IPv4 over IEEE 1394

STATUS OF THIS DOCUMENT

ABSTRACT

This document specifies how to use IEEE Std 1394-1995, Standard for a High Performance Serial Bus (and its supplements), for the transport of Internet Protocol Version 4 (IPv4) datagrams. It defines the necessary methods, data structures and codes for that purpose and additionally defines a method for Address Resolution Protocol (ARP).

TABLE OF CONTENTS

1. INTRODUCTION

This document specifies how to use IEEE Std 1394-1995, Standard for a High Performance Serial Bus (and its supplements), for the transport of Internet Protocol Version 4 (IPv4) datagrams. It defines the necessary methods, data structures and codes for that purpose and additionally defines a method for Address Resolution Protocol (ARP).

The group of IEEE standards and supplements, draft or approved, related to IEEE Std 1394-1995 is hereafter referred to either as 1394 or as Serial Bus.

1394 is an interconnect (bus) that conforms to the CSR architecture, ISO/IEC 13213:1994. Serial Bus permits communications between nodes over shared physical media at speeds that range, at present, from 100 to 400 Mbps. Both consumer electronic applications (such as digital VCRs, stereo systems, televisions and camcorders) and traditional desktop computer applications (*e.g.*, mass storage, printers and tapes), have adopted 1394. Serial Bus is unique in its relevance to both consumer electronic and computer domains and is EXPECTED to form the basis of a home or small office network that combines both types of devices.

The CSR architecture describes a memory-mapped address space that Serial Bus implements as a 64-bit fixed addressing scheme. Within the address space, ten bits are allocated for bus ID (up to a maximum of 1,023 buses), six are allocated for node physical ID (up to 63 per bus) while the remaining 48 bits (offset) describe a per node address space of 256 terabytes. The CSR architecture, by convention, splits a node’s address space into two regions with different behavioral characteristics. The lower portion, up to but NOT including 0xFFFF F000 0000, is EXPECTED to behave as memory in response to read and write transactions. The upper portion is more like a traditional IO space: read and write transactions in this area usually have side effects. Control and status registers (CSRs) that have FIFO behavior customarily is implemented in this region.

Within the 64-bit address, the 16-bit node ID (bus ID and physical ID) is analogous to a network hardware address---but 1394 node IDs are variable and subject to reassignment each time one or more nodes are added to or removed from the bus.

The 1394 link layer provides a packet delivery service with both confirmed (acknowledged) and unconfirmed packets. Two levels of service are available: "asynchronous" packets are sent on a best-effort basis while "isochronous" packets are guaranteed to be delivered with bounded latency. Confirmed packets are always asynchronous but unconfirmed packets MAY be either asynchronous or isochronous. Data payloads vary with implementations and MAY range from one octet up to a maximum determined by the transmission speed (at 100 Mbps, named S100, the maximum asynchronous data payload is 512 octets while at S400 it is 2048 octets).

NOTE: Extensions underway in IEEE P1394b contemplate additional speeds of 800, 1600 and 3200 Mbps.

2. DEFINITIONS AND NOTATION

2.1 Conformance

When used in this document, the keywords "MAY", "OPTIONAL", "RECOMMENDED", "REQUIRED", "SHALL" and "SHOULD" differentiate levels of requirements and optionality and are to be interpreted as described in RFC 2119.

Several additional keywords are employed, as follows:

EXPECTED: A keyword used to describe the behavior of the hardware or software in the design models assumed by this standard. Other hardware and software design models MAY also be implemented.

IGNORED: A keyword that describes bits, octets, quadlets or fields whose values are NOT checked by the recipient.

RESERVED: A keyword used to describe objects---bits, octets, quadlets and fields---or the code values assigned to these objects in cases where either the object or the code value is set aside for future standardization. Usage and interpretation MAY be specified by future extensions to this or other standards. A RESERVED object SHALL be zeroed or, upon development of a future standard, set to a value specified by such a standard. The recipient of a RESERVED object SHALL NOT check its value. The recipient of an object defined by this standard as other than RESERVED SHALL check its value and reject RESERVED code values.

2.2 Glossary

The following terms are used in this standard:

address resolution protocol: A method for a requester to determine the hardware (1394) address of an IP node from the IP address of the node.

bus ID: A 10-bit number that uniquely identifies a particular bus within a group of multiple interconnected buses. The bus ID is the most significant portion of a node’s 16-bit node ID. The value 0x3FF designates the local bus; a node SHALL respond to requests addressed to its 6-bit physical ID if the bus ID in the request is either 0x3FF or the bus ID explicitly assigned to the node.

encapsulation header: A structure that precedes all IP data transmitted over 1394. See also link fragment.

