A PROPOSED REVISION TO IRIG 218 BASED ON REAL WORLD EXPERIENCE

Gary A. Thom
GDP Space Systems
300 Welsh Road, Horsham, PA 19044
gthom@delta-info.com

Abstract

The Range Commanders Council has been attempting to standardize Telemetry over IP (TMoIP) for many years now. While the attempt has been valiant, the outcome to date has not been very successful. As a result, many vendors have implemented their own proprietary methods for sending PCM data over IP networks resulting in a lack of interoperability. As telemetry ground stations are finally making the move toward network centric architectures, it is worth considering the lessons learned over the previous 10 years of designing, installing, troubleshooting and optimizing telemetry data distribution over IP networks. This paper describes a proposed revision to IRIG 218 based on these real life experiences. It discusses the critical decisions and architectural decisions to be made and some of the pitfalls to be avoid.

Key Words: IRIG 218, TMoIP, IP, TCP, UDP, network, PCM.

1 Introduction

The motivation for moving to TMoIP was twofold: first, to find cost effective PCM data distribution and second, to provide reliable and robust PCM data distribution regardless of the destination. The global explosion of IP networking has provided a built in infrastructure with access to the most remote destinations. A wide variety of transport mechanisms for IP traffic provides ubiquitous connectivity, whether twisted pair, fiber optic cable, microwave links, satellite links, analog modems and cell phones, IP connectivity is everywhere.

This ubiquity and global deployment has driven down the cost of networking components such as routers and switches. It provided dynamic routing and redundant paths, improving reliability and fault tolerance. The insatiable appetite for more data has resulted in ever increasing bandwidth availability.

The result is a reliable, cost effective infrastructure for PCM data distribution, whether on private IP networks or globally via the public internet.

2 Network Basics

Much of the support needed by TMoIP is provided by the various layers of the network protocol stack. This section discusses these layers and the services that they provide.

2.1 Physical Layer

The Physical Layer consists of the hardware networking interface. This layer provides for the transmission of bits over some physical media. Media types include, but are not limited to optical fiber, coax cable or twisted pair cable. The specification of the Physical Layer includes features such as connector type, electrical signal levels, modulation, frequency, data rate and line coding.

The primary functionality that the Physical Layer provides is the efficient delivery of bits over a selected medium from one point to another.

2.2 Data Link Layer

The Data Link Layer provides for the transmission of data between nodes on the same network segment. For TMoIP, the Ethernet [5] protocol provides this layer functionality. There are two important aspects of this layer. The first is addressing. The Ethernet protocol provides for the use of the Media Access Control (MAC) address. A MAC address is defined for each node on the network segment and Ethernet packets are sent from one node to another based on the MAC address.

The second is the Ethernet packet. The Ethernet packet shown in Figure 1, consists of a header, a payload and a trailer.

<table>
<tr><td rowspan="3">Preamble</td><td rowspan="3">Start of frame delimiter</td><td>MAC destination</td><td>MAC source</td><td>Optional VLAN Tag</td><td>Ethertype or length</td><td rowspan="2">Payload Data</td><td>Frame check sequence</td><td rowspan="3">Inter-packet gap</td></tr>
<tr><td colspan="5">Packet Header</td><td>Trailer</td></tr>
<tr><td colspan="7">Ethernet Frame</td></tr>
<tr><td>7 bytes</td><td>1 bytes</td><td>6 bytes</td><td>6 bytes</td><td>4 bytes</td><td>2 bytes</td><td>46–1500 bytes</td><td>4 bytes</td><td>12 bytes</td></tr>
</table>

Figure 1 - Ethernet Packet Format

The Ethernet Frame has a minimum size of 64 bytes. For that reason, it has a minimum payload size of 46 bytes after accounting for the mandatory header bytes. If less data is to be sent, it will be padded out with zeros to reach the 64 byte minimum. The maximum size of the Ethernet Payload is 1500 bytes. Since the Length field is 2 bytes, payload lengths greater that 1500 bytes can be generated. These are called Jumbo Frames. Jumbo Frames are not universally supported, so their use may hamper interoperability and therefore should not be used in TMoIP systems.

