Providing Interoperability of, and Control over, Quality of Service Networks for Real-time Audio and Video Devices

Philip J. Foulkes and Richard J. Foss
Audio Research Group
Department of Computer Science
Rhodes University, P.O. Box 94, Grahamstown, 6140
Tel: +27 46 603 8291, Fax: +27 46 636 1915
E-mail: {p.foulkes, r.foss}@ru.ac.za

***Abstract* – In the IT industry there is a trend towards the convergence of the transportation of different types of content to single networking technologies. Web, e-mail and VoIP applications all use a common network to transport their data. This convergence leads to lower equipment costs and allows for more future-proof networks to be built.**

Audio and video production environments tend to use different technologies for the transportation of different types of data. Different types of equipment use varying command and control protocols. There are strict bandwidth, latency and timing requirements that need to be met for transporting real-time audio and video data. IEEE 1394 is a standards-based networking technology that meets these requirements. Ethernet AVB is an up and coming standards-based networking technology that is designed to meet these requirements. This paper discusses the initial phase of an investigation into the interoperability of, and control over, IEEE 1394 and Ethernet AVB networks. A tunnelling application has been developed that allows Ethernet devices to take advantage of the capabilities of IEEE 1394. Control over the parameters of the tunnelling software is performed by an IP based command and control protocol, AES-X170.

***Index Terms* – AES-X170, Ethernet AVB, IEEE 1394, Quality of Service, Real-time audio and video streaming**

I. INTRODUCTION

There is a trend in the Information Technology (IT) environment towards a convergence of technologies where various types of content (e.g., web, e-mail and VoIP data) coexist on a single networking medium. The cost of connecting these kinds of systems together is decreasing dramatically. The audio and video production environments, however, tend to be structured according to the format of the audio and video content being conveyed. This results in costly infrastructure purchases and upgrades when new audio and video formats need to be supported [1].

Audio and video production tends to have high processing and bandwidth requirements. Until recently, these requirements were not met by off-the-shelf computing and networking equipment. Many of the current solutions are proprietary and consist of point-to-point connections, switching systems and processors that are centred on specific media formats. Video signals may be distributed using SD-SDI [2] or HD-SDI [3] interfaces using coaxial cables to connected devices together. Audio signals may flow through AES interfaces [4] and coaxial cables. Command and control messages require further interfaces and cables. These messaging protocols vary in format between different types of devices. Each of these different signal types are processed separately [1].

The successful transportation of real-time audio and video data is subject to strict Quality of Service (QoS) requirements. Without these requirements in place, this data is subject to loss, unacceptable latency, and lack of synchronisation.

This paper discusses the initial phase of an investigation into two networking technologies used to provide appropriate QoS for the transportation of real-time audio and video. These networking technologies are IEEE 1394 [5] and an enhanced form of Ethernet [6], called Ethernet AVB [7]. This investigation aims to provide interoperability between these different networking technologies and to provide a network-neutral method of control over the various parameters of these networks, and the devices that reside on them. This control is provided by an Internet Protocol (IP) based command and control protocol called AES-X170 [8].

Having audio and video devices interconnected via a common networking technology allows for more flexibility and is more future-proof. Current heterogeneous networks would move to a homogeneous network architecture. This kind of network will provide flexible routing capabilities as all of the signals that are transported by the network will be available throughout the network. This will allow the convergence on a single network of audio and video stream data, control data, as well as Voice over IP (VoIP) and general computer data [1].

II. IEEE 1394

IEEE 1394 is a peer-to-peer high-speed serial bus networking technology. Data is transmitted between various nodes on a network bus without the intervention of a host system. IEEE 1394 natively provides deterministic low-latency transmission of data. Each node that resides on the bus has its internal clock synchronised with the clock of a master device. IEEE 1394 provides asynchronous and

The authors would like to acknowledge the financial support of Telkom, Comverse, Stortech, Tellabs, Amatole Telecom Services, Mars Technologies, Bright Ideas 39 and THRIP through the Telkom Centre of Excellence in the Department of Computer Science at Rhodes University.

isochronous forms of packet transmission. The latest IEEE 1394 specifications allow for cable hops of 100m and more between devices. Individual IEEE 1394 networks can be bridged together to form larger more complex networks.