IP datagram: An Internet message that conforms to the format specified by RFC 791.

link fragment: A portion of an IP datagram transmitted within a single 1394 packet. The data payload of the 1394 packet contains both an encapsulation header and its associated link fragment. It is possible to transmit datagrams without link fragmentation.

multicast channel owner: A multicast source that has allocated a channel for one or more multicast addresses and transmits MCAP advertisements to communicate these channel mapping(s) to other participants in the multicast group. When more than one source transmits MCAP advertisements for the same channel number, the source with the largest physical ID is the owner.

node ID: A 16-bit number that uniquely identifies a Serial Bus node within a group of multiple interconnected buses. The most significant 10 bits are the bus ID and the least significant 6 bits are the physical ID.

node unique ID: A 64-bit number that uniquely identifies a node among all the Serial Bus nodes manufactured worldwide; also known as the EUI-64 (Extended Unique Identifier, 64-bits).

octet: Eight bits of data.

packet: Any of the 1394 primary packets; these MAY be read, write or lock requests (and their responses) or stream data. The term "packet" is used consistently to differentiate 1394 packets from ARP requests/responses, IP datagrams or MCAP advertisements/solicitations.

physical ID: On a particular bus, this 6-bit number is dynamically assigned during the self-identification process and uniquely identifies a node on that bus.

quadlet: Four octets, or 32 bits, of data.

stream packet: A 1394 primary packet with a transaction code of 0x0A that contains a block data payload. Stream packets MAY be either asynchronous or isochronous according to the type of 1394 arbitration employed.

2.3 Abbreviations

The following are abbreviations that are used in this standard:

```
   ARP    Address resolution protocol
   BCM    Broadcast channel manager
   CSR    Control and status register
   CRC    Cyclical redundancy checksum
   EUI-64 Extended Unique Identifier, 64-bits
   GASP   Global asynchronous stream packet
   IP     Internet protocol (within the context of this document, IPv4)
   MCAP   Multicast channel allocation protocol
```

2.4 Numeric values

Decimal and hexadecimal numbers are used within this standard. By editorial convention, decimal numbers are most frequently used to represent quantities or counts. Addresses are uniformly represented by hexadecimal numbers, which are also used when the value represented has an underlying structure that is more apparent in a hexadecimal format than in a decimal format.

Decimal numbers are represented by Arabic numerals or by their English names. Hexadecimal numbers are prefixed by 0x and represented by digits from the character set 0 – 9 and A – F. For the sake of legibility, hexadecimal numbers are separated into groups of four digits separated by spaces.

For example, both 42 and 0x2A represent the same numeric value.

3. IP-CAPABLE NODES

Not all 1394 devices are capable of the reception and transmission of ARP requests/responses or IP datagrams. An IP-capable node SHALL fulfill the following minimum requirements:

- it SHALL implement configuration ROM in the general format specified by ISO/IEC 13213:1994 and SHALL implement the bus information block specified by IEEE P1394a and a unit directory specified by this standard;

- the *max_rec* field in its bus information block SHALL be at least 8; this indicates an ability to accept block write requests and asynchronous stream packets with data payload of 512 octets. The same ability SHALL also apply to read requests; that is, the node SHALL be able to transmit a block response packet with a data payload of 512 octets;

- it SHALL be isochronous resource manager capable, as specified by IEEE Std 1394-1995;

- it SHALL support both reception and transmission of asynchronous streams as specified by IEEE P1394a;

- it SHALL implement the BROADCAST_CHANNEL register as specified by IEEE P1394a; and

- it SHALL be broadcast channel manager (BCM) capable.

4. BROADCAST CHANNEL MANAGER (BCM)

In order for ARP or broadcast IP to function on 1394, a prerequisite is the presence of a broadcast channel manager (BCM). The functions of the BCM specified by IEEE P1394a are as follows:

- the allocation of a channel number for asynchronous stream broadcast (which includes ARP and broadcast IP); and

- the communication of that channel number to all interested nodes (which include IP-capable nodes) on the same bus.

All IP-capable nodes SHALL be capable of functioning as the BCM.

Subsequent to a Serial Bus reset a single BCM SHALL be determined as specified by IEEE P1394a.

