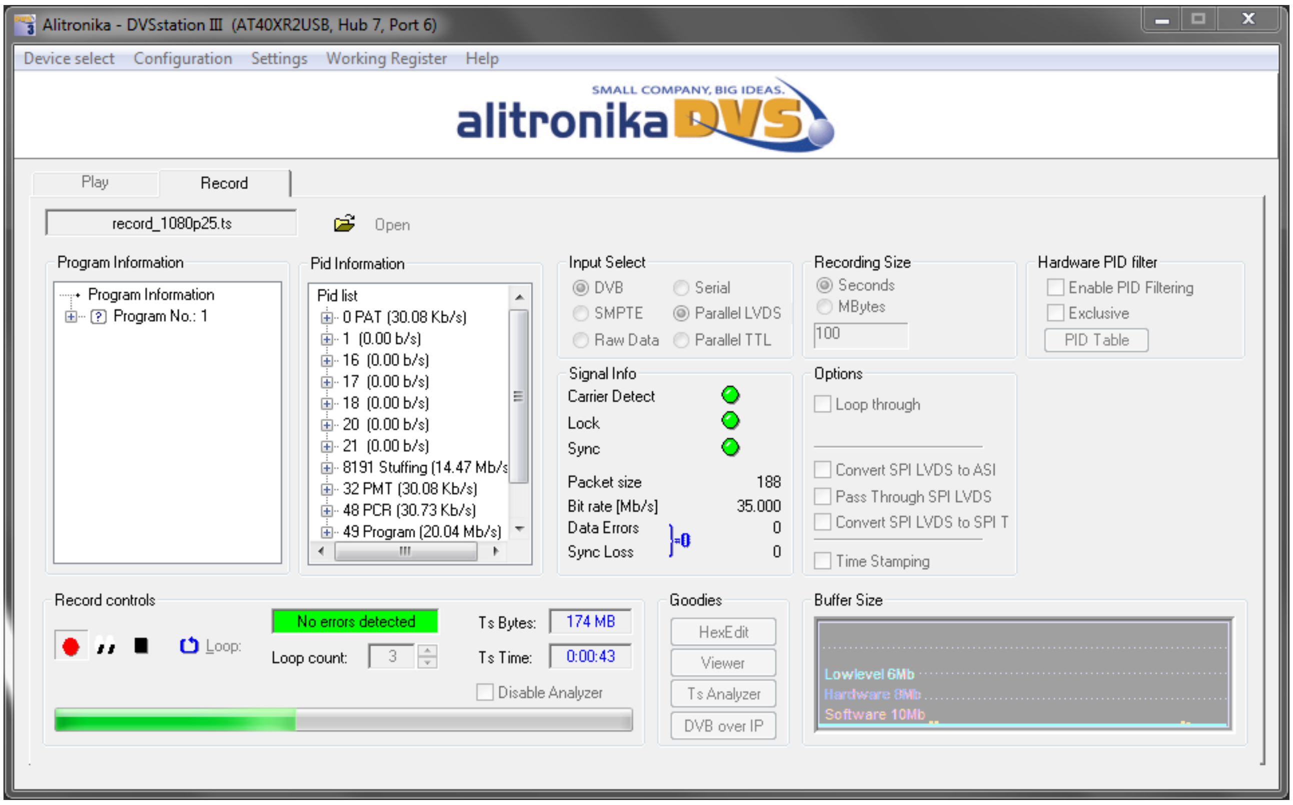
Please refer to the *DVSStation3 manual*[5] for more information on how to setup the devices.

B.2 MPEG Transport Stream Test Patterns

Example MPEG Transport Stream test patterns can be obtained from http://www.w6rz.net. The following streams were tested with the Alitronika setup:

Title	Format	TS Bitrate	File link
Elephants Dream	1080p@ 24, 40 Mbps CBR	48 Mbps	http://www.w6rz.net/ed24p_01.zip
Sony HDW-F900 footage	1080p@ 25, 18 Mbps av	35 Mbps	http://www.w6rz.net/1080p25.zip

[5] http://www.alitronika.com/datasheet/dvsstation3_user_manual.pdf



Alitronika - DVSstation III (AT40XR2USB, Hub 7, Port 6)

Device select | Configuration | Settings | Working Register | Help

Play | Record

record_1080p25.ts — Open

Program Information
- Program Information
  - Program No.: 1

Pid Information
Pid list
- 0 PAT (30.08 Kb/s)
- 1 (0.00 b/s)
- 16 (0.00 b/s)
- 17 (0.00 b/s)
- 18 (0.00 b/s)
- 20 (0.00 b/s)
- 21 (0.00 b/s)
- 8191 Stuffing (14.47 Mb/s
- 32 PMT (30.08 Kb/s)
- 48 PCR (30.73 Kb/s)
- 49 Program (20.04 Mb/s)

Input Select: [✓ DVB] Serial SMPTE [✓ Parallel LVDS] Raw Data Parallel TTL

Recording Size: [✓ Seconds] MBytes — 100

Hardware PID filter: ☐ Enable PID Filtering ☐ Exclusive — PID Table

Signal Info

Carrier Detect	●
Lock	●
Sync	●
Packet size	188
Bit rate [Mb/s]	35.000
Data Errors	0
Sync Loss	0

Options: ☐ Loop through ☐ Convert SPI LVDS to ASI ☐ Pass Through SPI LVDS ☐ Convert SPI LVDS to SPI T ☐ Time Stamping

Record controls: Loop: — No errors detected — Loop count: 3 — Ts Bytes: 174 MB — Ts Time: 0:00:43 — ☐ Disable Analyzer

Goodies: HexEdit | Viewer | Ts Analyzer | DVB over IP

Buffer Size: Lowlevel 6Mb, Hardware 8Mb, Software 10Mb

Figure 4: DVSStation3 Record configuration

APPENDIX C - References

XMOS Tools User Guide

http://www.xmos.com/published/xtimecomposer-user-guide

XMOS xCORE Programming Guide

http://www.xmos.com/published/xmos-programming-guide

XMOS AVB Design Guide:

http://www.xmos.com/published/avb-design-guide

IEEE 1722-2011:

http://standards.ieee.org/findstds/standard/1722-2011.html

IEC 61883-4:

http://webstore.iec.ch/Webstore/webstore.nsf/ArtNum_PK/32987

EN 50083-9 2002

https://www.dvb.org/resources/public/standards/En50083-9.2002.pdf

Alitronika iDo it All AT40XR2USB

http://www.alitronika.com/at40xr2usb.htm