Isochronous packets are broadcast onto the network on one of 64 channels. Nodes on the network can be configured to receive isochronous packets on certain channels. Before a node is able to transmit isochronous packets, it requests its required network resources (bandwidth and a channel number) from an elected resource manager node. If any of the requested resources are unavailable, the node is unable to transmit isochronous packets. If, however, the requested resources are available, the node that made the request is guaranteed those resources. Each node that has successfully obtained its required resources is given an opportunity to transmit an isochronous packet every 125µs. This translates to a packet being transmitted 8000 times per second. This mechanism allows for the deterministic low-latency transmission of audio and video data across a network. For each 125µs cycle, once all of the nodes needing to send an isochronous packet have sent their packet, each node is able to transmit asynchronous packets.

Connection management requests can be communicated with asynchronous reads and writes to devices (via asynchronous packets). Devices on an IEEE 1394 network are able to read from, and write to, other devices registers. For example, a controlling device may write to another device in order to get it to start transmitting an isochronous stream on a particular channel. The same controlling device may then also write to yet another device to instruct it to start receiving that isochronous stream (on the configured channel).

Every 125µs cycle the internal clock in each device is synchronised with the clock of a cycle master node. This synchronisation of clocks allows for the synchronisation of multiple audio and video streams. It allows for packets to be time-stamped with a presentation time as they are transmitted onto the network.

III. ETHERNET AVB

Ethernet is the most dominant form of Local Area Network (LAN) technology on the market today. The Ethernet protocol is a simple protocol and has been widely adopted. Standard Ethernet and Ethernet bridging (switching) are not suitable for the transmission of real-time audio and video as Ethernet frames are subject to unpredictable frame loss and unacceptable latency variation. Ethernet is unable to provide network nodes with guaranteed network resources (such as bandwidth).

The requirement to be able to transport various kinds of real-time and non-real-time data over Ethernet (each with their own unique resource requirements) requires some enhancement to be made to standard Ethernet and Ethernet bridging. The 802.1 Audio/Video Bridging (AVB) Task Group [7] has been formed in order to create a set of standards that will allow for Ethernet networks to be built with appropriate QoS for real-time audio and video transmission. These standards will provide a number of enhancements to Ethernet and Ethernet bridging. Specifically, these standards specify:

- A protocol that allows various devices to communicate their network resource requirements (from a stream talker to a stream listener) for their audio and video streams. This protocol is known as the Multiple Stream Reservation Protocol (MSRP) [9].
- Network bridge queuing and forwarding rules to guarantee that data streams' resource requirements will be met [10].
- Precise clock synchronisation between devices [11].

A key requirement for the transmission of real-time audio and video streams is minimal packet loss. An Ethernet AVB system has to make use of Ethernet full-duplex links. The implication of this is that there is no collision of frames on the Ethernet medium.

Ethernet AVB frames are standard Ethernet frames that are VLAN tagged [12]. VLAN tagging allows for priority information to be carried within the frames, via the VLAN tag's Priority Code Point (PCP) field. These priority values are mapped into traffic classes.

A. MSRP

In order to successfully transmit audio and video data across a bridged network (with the required QoS), AVB devices need to ensure that there are sufficient network resources (such as bandwidth and buffer space) on the path from the stream talker to the stream listener.

A stream listener will register its intention to receive a particular data stream. It does so by sending out a specific frame onto the network (this frame will frequently include a multicast address). The intermediate bridges that receive this frame are able to set up various internal parameters to ensure that there are sufficient resources to transfer the requested stream frames to the listener device.

A stream talker device is able to register its intention to transmit a data stream with the network. It transmits an advertise frame onto the network. This frame includes details of the characteristics of the stream (e.g., bandwidth and traffic class). Each intermediate bridge that receives the frame is able to set up its internal parameters to ensure that there are sufficient resources to guarantee that the stream frames will be transmitted with the requested parameters. If any resource reservations fail, the devices are notified of this and streaming fails.

Once that a successful path has been established through a bridged network, the stream talker is able to transmit a stream.