In the case that the BCM is unable to allocate a channel number for asynchronous stream broadcast (and consequently for ARP and broadcast IP), a warning SHOULD be communicated to a user that IP initialization could NOT complete because of a lack of Serial Bus resources. The user SHOULD be advised to reconfigure or remove other devices if she wishes to make use of IP.

NOTE: If the BCM is unable to allocate a channel number, IP-capable nodes are unable to use the ARP and broadcast IP methods specified by this document. If other methods ~~(*e.g.*, a search of configuration ROM)~~ | permit IP-capable nodes to discover each other, they MAY be able to send and receive IP datagrams.

5. LINK ENCAPSULATION AND FRAGMENTATION

All IP datagrams (broadcast, unicast or multicast), ARP requests/responses and MCAP advertisements/solicitations that are transferred *via* 1394 block write requests or stream packets SHALL be encapsulated within the packet’s data payload. The maximum size of data payload, in octets, is constrained by the speed at which the packet is transmitted.

Table 1 - Maximum data payloads (octets)

Speed	Asynchronous	Isochronous
S100	512	1024
S200	1024	2048
S400	2048	4096
S800	4096	8192
S1600	8192	16384
S3200	16384	32768

NOTE: The maximum data payloads at speeds of S800 and faster MAY be reduced (but will not be increased) as a result of standardization by IEEE P1394b.

The maximum data payload for asynchronous requests and responses MAY also be restricted by the capabilities of the sending or receiving node(s); this is specified by *max_rec* in either the bus information block or ARP response.

For either of these reasons, the maximum data payload transmissible between IP-capable nodes MAY be less than the 1500 octet maximum transmission unit (MTU) specified by this document. This requires that the encapsulation format also permit 1394 link-level fragmentation and reassembly of IP datagrams.

5.1 Global asynchronous stream packet (GASP) format

Some IP datagrams, as well as ARP requests and responses, MAY be transported *via* asynchronous stream packets. When asynchronous stream packets are used, their format SHALL conform to the global asynchronous stream packet (GASP) format specified by IEEE P1394a. The GASP format illustrated below is INFORMATIVE and reproduced for ease of reference, only.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          data_length          |tag|  channel  |  0x0A |   sy  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           header_CRC                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           source_ID           |        specifier_ID_hi        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|specifier_ID_lo|                    version                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+---                            data                         ---+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            data_CRC                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1 - GASP format

The *source_ID* field SHALL specify the node ID of the sending node and SHALL be equal to the most significant 16 bits of the sender’s NODE_IDS register.

The *specifier_ID_hi* and *specifier_ID_lo* fields together SHALL contain the value 0x00005E, the 24-bit organizationally unique identifier (OUI) assigned by the IEEE Registration Authority Committee (RAC) to IANA.

The *version* field SHALL be zero.

NOTE: Because the GASP format utilizes the first two quadlets of data payload in an asynchronous stream packet format, the maximum payloads cited in Table 1 are effectively reduced by eight octets. In the clauses that follow, references to the first quadlet of data payload mean the first quadlet usable for an IP datagram or ARP request or response. When the GASP format is used, this is the third quadlet of the data payload for the packet.

5.2 Encapsulation header

All IP datagrams transported over 1394 are prefixed by an encapsulation header with one of the formats illustrated below.

If an entire IP datagram MAY be transmitted within a single 1394 packet, it is unfragmented and the first quadlet of the data payload SHALL conform to the format illustrated below.

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | lf|          reserved         |           ether_type          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2 - Unfragmented encapsulation header format

The *lf* field SHALL be zero.

The *ether_type* field SHALL indicate the nature of the datagram that follows, as specified by the following table.

ether_type	Datagram
0x800	IPv4
0x806	ARP
0x8861	MCAP

NOTE: Other network protocols, identified by different values of *ether_type*, MAY use the encapsulation formats defined herein but such use is outside of the scope of this document.

In cases where the length of the datagram exceeds the maximum data payload supported by the sender and all recipients, the datagram SHALL be broken into link fragments; the first two quadlets of the data payload for the first link fragment SHALL conform to the format shown below.

