

# Linux Network Enabling Requirements for Audio/Video Bridging (AVB) Linux Plumbers 2012

Eric Mann Software Architect - Intel

August 29 2012

INTEL CORPORATION ° 2012 ALL RIGHTS RESERVED

### Legal Disclaimer

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY RELATING TO SALE AND/OR USE OF INTEL PRODUCTS, INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT.

All information provided related to future Intel products and plans is preliminary and subject to change at any time, without notice.

Intel, the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

\* Other names and brands may be claimed as the property of others.

Copyright © 2012





## Audio Video Bridging (AVB) Overview

AVB on Linux

Questions



## Why AVB?

People have been sending audio & video over networks for decades now ... so what is new about AVB?

- Buffering data to hide packet loss and bandwidth issues causes presentation lag (the echo cancellation in your ear tolerates up to 10 msec)
- Sending audio to multiple devices requires synchronization of the presentation-time (e.g. lip-sync of video to audio, warbling in 7.1 audio systems),
- Auto companies want to use AVB to reduce cost.





intel

### What is "AVB"? for this discussion at least ...\*

AVB := 802.1Qat + 802.1Qav + 802.1AS

- 802.1Qat := MMRP + MVRP + MSRP = "Stream Reservation"
- 802.1Qav := "Hardware-based traffic shapers"
- 802.1AS := "Time Synchronization"

\* There are a number of additional user-mode protocols layered on top to handle AVB device discovery & control, as well as media framing:

- L2-based IEEE 1722, 1722.1
- L3-based 1733 & RTP
- I also don't talk about the receiving end of these streams as they only really need to support 802.1AS and MRP – no need for traffic shaping.



### **Functional Overview**

### Streams to Classes to User Priorities to Ethernet



#### Notes

Each stream contains zero (0) or more media channels.

As streams are added/removed to a class, bandwidth of the <u>class</u> is adjusted.

Discovery of UserPrio and VLAN made during Qat (the "DOMAIN" msg).



inte



# Proof of Concept Linux Stack [DEMO]

e1000.soureforge.net /linux\_igb\_avb



#### Notes

In the current design, one of the queues is kept by the in-kernel igb module for best-effort traffic. Media queues are mapped to userspace.



### **Remember DCB/FCoE? AVB drops right in ...**





# Gap 1 - Timed Transmission

- Prior patches enabled passing timestamp information to applications on receive, and returning tx timestamps on the ERRMSG queue ...
- Need to extend timestamps to control actual packet transmit time ...
  - As the class shaper is adjusted, the class shaper may momentarily send more or less data on existing streams with traffic outstanding on a class. This can cause over-runs or under-runs on receivers.
  - Helps with sequencing of streams onto the same Class shaper (e.g. multiple apps write onto streams and expects the class shaper to pace out the traffic fairly ...) ← can workaround with driver-based knowledge of individual streams (not desirable).
  - This works with L2 or message-based protocols (like UDP) not stream based like TCP however.



# CMSG PROPOSAL for Timed Tx

New CMSG type to indicating timed transmission (as well as corresponding transmission time).

- Transmission time is relative to the physical interface.
- User-mode application will need to perform translation if required from the global "AS" time to the physical interface relative time.

```
cmsg->cmsg_level = SOL_RAW; /* UDP too? */
cmsg->cmsg_type = TIMED_TX;
cmsg->cmsg_len = CMSG_LEN(8); /* seconds.nsec format */
/* Initialize the payload: */
tx_time = (u_int64_t*) CMSG_DATA(cmsg);
*tx time = tx seconds << 32 | tx nsec;</pre>
```



# Gap (2) – ethtool support

Would be desirable to dynamically configure "AVB" functionality (enable/disable) via ethtool.

- Based on our experience, requires device reset.
- New option to display as well as change "AV" state of adapter interface.



## For More Information & Planned Future Steps

- "linux\_igb\_avb" tarballs of this code are posted to Intel's existing e1000.sourceforge.net project.
- In process of establishing a github repos\* for open source collaboration and contribution \*project name pending Intel Legal approval
- Enhancements to example applications.





# Q & A?



INTEL CORPORATION ° 2012 ALL RIGHTS RESERVED

#### P1722 packet format for 61883-6/AM824 (Multi-bit linear audio) 48kHz stereo stream

	802.3 MAC header	00	DA (MAC destination address, upper 32 bits)				
		00					
		04	DA (MAC destination address, lower 16 bits) SA (MAC source address, upper 16 bits)				
		08	SA (MAC source address, lower 32 bits)				
	VLAN Tag field	12	TPID (802.1Q = 8100 <sub>16</sub> ) PCP <sup>C</sup> <sub>F</sub> VID (VLAN Identifier)				
	AVBTP type/cntl	16	EtherType (AVBTP = 88B5 <sub>18</sub> ) <sup>C</sup> <sub>D</sub> subtype stream_reserved <sup>t</sup> <sub>v</sub>				
	802.1Qat Stream ID	20	stream_id (upper 32 bits)				
		24	stream_id (lower 32 bits)				
	AVBTP Time	28	avbtp_presentation_time				
			fa recovered framment lon (quadletr) stream conjugater)				
	Fragment header	32	ic reserved inagment_ien (quadiets) siteam_sequence (quadiets)				
	Packet header	36	nacket data length (octets) tagchannel (= 31) tcodesv				
			packet_data_length (octets) tag channel (= 31) (code sy (= A <sub>16</sub> )				
	61883 CIP header	40	EOH/ SID (= 63) DBS FN QPC <sup>8</sup> / <sub>8</sub> rsv DBC				
		40					
			EOHV FMT FDF SYT				
		44					
	61883-6/AM824 (sample #1)	48	label 24-bit audio sample (Left channel)				
		40					
		52	label 24-bit audio sample (Right channel)				
	61883-6/AM824 (sample #2)	56	label 24-bit audio sample (Left channel)				
		60	label 24-bit audio sample (Right channel)				
	(Samples 3-6)	/					
	• • •						
	FCS	96	Ethernet_crc				



Subtype data	00	D subtype	stream_reserved	t subtype_data2	gm_info		
		0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0			
	04	stream_id (upper 32 bits)					
802.1Qat							
Stream ID	<mark>08</mark>	stream_id (lower 32 bits)					
	12	avbtp_timestamp					
AVBIP TIME							
Fragment	16	fc pkt_hdr_len	fragment_len (quadlets)	stream_seque	ence (quadlets)		
header							
Packot boador	20	packet_data_	length (octets)	tag channel (= 31)	tcode sy		
Facket neader		0 0 0 0 0 0 1 0			(= A <sub>16</sub> ) 1 0 1 0 0 0 0		
-	24	EOH/ SID (= 63)	DBS	FN QPC P rsv	DBC		
61883		0 0 1 1 1 1 1 1 1					
CIP header	28	EOH/ FMT	FDF	S	YT		
		1 0 0 0 0 0 0 1	reserved				
	32	Comp	ression Mode Specific Inf	ormation	r Ver Type		
		VDSPC		Line number			
	36	byte 1 video data	byte 2 video data	byte 3 video data	byte 4 video data		
61883-8 Source Packet -							
Data				•••			
Data							
	750	byte 717 video data	byte 718 video data	byte 719 video data	byte 720 video data		
	152						

#### P1722 packet format for 61883-8, Source Packet Type=0 (video data)