Talkers are able to tear down streams and un-reserve resources that are reserved within the AVB devices. Listeners are able to stop receiving a stream by sending an appropriate notification to the network. The necessary resources are then released.

B. Queuing and forwarding rules

The priority information contained in AVB frames allows for specific bridge frame-forwarding characteristics to be applied to the frames and for traffic-shaping to happen at AVB end-points. End-point devices must transmit frames onto the network evenly. These transmission characteristics are defined by the traffic class that the stream is a part of, and the various QoS parameters that were used when the stream was approved by the network. For example, when a stream is initialised, it may be agreed that the stream talker will transmit frames at a rate of 4.7Mb/s with a frame rate of 8000 packets per second. These 8000 packets must then be transmitted evenly within each second (i.e., one packet transmission every 125µs).

Within an Ethernet bridge, audio and video frames will be forwarded as per usual. They will, however, also be subject to specific traffic manipulation rules. These rules are based on the specific traffic class that the frames belong to and the allocated bandwidth for the traffic class for the egress port. The frames that belong to a specific traffic class cannot go beyond the allocated resources.

An Ethernet AVB network also allows for the transmission of non-AVB traffic (such as e-mail traffic). This kind of traffic does not have any reserved QoS and frames belonging to the category of traffic may experience frame loss if network resources are constrained.

C. Clock synchronisation

AVB devices each contain a time-of-day clock, and these clocks are synchronised to the clock of the chosen grandmaster clock in the system. Each AVB source device is master-capable. This master is either dynamically chosen, or can be statically configured. This clock synchronisation allows multiple data streams to be synchronised, and provides a common time base for the sampling (at a stream talker) and presenting (at the stream listener) of stream data.

IV. AES-X170

The AES-X170 protocol is a User Datagram Protocol (UDP)/IP based peer-to-peer command and control protocol. Any device on a network that participates in the protocol may initiate and accept control, monitoring and connection management commands from any of its peers.

The essence of the protocol is its seven level addressing scheme. Multi-level addresses were created for a number of professional audio-related devices, and it was found that most parameters conform to a similar hierarchical structure. Each parameter in an X170 system is addressed using a fixed seven level address. These addresses intuitively reflect the hierarchical structure of the device being described. Each level in the hierarchy has a unique value assigned to it. Associated with each one of these values is an alias used to describe the level's functionality. This hierarchical addressing scheme is shown in Table 1. Also shown are the hierarchical address aliases that would be used to describe an equalisation parameter of an audio mixing desk.

Address level	Address level name	Example level alias
1	Section Block	Input block
2	Section Type	Analogue inputs
3	Channel Number	Analogue input 1
4	Parameter Block	Equaliser
5	Parameter Block Index	Equalisation band 1
6	Parameter Type	Equalisation parameter
7	Parameter Index	Equalisation parameter 1

Table 1: The X170 seven level addressing scheme

Address level values are defined within the AES-X170 specification. New devices describing existing parameter types (e.g., gain parameters) would use the same addresses used by existing devices. This allows new devices to be easily integrated into existing networks.

Parameters may be addressed individually using a full seven level address, or groups of parameters may be addressed by using wildcards at address levels. Disparate parameters may also be joined together into groups. This allows groups of parameters to be controlled via single control messages.

A device that participates in the X170 protocol builds up an internal seven level tree structure that reflects the actual device. Each of the leaf nodes of the tree is associated with a user supplied callback function that allows the device to respond to commands from its peers. Fig. 1 shows an example X170 stack with a built up tree structure. Each level in the tree structure has a unique address (and an address alias) used to identify it. The leaf nodes of the tree structure are associated with user supplied callback functions for a specific X170 application.