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | lf|rsv|      buffer_size      |           ether_type          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              dgl              |            reserved           |     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3 - First fragment encapsulation header format

The second and subsequent link fragments (up to and including the last) SHALL conform to the format shown below.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| lf|rsv|      buffer_size        |  rsv  |    fragment_offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              dgl                |           reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4 - Subsequent fragment(s) encapsulation header format

The definition and usage of the fields is as follows:

The *lf* field SHALL specify the relative position of the link fragment within the IP datagram, as encoded by the following table.

lf	Position
0	Unfragmented
1	First
2	Last
3	Interior

buffer_size: The size of the buffer, expressed as *buffer_size* + 1 octets, necessary for the recipient to reassemble the link fragments.

ether_type: This field is present only in the first link fragment and SHALL have a value of 0x800, which indicates an IPv4 datagram.

fragment_offset: This field is present only in the second and subsequent link fragments and SHALL specify the offset, in octets, of the fragment from the beginning of the IP datagram. The first octet of the datagram (the start of the IP header) has an offset of zero; the implicit value of *fragment_offset* in the first link fragment is zero.

dgl: The value of *dgl* (datagram label) SHALL be the same for all link fragments of an IP datagram. The sender SHALL increment *dgl* for successive, fragmented datagrams; the incremented value of *dgl* SHALL wrap from 65,535 back to zero.

~~*signature*: The sender SHALL set this field to the most significant 16-bits of its own NODE_IDS register.~~

All IP datagrams, regardless of the mode of transmission (block write requests or stream packets) SHALL be preceded by one of the above described encapsulation headers. This permits uniform software treatment of datagrams without regard to the mode of their transmission.

5.3 Link fragment reassembly

The recipient of an IP datagram transmitted *via* more than one 1394 packet SHALL use both ~~*signature*~~the sender's *source ID* (obtained either from the asynchronous packet header or the GASP header) and *dgl* to identify all the link fragments from a single datagram.

~~NOTE: The use of *signature* for any purposes other than link fragment reassembly is fraught with error and is strongly discouraged.~~

Upon receipt of a link fragment, the recipient MAY place the data payload (absent the encapsulation header) within an IP datagram reassembly buffer at the location specified by *fragment_offset*. The size of the reassembly buffer MAY be determined from *buffer_size*.

If a link fragment is received that overlaps another fragment ~~for~~identified by the same ~~*signature*~~*source ID* and *dgl*, the fragment(s) already accumulated in the reassembly buffer SHALL be discarded. A fresh reassembly MAY be commenced with the most recently received link fragment. Fragment overlap is determined by the combination of *fragment_offset* from the encapsulation header and *data_length* from the 1394 packet header.

Upon detection of a Serial Bus reset, recipient(s) SHALL discard all link fragments of all partially reassembled IP datagrams and sender(s) SHALL discard all not yet transmitted link fragments of all partially transmitted IP datagrams.

6. ADDRESS RESOLUTION PROTOCOL (ARP)

ARP requests SHALL be transmitted by the same means as broadcast IP datagrams; ARP responses MAY be transmitted in the same way or they MAY be transmitted as block write requests addressed to the *sender_unicast_FIFO* address identified by the ARP request. An ARP request/response is 32 octets and SHALL conform to the format illustrated by Figure 5 ~~below~~.

ARP requests are transported by asynchronous stream packets and SHALL be encapsulated within the GASP format specified by IEEE P1394a (see also 5.1). The recipient of an ARP request SHALL ignore it unless the most significant ten bits of the *source_ID* field in the GASP header are equal to either 0x3FF or the most significant ten bits of the recipient's NODE_IDS register.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    hardware_type (0x0018)     |    protocol_type (0x0800)     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  hw_addr_len  |  IP_addr_len  |            opcode             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+---                    sender_unique_ID                     ---+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| sender_max_rec|      sspd     |     sender_unicast_FIFO_hi    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    sender_unicast_FIFO_lo                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      sender_IP_address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      target_IP_address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5 - ARP request/response format

Field usage in an ARP request/response is as follows:

hardware_type: This field indicates 1394 and SHALL have a value of 0x0018.

protocol_type: This field SHALL have a value of 0x0800; this indicates that the protocol addresses in the ARP request/response conform to the format for IP addresses.

hw_addr_len: This field indicates the size, in octets, of the 1394-dependent hardware address associated with an IP address and SHALL have a value of 16.