A key feature of the Ethernet Frame is the Frame Check Sequence. This error detection mechanism will identify errors in data transmission over the link. Typically, the Data Link Layer will drop packets that fail the FCS check and will not pass the resulting payload up to the next layer in the protocol stack.

2.3 Internet Layer

The Internet Layer provides connectivity between network segments. To do this, the Internet Protocol (IP) is used to provide a higher level address, the IP Address, which is routable between network segments. This is what allows the routing of data across the global internet.

Today there are two IP protocols: IPv4 [6] and IPv6 [7]. IPv4 was the original internet protocol. It provided for a 32-bit address. The address is divided between a network address and a subnet address. The size of these address components is determined by the subnet mask. This addressing information is carried in the header of the IPv4 packet as shown in Figure 2.

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>09</td><td>08</td><td>07</td><td>06</td><td>05</td><td>04</td><td>03</td><td>02</td><td>01</td><td>00</td><td>W</td></tr>
<tr><td colspan="4">Version</td><td colspan="4">Hdr Len</td><td colspan="8">Type of Service</td><td colspan="16">Total Length</td><td>1</td></tr>
<tr><td colspan="16">Identification</td><td colspan="3">Flags</td><td colspan="13">Fragment Offset</td><td>2</td></tr>
</table>

Time to Live	Protocol	Header Checksum	3
Source Address			4
Destination Address			5
Payload Data (0 to 1480 bytes)			

Figure 2- IPv4 Packet Format

Besides the addresses, there are a few other useful fields in the IPv4 header. The Total Length field specifies the size of the IP header plus the payload. This is very useful when the IP packet is smaller than 46 bytes because it tells how much of the Ethernet frame payload is data and not fill. It is also useful for packets which are larger than a single Ethernet frame and used with the Fragment Offset field allow large payloads to be fragmented into multiple Ethernet frames. The Protocol field indicates the contents of the packet Payload. The Header Checksum only protects the header information. It does not protect the data. If the checksum check fails, the packet is discarded. Since the FCS is also being checked at the Ethernet level, this check will only catch the rare errors that get through that check or some implementation error at the IP layer.

IPv6 was developed primarily because the world was running out of IPv4 addresses. IPv6 expands the IP address to 128 bits. There is a similar split between network address and subnet address, the sizes of which are determined by the IP Address Prefix. The IPv6 packet is shown in Figure 3.

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>09</td><td>08</td><td>07</td><td>06</td><td>05</td><td>04</td><td>03</td><td>02</td><td>01</td><td>00</td><td>W</td></tr>
<tr><td colspan="4">Version</td><td colspan="8">Traffic Class</td><td colspan="20">Flow Label</td><td>1</td></tr>
<tr><td colspan="16">Payload Length</td><td colspan="8">Next Header</td><td colspan="8">Hop Limit</td><td>2</td></tr>
<tr><td colspan="32" rowspan="4">Source Address</td><td>3</td></tr>
<tr><td>4</td></tr>
<tr><td>5</td></tr>
<tr><td>6</td></tr>
<tr><td colspan="32" rowspan="4">Destination Address</td><td>7</td></tr>
<tr><td>8</td></tr>
<tr><td>9</td></tr>
<tr><td>10</td></tr>
<tr><td colspan="32">Payload Data (0 to 1460 bytes)</td><td></td></tr>
</table>

Figure 3- IPv6 Packet Format

In addition to the address change, IPv6 removed the unnecessary Header checksum and some other fields.

2.4 Transport Layer

The Transport Layer provides the data delivery services. There are two common data delivery services used over IP. These are User Datagram Protocol (UDP) [9] and Transport Control Protocol (TCP) [8].

UDP provides a connectionless, best-effort datagram delivery service. There is no connection established. Packets are just sent to the destination address. There is no packet loss detection, no packet acknowledgement or retransmission. One benefit of this service is low latency transmission. Buffering for acknowledgement and retransmission is not required and buffering increases latency. Another benefit is

datagram delivery. That is the payload of the packet is considered an atomic message which is delivered intact. UDP has a packet format shown in Figure 4.