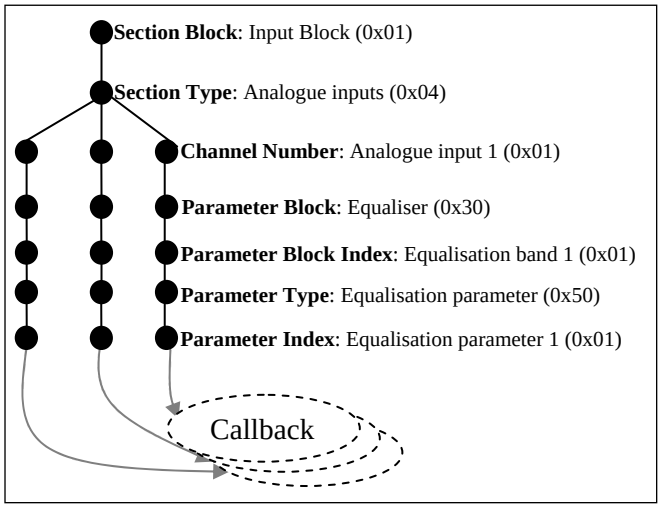

Fig. 1: An example hierarchical address for an equalisation parameter

The primary purpose of X170 messages is to allow various parameter values within devices to be obtained and set. Each X170 message contains a command type and a seven level address. For example, an X170 message may contain a GET VALUE command and an address for a gain parameter. When this gain parameter's callback function is called, it is responsible for returning its current value to the requesting device. Similarly, an X170 message may contain a SET VALUE command and an address for a gain parameter. The addressed gain parameter (via the callback function) then has to set its gain value appropriately.

Due to the fact that the X170 protocol is based on UDP/IP, it has been implemented independent of any networking architecture. This allows for this protocol to work across multiple networking architectures without any changes.

V. TUNNELLING ETHERNET TRAFFIC OVER IEEE 1394

IEEE 1394 natively provides an environment whereby audio and video data may be deterministically transmitted over a digital network with low latency. Ethernet and Ethernet bridging currently do not provide these mechanisms. The first phase of this investigation looked at a mechanism for transmitting Ethernet frames over an IEEE 1394 network to provide a deterministic low latency environment for this traffic.

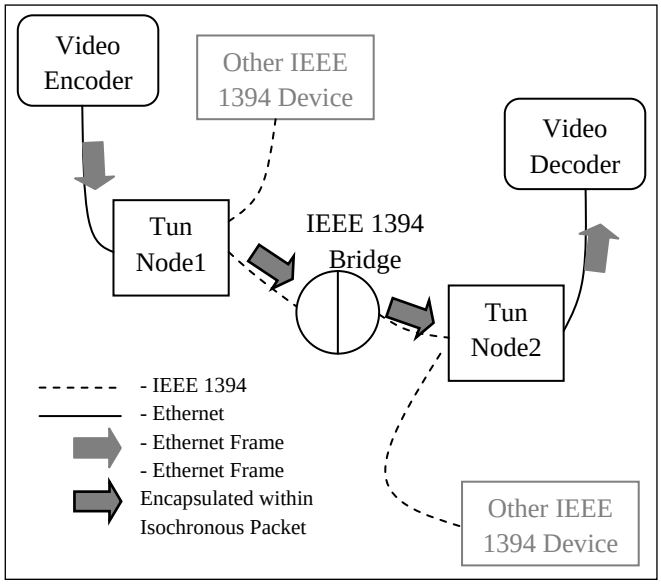

Fig. 2: Tunnelling of Ethernet traffic over IEEE 1394

Given in Fig. 2 is a sample set up of two IEEE 1394 Ethernet tunnelling devices (Tunnel Nodes). These devices are used as entry and exit points for tunnelling Ethernet frames over IEEE 1394 networks. These devices are used to capture all incoming Ethernet traffic that is not addressed to the node itself, and to forward this traffic to a second Tunnel Node. This captured Ethernet traffic is encapsulated within IEEE 1394 isochronous packets and is sent onto the network. The second Tunnel Node is responsible for receiving these isochronous packets and for extracting the Ethernet frames. These Ethernet frames are then transmitted out of its Ethernet interface.

The tunnelling capabilities provided by the system mentioned in Fig. 2 allow for the deterministic low-latency transmission and routing of Ethernet frames between Ethernet devices. Devices that connect (via Ethernet) to the Tunnel Nodes use full-duplex links. This has the implication that the transmitting device is the only device contending for access to the medium, and thus collision detection is not needed, so it is able to transmit frames at will.