IP_addr_len: This field indicates the size, in octets, of an IP version 4 (IPv4) address and SHALL have a value of 4.

opcode: This field SHALL be one to indicate an ARP request and two to indicate an ARP response.

sender_unique_ID: This field SHALL contain the node unique ID of the sender and SHALL be equal to that specified in the sender’s bus information block.

sender_max_rec: This field SHALL be equal to the value of *max_rec* in the sender’s configuration ROM bus information block.

sspd: This field SHALL be set to the lesser of the sender’s link speed and PHY speed. The link speed is the maximum speed at which the link MAY send or receive packets; the PHY speed is the maximum speed at which the PHY MAY send, receive or repeat packets. The table below specifies the encoding used for *sspd*; all values NOT specified are RESERVED.

Table 2 - Speed codes

Value	Speed
0	S100
1	S200
2	S400
3	S800
4	S1600
5	S3200

sender_unicast_FIFO_hi and *sender_unicast_FIFO_lo*: These fields together SHALL specify the 48-bit offset of the sender's FIFO available for the receipt of IP datagrams in the format specified by section 7. The offset of a sender's unicast FIFO SHALL NOT change, except as the result of a power reset.

sender_IP_address: This field SHALL specify the IP address of the sender.

target_IP_address: In an ARP request, this field SHALL specify the IP address from which the sender desires a response. In an ARP response, it SHALL be IGNORED.

7. CONFIGURATION ROM

Configuration ROM for IP-capable nodes SHALL contain a unit directory in the format specified by this standard. The unit directory SHALL contain Unit_Spec_ID and Unit_SW_Version entries, as specified by ISO/IEC 13213:1994~~, and OPTIONALLY a Unicast_FIFO entry, as specified by this standard~~.

The unit directory MAY also contain other entries permitted by ISO/IEC 13213:1994 or IEEE P1212r.

7.1 Unit_Spec_ID entry

The Unit_Spec_ID entry is an immediate entry in the unit directory that specifies the organization responsible for the architectural definition of the Internet Protocol capabilities of the device.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     0x12      |            unit_spec_ID (0x00005E)            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6 - Unit_Spec_ID entry format

0x12 is the concatenation of *key_type* and *key_value* for the Unit_Spec_ID entry.

0x00005E is the *unit_spec_ID* obtained by IANA from the IEEE/RAC. The value indicates that IANA and its technical committees are responsible for the maintenance of this standard.

7.2 Unit_SW_Version entry

The Unit_SW_Version entry is an immediate entry in the unit directory that, in combination with the *unit_spec_ID*, specifies the document that defines the software interface of the unit.

```
                    1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     0x13      |                unit_sw_version                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7 - Unit_SW_Version entry format

0x13 is the concatenation of *key_type* and *key_value* for the Unit_SW_Version entry.

A value for *unit_sw_version* is NOT yet specified. When this standard is approved it is EXPECTED that *unit_sw_version* will assume the value of the RFC number assigned at that time. This ~~provides a means to reference~~ assignment algorithm for *unit_sw_version* (or a similar method ratified by IANA) permits the unique identification of future standards that MAY define, for example, IPv6 or extensions to the IPv4 protocol that operate across Serial Bus bridges.

~~7.3Unicast_FIFO entry~~

~~The Unicast_FIFO entry is an immediate entry in the unit directory that, when present, SHALL specify the address of the node's unicast FIFO.~~

```
                    1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     0x54      |              unicast_FIFO_offset               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

~~Figure 8 - Unicast_FIFO entry format~~

~~0x54 is the concatenation of *key_type* and *key_value* for the Unicast_FIFO entry.~~

~~The *unicast_FIFO_offset* field SHALL contain the offset, in quadlets, from the base address of initial register space, 0xFFFF F000 0000, to the address of the unicast FIFO register for the unit. All unit CSRs SHALL be located at or above address 0xFFFF F001 0000; therefore the value of *unicast_FIFO_offset* SHALL NOT be less than 0x4000. The unicast~~

~~FIFO address derived from the Unicast_FIFO entry SHALL be equal to the *sender_unicast_FIFO* address supplied by the node in ARP requests or responses.~~

7.3 Textual descriptors

Textual descriptors within configuration ROM are OPTIONAL; when present they provide additional descriptive information intended to be intelligible to a human user. IP-capable nodes SHOULD associate a textual descriptor with a content of "IANA" with the Unit_Spec_ID entry and a textual descriptor with a content of "IPv4" for the Unit_SW_Version entry.

The figure below illustrates a unit directory implemented by an IP-capable node; it includes OPTIONAL textual descriptors ~~and the OPTIONAL Unicast_FIFO entry~~. Although the textual descriptor leaves are NOT part of the unit directory, for the sake of simplicity they are shown immediately following the unit directory.