30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	W
Source Port Number															Destination Port Number																1
Packet Length															Checksum																2
Payload Data (0 to 1476 bytes)																															

Figure 4- UDP Packet Format

The UDP header provides length information for the entire UDP packet and a checksum that protects the header and the data. The checksum is optional for IPv4 and mandatory for IPv6. The port numbers are used to route the payload to a specific application at the Application Layer.

Typical UDP transmission is unicast where one source sends packets to one destination. The connectionless nature of UDP also allow for multicast transmission of packets. In this case, packets sent to special multicast IP addresses can be distributed to multiple destinations which join the multicast group. This provides not only an efficient delivery mechanism, but also provides a receiver directed addressing model where the receiving device determines what packet stream to receive.

TCP provides a connection oriented, guaranteed delivery byte-stream protocol. First a connection is established between the source and destination device using a handshaking protocol. After the connection is established a predetermined number of packets are sent before waiting for an acknowledgement. As acknowledgements are received, more packets are sent. If an acknowledgement is not received, the packet is retransmitted. The packets contain a byte stream and not atomic packets. That is, the data that is sent may not arrive in a packet of the same size as the original data. TCP fragments the packets as needed to support the acknowledgement and retransmission process. This requires the data to be parsed to find the start of a payload header which must have some unique synchronization information. The TCP Packet is shown in Figure 5.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	W
Source Port Number																Destination Port Number																1
Sequence Number																																2
Acknowledgement Number																																3
Offset				Flags												Window Size																4
Checksum																Urgent Pointer																5
Payload Data (0 to 1464 bytes)																																

Figure 5 - TCP Packet Format

TCP uses port number as described above for UDP. Sequence Number, Acknowledgement Number, Window Size are used for acknowledgement and retransmission.

In addition to higher latency and packet parsing requirements, another drawback of TCP is that it requires bi-directional communications making it unusable for some applications that do not provide a return path.

The need for a checksum at this layer is questionable. Studies have shown that some errors get through the Ethernet FCS check and may be caught at this layer. In addition, it has been reported that errors are

introduced in intermediate network devices. In any case, packets which fail the checksum check at this layer are also discarded and ultimately retransmitted.

2.5 Application Layer

TMoIP is our Application Layer. It needs to provide those functions that are not provided by the other layers of the protocol but which are required by the application.

The Range Commanders Council (RCC) has developed the IRIG 218 standard [1], the latest version having been published in 2010. This is a very efficient TMoIP header consisting of only four bytes as shown in Figure 6**Error! Reference source not found.**. It is derived from the Pseudo-Wire protocol [2][3][4] defined by the IETF.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Res				L	R	M		Res		Length						Sequence Number															
Payload Data (0 to 1472 bytes)																															

Figure 6- IRIG 218-10 Packet Format

The IRIG 218 header contains six bits that are reserved (Res) and set to zeros. It contains Four bits (L R, M) that carry alarm information. The alarm bits are a carryover from circuit switched telephony over IP defined in the Pseudo-Wire standard. These bits are remapped from forward and reverse alarm indication to a local and remote error indication. These indications are poorly defined in IRIG 218 and meaningless for TMoIP.

The Length field is used only when the payload is less than 64 bytes. If the Length field is zero, then the payload length is determined from lower level protocols. This may have been added because of the minimum payload size requirement for Ethernet, which is 64 bytes. However, the IP and UDP layers provide correct packet size information even when fill is inserted to meet the minimum Ethernet payload size requirement. This makes this field unnecessary.

So the only real useful information in the IRIG-218-10 header is the Sequence Number. This field allows the receiving device to determine if packets are lost or if they arrive out of order.

The IRIG 218-10 packet format does not contain any information that would be useful in reconstructing the PCM output timing at the receiving device. Lacking any other timing information, an Ethernet-to-PCM device will need to measure the arrival time difference between two packets and knowing the number of bits in the packet, it can calculate a data rate. However, there could be significant jitter in the packet arrival times, requiring several measurement to be averaged in order to remove the effects of the jitter. This will result in either a delayed startup while waiting for sufficient packets to average or a drifting output clock rate as the calculation is improved.