The Tunnel Nodes allocate resources (bandwidth and an isochronous channel number) before they transmit any isochronous packets. This bandwidth allocation affects the maximum size that an isochronous packet may be. An IEEE 1394 device is able to transmit isochronous packets at a rate of 8000 packets per second. It could be possible that a Tunnel Node is receiving more than 8000 Ethernet frames per second. To counter this, if there is more than one Ethernet frame waiting to be transmitted onto the IEEE 1394 network, the Tunnel Node will pack multiple Ethernet frames into a single isochronous packet (if there is sufficient space left in the packet for this to happen). At the receiving Tunnel Node, these frames are unpacked and individually sent onto the Ethernet network.

If, on the other hand, an Ethernet frame waiting to be sent onto the IEEE 1394 network is larger than the maximum allowed isochronous packet payload size, the Tunnel Node will fragment the packet and send the fragments across to the receiving Tunnel Node. The receiving Tunnel Node will rebuild the Ethernet frame as it receives the fragments. Once it has a complete Ethernet frame, it will transmit it out of its Ethernet interface.

In the scenario presented in Fig. 2 a video encoder is directly attached to the Ethernet interface of the first Tunnel Node. Directly attached to the Ethernet interface of the second Tunnel Node is a video decoder. The video encoder is able to transmit its Ethernet frames to the first Tunnel Node in a timely fashion. These frames can then be deterministically routed through a complex network of IEEE 1394 devices with low latency. At the second Tunnel Node, the frames are directly transmitted out of the Ethernet interface towards the video decoder.

If the video encoder and decoder were connected together through a bridged Ethernet network, they will have to contend for network resources with other devices on the network. This could lead to unpredictable frame loss and unacceptable latency.

The Tunnel Node application has allowed existing Ethernet devices to take advantage of the characteristics of IEEE 1394 networks.

VI. Providing Control over the Tunnel Nodes

Within each Tunnel Node is an X170 stack. Each Tunnel Node builds up a tree structure representing the hierarchical structure of the device itself. This hierarchical structure allows the parameters of the devices to be addressed such that their values can be obtained and set.

Fig. 3 shows how the X170 stack is integrated with the tunnelling component of the Tunnel Node. The tunnelling component of the Tunnel Node has parameters that allow the transmission and reception isochronous channels, and the transmission frame sizes to be obtained and set, for example. The obtaining and setting of these parameters is handled with the user-defined callbacks within the tunnelling component of the Tunnel Nodes.

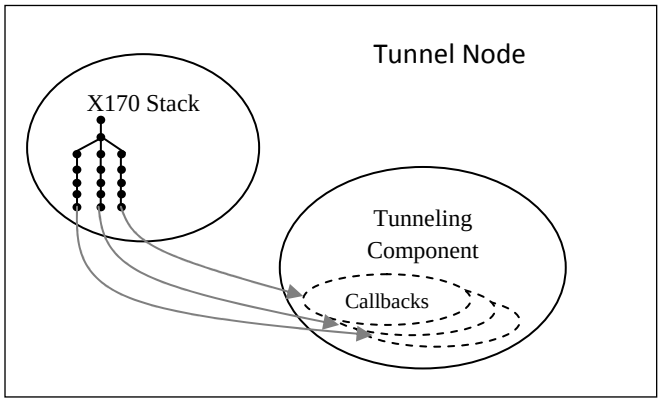

Fig. 3: A Tunnel Node with an X170 stack

Fig. 4 shows an example of the tree structure that is built up for two of the Tunnel Node's parameters. These parameters are a transmitting IEEE 1394 isochronous stream (multicore) channel number and its maximum packet payload size.