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      directory_length (4)     |              CRC              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      0x12     |            unit_spec_ID (0x00005E)            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      0x81     |         textual descriptor offset (3)         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      0x13     |                unit_sw_version                |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      0x81     |         textual descriptor offset (5)         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      0x54     |              unicast_FIFO_offset              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | textual_descriptor_length (3) |              CRC              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   +---                          zeros                          ---+
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      "I"      |      "A"      |      "N"      |      "A"      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | textual_descriptor_length (3) |              CRC              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   +---                          zeros                          ---+
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      "I"      |      "P"      |      "v"      |      "4"      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 9 – Sample unit directory and textual descriptors

8. IP UNICAST

A unicast IP datagram MAY be transmitted to a recipient within a 1394 primary packet that has one of the following transaction codes:

tcode	Description	Arbitration
0x01	Block write	Asynchronous
0x0A	Stream packet	Isochronous
0x0A	Stream packet	Asynchronous

Block write requests are suitable when 1394 link-level acknowledgement is desired but there is no need for bounded latency in the delivery of the packet (quality of service).

Isochronous stream packets provide quality of service guarantees but no 1394 link-level acknowledgement.

The last method, asynchronous stream packets, is mentioned only for the sake of completeness. This method SHOULD NOT be used for IP unicast, since it provides for neither 1394 link-level acknowledgment nor quality of service---and consumes a valuable resource, a channel number.

Regardless of the IP unicast method employed, asynchronous or isochronous, it is the responsibility of the sender of a unicast IP datagram to determine the maximum data payload that MAY be used in each packet. The necessary information MAY be obtained from:

- the SPEED_MAP maintained by the 1394 bus manager, which provides the maximum transmission speed between any two nodes on the local Serial Bus. The bus manager analyzes bus topology in order to construct the speed map; the maximum transmission speed between nodes reflects the capabilities of the intervening nodes. The speed in turn implies a maximum data payload (see Table 1);

- the *target_max_rec* field in an ARP response. This document requires a minimum value of 8 (equivalent to a data payload of 512 octets). Nodes that operate at S200 and faster are encouraged but NOT REQUIRED to implement correspondingly larger values for *target_max_rec*; or

- other methods beyond the scope of this standard.

The maximum data payload SHALL be the minimum of the largest data payload implemented by the sender, the recipient and the PHYs of all intervening nodes (the last is implicit in the SPEED_MAP entry indexed by sender and recipient).

NOTE: The SPEED_MAP is derived from the self-ID packets transmitted by all 1394 nodes subsequent to a bus reset. An IP-capable node MAY observe the self-ID packets directly.

8.1 Asynchronous IP unicast

Unicast IP datagrams that do NOT require any quality of service SHALL be contained within the data payload of 1394 block write transactions addressed to the ~~*target_node*~~*source_ID* and ~~*target*~~*sender_unicast_FIFO* obtained from an ARP response.

If no acknowledgement is received in response to a unicast block write request the state of the target is ambiguous.

NOTE: An acknowledgment MAY be absent because the target is no longer functional, MAY NOT have received the packet because of a header CRC error or MAY have received the packet successfully but the acknowledge sent in response was corrupted.

8.2 Isochronous IP unicast

Unicast IP datagrams that require quality of service SHALL be contained within the data payload of 1394 isochronous stream packets.
The details of coordination between nodes with respect to allocation of channel number(s) and bandwidth are beyond the scope of this standard.

9. IP BROADCAST

Broadcast IP datagrams are encapsulated according to the specifications of section 5 and are transported by asynchronous stream packets. There is no quality of service provision for IP broadcast over 1394. The channel number used for IP broadcast is specified by the BROADCAST_CHANNEL register.

All broadcast IP datagrams SHALL use asynchronous stream packets whose channel number is equal to the *channel* field from the BROADCAST_CHANNEL register.

Although 1394 permits the use of previously allocated channel number(s) for up to one second subsequent to a bus reset, IP-capable nodes SHALL NOT transmit asynchronous stream packets at any time the *valid* bit in their BROADCAST_CHANNEL register is zero. Since the *valid* bit is automatically cleared to zero by a bus reset, this prohibits the use of ARP or broadcast IP until the BCM allocates a channel number.