The standard makes a reference to using the Real Time Protocol (RTP) [10] to provide clock recovery support for TMoIP. The required RTP header fields are shown in Figure 7.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	W
Ver		P	X	CSRC				M	Payload Type							Sequence Number																1
Timestamp																																2
SSCR Identifier																																3

IRIG 218 Packet	

Figure 7 - RTP Packet Format

While RTP does provide a 4-byte Timestamp which could be used to calculate the data rate, none of the other information in the RTP header is relevant. There is 2-byte sequence number which is redundant with the IRIG 218 header. The remaining fields: Payload Type, CSRC Count, X and P Flags and SSRC Identifier provide no additional value and are undefined in IRIG 218 for use in TMoIP.

The format and resolution of the one useful field in the RTP header, the Timestamp, is undefined in IRIG 218. This will lead to interoperability problems as different vendors use different time bases and resolutions.

The use of RTP seems to be a waste of space for the value received and an opportunity for confusion. For this reason, many vendors of TMoIP products have developed their own TMoIP packet formats which they feel better serve the application.

3 What do we want to use TMoIP for?

There are two main applications for TMoIP. The first is for PCM to PCM distribution over IP networks. In this application, the goal is to replace more traditional distribution systems such as coax cable, fiber optic cable, microwave links, matrix switches, patch panels, etc. with packet based transmission over IP networks. This is the application that the current IRIG 218-10 addresses. The application uses two TMoIP devices: one to convert PCM data to Ethernet packets and the second to convert the Ethernet packets back to PCM as shown in Figure 8.

Figure 8 - PCM to PCM Data Distribution

The second application is for PCM to Computer distribution. In this application, the goal is to provide a convenient interface to get PCM data into a computer for software decommutation and/or recording. The current IRIG-218 does not address this application. In addition to converting serial data to IP packets, this application requires frame/subframe synchronization and frame alignment of synchronized data to the packet boundaries. This application only requires a single TMoIP device as shown in Figure 9.

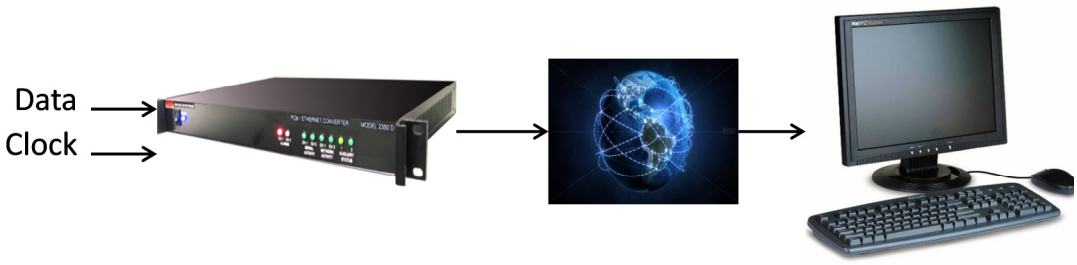

Figure 9 – PCM-to-Computer Data Distribution

4 What is needed for a TMoIP Protocol?

In order to answer this question, we need to divide the problem it several parts. For the first part, let's assume that we will be using UDP because of the connectionless, low-latency characteristics of the transmission. Later we will look at TCP.

4.1 UDP

As we have seen above, the network layers up through UDP provide addressing, routing, errored packet removal, payload size information and packet alignment.

4.1.1 Packet Data Alignment

Since UDP datagrams are atomic, the data sent by the source device in a UDP packet will be the same as the data received at the destination, assuming MTU sizes are not violated. That is, the packet size and location of data within the payload will be unchanged. To take advantage of this feature, we need to require that the first byte of the TMoIP header is the first byte of the UDP payload.

In the general case, the PCM data in the TMoIP packet is unaligned. However, as will be discussed later, there are cases where alignment may be helpful.