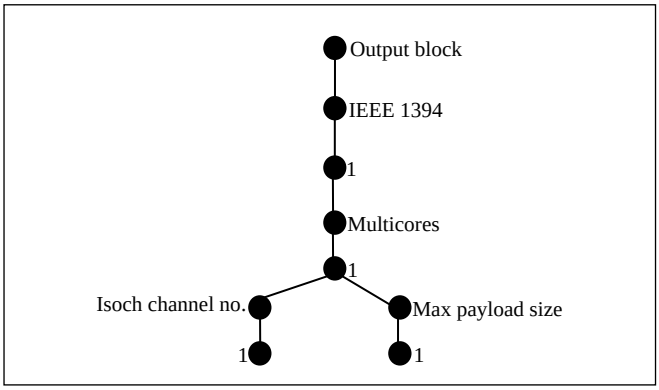

Fig. 4: An example X170 tree structure for two parameters of a Tunnel Node

A desktop application (Connection Manager) has been developed that allows for the Tunnel Nodes (and its parameters) to be remotely represented and adjusted. The Connection Manager application uses an X170 stack in order to communicate with the X170 stack of the Tunnel Nodes.

When the Connection Manager application starts up, it discovers all of the X170 devices that exist on the network. The main display of the application represents the discovered devices along the axes of a grid. On this grid, the devices along the left are viewed as source devices. The devices along the top of the grid are viewed as destination devices. The buttons between the devices allow for connections between the various devices to be made. The main display of the Connection Manager is shown in Fig. 5. This figure is displaying two Tunnel Nodes. A typical network, however, will consist of many different types of devices.

Fig. 5: The Connection Manager's main interface

Selecting one of the cross points on the grid will display the various multicores associated with the devices, in the form of a grid (as shown in Fig. 6). The multicore along the left hand side of the grid is viewed as a source multicore. This multicore is the originator of isochronous packets. The multicore along the top of the grid is viewed as a destination multicore. This multicore is the consumer of isochronous packets. Fig. 6 is displaying the output multicore of the selected source Tunnel Node, and the input multicore of the selected destination Tunnel Node. Typically, a device has a much larger number of multicores (allowing a single device to stream multiple streams of data). Hence, there is a desirability to display these multicores in the form of a grid.

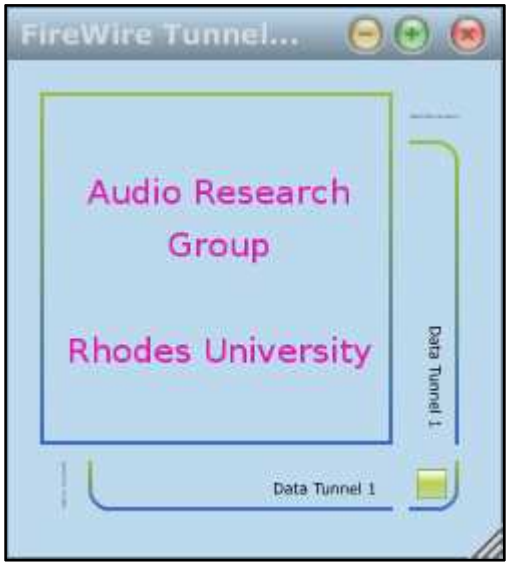

Fig. 6: The Connection Manager's multicores display

With this grid display, it is possible to ensure that the source Tunnel Node's output multicore streams its isochronous packets to the destination Tunnel Node's input multicore. When the grid button (shown in Fig. 6) is selected, the X170 stack of the Connection Manager sends X170 messages to the devices that are involved in the data streaming. These messages instruct the Tunnel Nodes to set up their internal parameters to allow for the streaming to take place.

Selecting the source multicore in Fig. 6 displays the various parameters associated with it. Shown in Fig. 7 is the window that displays these parameters. These are the parameters that are depicted in Fig. 4. Any adjustments to these values are communicated to the relevant Tunnel Nodes via X170 messages.

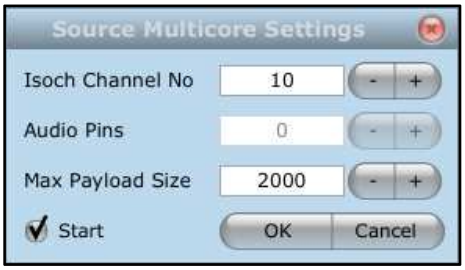

Fig. 7: The Connection Manager's source multicore settings window

VII. Towards Network Interoperability and Control

The development of the Tunnel Node application and the control thereof by the X170 protocol has laid the ground work for the interoperability of, and control over, IEEE 1394 and Ethernet AVB audio and video networks. Further development of the Tunnel Node software is required such that it is able to understand the AVB suite of protocols. Ethernet bridges need to be implemented such that they are

able to understand the MSRP protocol and to be able to honour the requests of this protocol. An X170 hierarchy needs to be developed such that it is able to accurately represent, and allow for control over Ethernet AVB devices. This control requires an interaction between an MSRP module and X170 stack.