10. IP MULTICAST

Multicast IP datagrams are encapsulated according to the specifications of section 5 and are transported by stream packets. Asynchronous streams are used for best-effort IP multicast while isochronous streams are used for IP multicast that requires quality of service.

CAUTION: The working group has yet to define facilities and methods for the provision of quality of service for IP multicast.

By default, all best-effort IP multicast SHALL use asynchronous stream packets whose channel number is equal to the *channel* field from the

BROADCAST_CHANNEL register. In particular, datagrams addressed to 224.0.0.1 and 224.0.0.2 SHALL use this channel number. Best-effort IP multicast for other multicast group addresses MAY utilize a different channel number if such a channel number is allocated and advertised prior to use, as described below.

IP-capable nodes MAY transmit best-effort IP multicast only if one of the following two conditions is met:

- the channel number in the stream packet is equal to the *channel* number field in the BROADCAST_CHANNEL register and the *valid* bit in the same register is one; or

- for other channel number(s), some source of IP multicast has allocated and is advertising the channel number used.

The remainder of this section describes a multicast channel allocation protocol (MCAP) employed by both IP multicast sources and recipients whenever a channel number other than the default is used. MCAP is a cooperative protocol; the participants exchange messages over the broadcast channel used by all IP-capable nodes on a particular Serial Bus.

CAUTION: The working group has yet to define facilities and methods for shared use of a single channel number (other than the default channel number specified by the BROADCAST_CHANNEL register) by more than one IP multicast address.

10.1 MCAP message format

MCAP messages, whether sent by a multicast channel owner or recipient, have the format illustrated below. The first four octets of the message are fixed; the remainder consists of variable-length tuples, each of which encodes information about a particular multicast group. Individual MCAP messages SHALL NOT be fragmented and SHALL be encapsulated within a stream packet as *ether_type* 0x8861.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            length             |    reserved   |     opcode    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                          message data                         +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 10 - MCAP message format

Field usage in an MCAP message is as follows:

length: This field SHALL contain the size, in octets, of the entire MCAP message.

opcode: This field SHALL have one of the values specified by the table below.

opcode	Name	Comment
0	Advertise	Sent by a multicast channel owner to broadcast the current mapping(s) from one or more group addresses to their corresponding channel number(s).
1	Solicit	Sent to request multicast channel owner(s) to advertise the indicated channel mapping(s) as soon as possible.

message data: The remainder of the MCAP message is variable in length and SHALL consist of zero or more group address descriptors with the format illustrated below.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    length     |     type      |           reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  expiration   |    channel    |     speed     |   reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           bandwidth                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                         group_address                         +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 11 - MCAP group address descriptor format

length: This field SHALL contain the size, in octets, of the MCAP group address descriptor.

type: This field SHALL have a value of one, which indicates a group address descriptor.

expiration: The usage of this field varies according to *opcode*. For solicit messages the *expiration* field SHALL be IGNORED. Otherwise, for advertisements, this field SHALL contain a time-stamp, in seconds, that specifies a future time after which the channel number specified by *channel* MAY no longer be used.

channel: This field is valid only for advertise messages, in which case it SHALL specify an allocated channel number, in the range zero to 63 inclusive. All other values are RESERVED.

speed: This field is valid only for advertise messages, in which case it SHALL specify the speed at which stream packets for the indicated *channel* are transmitted. Table 2 specifies the encoding used for *speed*.

bandwidth: This field SHALL be zero; it is allocated in the group address descriptor to accommodate future extensions to MCAP that specify quality of service and utilize the isochronous capabilities of Serial Bus.

group_address: This variable length field SHALL specify the IP address of a particular multicast group. The length of *group_address*, in octets, is derived from the length of the group address descriptor by subtracting 12 from the *length* field.

10.2 MCAP message domain

MCAP messages carry information valid only for the local Serial Bus on which they are transmitted. Recipients of MCAP messages SHALL IGNORE all MCAP messages from other than the local bus, as follows. The *source_ID* of the sender is contained in the GASP header that precedes the encapsulated MCAP message. A recipient of an MCAP message SHALL examine the most significant ten bits of *source_ID* from the GASP header; if they are NOT equal to either 0x3FF or the most significant ten~~16~~ bits of the recipient's NODE_IDS register, the recipient SHALL IGNORE the message.

Within an MCAP message domain, the owner of a channel mapping is identified by the *source ID* field in the GASP header of an MCAP advertisement. The owner is the node with the largest physical ID, the least significant six bits of *source ID*.