4.1.2 Packet Size

It is not necessary for the TMoIP Layer to explicitly indicate the packet size because this information can be derived from the UDP layer. Adjusting the packet size provides one means of controlling the latency and adapting to the data rate of the TMoIP stream. Packets from one byte to the maximum UDP payload size should be supported.

4.1.3 Packet Loss Detection

Since errored packets are discarded by the lower protocol layers and since UDP does not provide packet acknowledgement and retransmission, we must detect missing packets. This can be accomplished with a Sequence Number which increments for every packet transmitted. The 16-bit Sequence Number allows for detection of up to 65534 missing packets.

Missing packets need to be accounted for in the output PCM stream by inserting a "dummy" packet. If they are not, then there will be a dip in the output data rate or a discontinuity in the output clock. It may be desirable to fill the "dummy" packets with alternating ones and zeros in order to keep up the transition density, so as to not cause problems for downstream bit synchronizers which do not like long runs without transitions.

Sequence numbers will also allow the detection of packets that arrive out of order due to routing differences. Detecting an out-of-order packet and doing something about it are two different things. Correcting an out-of-order packet can only be done if the time to use that packet has not passed. Since the UDP layer is not providing any buffering, any buffering required to correct out-of-order packets would need to be handled by the TMoIP application. This will increase latency. In low latency applications, out-of-order packets should be discarded.

4.1.4 Data Rate Recovery

It is desirable to have a quick way to determine the PCM data rate in order to set the output PCM clock at the destination device. It can be argued that no additional information is required. You know how often

you are receiving packets and you know how much data is in each packet so you can calculate the data rate. However, due to network jitter, the arrival time of packets may not be consistent, so many packets will need to be received and their arrival times averaged in order to get a good approximation of the data rate. While this is true, it will cause either a delay in the start of the PCM output or it will result in a change in the PCM output clock as the data rate calculation accuracy improves.

There are two alternate approaches. The first approach measures the data rate of the PCM input at the source device and sends that information in the TMoIP header. This method provides the fastest startup time and good stability. Fine control of the output clock must still be performed by a control loop in order to match the input and output clocks to avoid buffer overflow or underflow.

The second method timestamps the first bit of the PCM data in each packet. The difference in timestamps of consecutive packets along with the payload size can be used to calculate the data rate. While this has the downside of requiring two packets before starting the PCM output it removes any network jitter from the calculation and has some additional benefits that will be discussed later.

In either case, it is necessary to completely specify the range, accuracy and resolution of the measurement.

4.2 TCP

Like UDP, the layers up through TCP also provide addressing, routing, errored packet removal and payload size information. However, they do not provide packet alignment.

TCP does provide guaranteed delivery of packets, so we do not need to worry about lost packet detection. This eliminates the need for a Sequence Number in the header.

Since TCP packets may be segmented by the protocol stack and intermediate network devices, the received TCP packet may not contain a complete TMoIP payload. Assuming that the first byte sent over a new TCP connection is the first byte of a TMoIP header, we only need to know the TMoIP payload size in order to find the next header.

5 The Proposed TMoIP Protocol

In order to accommodate the functionality described in the previous section, the following TMoIP packet header and rules are proposed. The recommended TMoIP Header is shown in Figure 10.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	W
Version				Reserved				Flags								Sequence Number / Payload Size																1
Timestamp[Course]																																2
RES	Timestamp[Fine]																															3
Payload Data (0 to																																

Figure 10 - Proposed TMoIP Header

This header contains the following fields:

- A 4-bit Version Number which indicates the format of the packet. Since the original IRIG 218 header had Reserved bits in this position which should always be set to '0000', we can use this field as a Version number. This new header format would use '0001'.
- A 4-bit field Reserved for future use. Shall be set to '0000'
- The 8-bit Flag field provides information about the header or the data.
    - Bit 16: Timestamp Present – If set to one, the Timestamp fields would be present. If set to zero, the Timestamp would be dropped making the packet very similar to the current IRIG 218-10 packet header.
    - Bit 17: Test Data – If set to one, payload contains 2^15-1 PN Pattern for automatic BER testing. If set to zero, payload contains PCM data.
    - Bit 19-18: Frame Alignment – 00 = no frame alignment, 01 = Frame aligned, first packet (Bits 23-20 contains frame/sub-frame lock status), 10 = Frame aligned, continuation packets (Bits 23-20 not used), 11 = Reserved.
    - Bits 21-20: Minor Frame Sync Status – 00 = Search, 01 = Check, 10 = Lock, 11 = Flywheel.
    - Bits 23-22: Major Frame Sync Status – 00 = Search, 01 = Check, 10 = Lock, 11 = Flywheel.
- The Sequence Number / Payload Size field is dual use. When UDP is used, this field contains a 16-bit Sequence Number which increments for every packet and is used to detect missing or out of order packets. When TCP is used, this field contains an 16-bit Payload Size which indicates the number of TMoIP data bytes following the header.
- The Timestamp field is made up of three sub-fields. The full Timestamp marks the time that the first data bit of the packet payload is received at the ingest device relative to the selected time system (i.e. IRIG B, NTP, PTP). The Timestamp consists of the following sub-fields:
    - 32-bit Course Timestamp. This represents Seconds since 12:00 AM 1/1/1970 which is the same as the Posix epoch and can easily be derived from IRIG B, GPS, NTP or PTP.
    - 30-bit Fine Timestamp. This represents fractions of the second and is usually derived by phase locking a local oscillator to the time source.
    - 2-bit Fine Timestamp Resolution indicator. This field indicates the resolution of the Fine Time field as follows:
        - 00 – Milliseconds (0 – 999)
        - 01 – Microseconds (0 – 999,999)
        - 10 – Nanoseconds (0 – 999,999,999)
        - 11 – Reserved

The following rules apply:

- The first byte of the UDP Payload shall be the first byte of the TMoIP Header.
- The UDP packet TMoIP Header shall contain a Sequence Number. The Sequence Number shall increment by one for every packet transmitted and shall roll over from all ones to all zeros.
- The first byte of the TCP Payload after the connection is established shall be the first byte of the TMoIP header.
- The TCP packet TMoIP Header shall contain a Payload Size. The Payload Size shall indicate the number of bytes in the TMoIP Payload (the number of bytes until the next TMoIP Header).
- An Ingress device shall insert a Timestamp in every packet. It shall indicate the resolution of the fine timestamp based on the resolution that the device can support.
- In a frame aligned packet, the first byte of the frame shall be the first byte of the TMoIP Payload and the Frame Alignment flags shall be set to '01'. Additional frame bytes shall continue contiguously in subsequent TMoIP payloads with the Frame Alignment flags set to '10'. The Timestamp shall not be present in these packets.

6 Conclusion

This proposed TMoIP header provides all of the information necessary to transmit PCM data over an IP network. It provides packet loss detection using the Sequence Number field. It provides data rate determination using the Timestamp. It provides data alignment for TCP using the Payload Size field. It supports both PCM-to-PCM distribution applications as well as frame aligned PCM-to-Computer applications. It is efficient adding only 3 32-bit words to the packet overhead. It provides for future expansion by making use of the Version and Reserved fields. The payload of the TMoIP packet can be any serial stream of PCM data. The proposed protocol is independent and unaware of the content.

7 References

[1]	IRIG STANDARD 218-10	Telemetry Transmission Over Internet Protocol (TMoIP) Standard.
[2]	IETF RFC 3985	Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture.
[3]	IETF RFC 3916	Requirements for Pseudo-Wire Emulation Edge-to-Edge (PWE3).
[4]	IETF RFC 4553	Structure Agnostic Time Division Multiplexing (TDM) over Packet (SAToP).
[5]	IEEE 802.3	IEEE Standard for Ethernet.
[6]	IETF RFC 791	Internet Protocol.
[7]	IETF RFC 2460	Internet Protocol, Version 6 (IPv6) Specification
[8]	IETF RFC 793	Transmission Control Protocol.
[9]	IETF RFC 768	User Datagram Protocol.
[10]	IETF RFC 3550	RTP: A Transport Protocol for Real-Time Applications