Audio and video devices that exist on each of these networks need to be able to communicate with each other. Audo and video gateway devices need to be developed that understand the protocols of each network such that devices on one network are able to communicate with devices on the other network.

VIII. CONCLUSION

There is a trend towards a convergence of different forms of content to single networking technologies. Different types of content have different network requirements. IEEE 1394 and Ethernet AVB are two networking technologies that allow for both real-time and non-real-time content to be transferred on a single network. Different audio and video devices tend to have their own command and control protocols.

In this paper we described a system that allows for the transportation of Ethernet traffic between Ethernet devices in a deterministic low latency environment. This was achieved by transporting this traffic over IEEE 1394 networks. An IP based command and control protocol has been used to control the parameters of this system. This system has laid the groundwork for developing a network-neutral command and control protocol. It is envisioned that this protocol will seemingly be integrated into devices of a range of different networking technologies. This should allow existing devices to communicate with new devices without any changes.

REFERENCES

[1] **Teener, Michael J and Gaél, Macé.** Ethernet in the HD studio. *Broadcast Engineering.* [Online] May 1, 2008. [Cited: April 28, 2009.] http://broadcastengineering.com/hdtv/ethernet-hd-studio/.

[2] **Editors of SMPTE 259M-2008.** *Television – SDTV Digital Signal/Data – Serial Digital Interface.* New York : Society of Motion Picture and Television Engineers, 2008. SMPTE 259M-2008.

[3] **Editors of SMPTE 292M.** *1.5 Gb/s Signal / Data Serial Interface.* New York : Society of Motion Picture and Television Engineers, 2008. SMPTE 292M.

[4] **Editors of AES3-2003.** *AES Standard for Digital Audio Engineering - Serial Transmission Format for Two-channel Linearly Represented Digital Audio Data.* New York : Audio Engineering Sociery, 2003. AES3-2003.

[5] **Anderson, Don.** *FireWire System Architecture.* Massachusetts : MindShare Inc, 1999.

[6] IEEE 802.3 Working Group. *IEEE 802.3 Working Group.* [Online] March 13, 2009. [Cited: April 28, 2009.] http://www.ieee802.org/3/.

[7] The Audio/Video Bridging Task Group. *IEEE 802.1.* [Online] September 18, 2008. [Cited: April 28, 2009.] http://www.ieee802.org/1/pages/avbridges.html.

[8] **Editors of AES-X170.** *DRAFT AES Standard for Audio Applications of Networks - Integrated Control, Monitoring, and Connection Management for Digital Audio and Other Media Networks.* New York : AES, 2009. AES-X170.

[9] **Editors of IEEE 801.1Qat.** *Stream Reservation Protocol.* New York : Institute of Electrical and Electronics Engineers, 2009. 802.1Qat Draft 2.1.

[10] **Editors of IEEE 802.1Qav.** *Forwarding and Queuing Enhancements for Time-Sensitive Streams.* New York : Institute of Electrical and Electronics Engineers, 2009. 802.1Qav Draft 4.0.

[11] **Editors of IEEE 802.1AS.** *Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks.* New York : Institute of Electrical and Electronics Engineers, 2009. 802.1AS Draft 5.0.

[12] **Editors of IEEE 802.1Q.** *Virtual Bridged Local Area Networks.* New York : Institute of Electrical and Electronics Engineers, 2005. 802.1Q.

Philip Foulkes has an MSc in Computer Science from Rhodes University. This MSc was obtained whilst under the supervision of Professor Richard Foss. Philip is now reading towards his PhD in Computer Science. Philip's main research interests are the use of digital networks for the transportation of real-time media streams.