10.3 Multicast receive

An IP-capable device that wishes to receive multicast data SHALL first ascertain the channel mapping (if any) that exists between a group address and a channel number other than the default channel specified by the BROADCAST_CHANNEL register. Such a device MAY observe the MCAP advertisements on the broadcast channel for the desired channel mapping(s).

An intended multicast recipient MAY transmit MCAP solicitation requests in order to request multicast channel owner(s) to broadcast advertisements sooner than the next ten second interval. Originators of MCAP solicitation requests SHALL limit the rate at which they are transmitted. Subsequent to sending a solicitation request, the originator SHALL NOT send another MCAP solicitation request until 10 seconds have elapsed.

In either case, if a mapping exists for the group address for other than the default channel, an MCAP advertise message is EXPECTED within ten seconds. Upon receipt of an MCAP advertise message that describes one or more channel mappings, the intended multicast recipient MAY receive IP datagrams on the indicated channel number(s) until the *expiration* time.

If multiple MCAP advertise messages are observed that specify the same group address, the channel number SHALL be obtained from the advertisement message with the largest *phy_ID*, which SHALL be obtained from the least significant six bits of *source_ID* from the GASP header.

If no MCAP advertise message is received for a particular group address within 10 seconds, no multicast source(s) are active for channel(s) other than the default. Either there is there is no multicast data or it is being transmitted on the default channel.

Once a multicast recipient has observed an advertisement for the desired group address, it SHALL continue to monitor the broadcast channel for MCAP advertisements for the same group address in order to refresh the *expiration* time of channel number(s) in use.

10.4 Multicast transmit

An IP-capable device that wishes to transmit multicast data on other than the default channel SHALL first ascertain whether or NOT another multicast source has already allocated a channel number for the group address. The intended multicast source MAY transmit an MCAP solicitation request with one or more group address descriptors.

Whether or NOT a solicitation request has been transmitted, the intended multicast source SHALL monitor the broadcast channel for MCAP advertisements. If a channel mapping already exists for the group address, an MCAP advertisement SHOULD be received within ten seconds. In this case the intended multicast source MAY commence transmission of IP datagrams on the indicated channel number(s) and MAY continue to do so until their *expiration* time. The multicast source SHALL monitor MCAP advertisements in order to refresh the *expiration* time of channel number(s) in use.

When no other multicast source has established a channel mapping for the group address, the intended multicast source MAY attempt to allocate a channel number from the isochronous resource manager’s CHANNELS_AVAILABLE register according to the procedures described in IEEE Std 1394-1995. If the channel number allocation is successful, the multicast source SHALL advertise the new channel mapping(s) as soon as possible; once such advertisement has been made, the multicast source MAY transmit IP datagrams using the channel number obtained.

Multicast IP datagrams MAY be transmitted on the default channel until the sender observes (or transmits) an advertisement that specifies non-default channel mapping(s) for the multicast addresses. This permits the smooth transition of multicast from the default channel to an explicitly allocated channel.

10.5 Advertisement of channel mappings

Each multicast source SHALL periodically broadcast an advertisement of all multicast group addresses for which it has allocated a channel number different from the default multicast channel number. An advertisement SHALL consist of a single MCAP message with an *opcode* of zero that contains one or more group address descriptors (one for each group address assigned a channel number other than that specified by the BROADCAST_CHANNEL register).

Within each group address descriptor, the *group_address* and *channel* fields associate a multicast group address with a Serial Bus channel number. More than one multicast group address MAY be mapped to a single Serial Bus channel number by means of separate group address descriptors. The *speed* field specifies the maximum 1394 speed at which any of the senders within the multicast group is permitted to transmit data. The *expiration* field specifies a future time after which the channel mapping(s) are no longer valid. Except when a channel owner intends to relinquish ownership (as described in 10.7 below), the *expiration* time SHALL be at least 60 seconds in the future measured from the time the advertisement is transmitted.

No more than ten seconds SHALL elapse from the transmission of its most recent advertisement before the owner of a channel mapping initiates transmission of the subsequent advertisement. The owner of a channel mapping SHOULD transmit an MCAP advertisement in response to a solicitation as soon as possible after the receipt of the request.

10.6 Overlapped channel mappings

When two intended multicast sources wish to transmit to the same multicast group and no channel mapping exists for the group address, there is a chance that both will allocate channel numbers and both will advertise the channel mappings. These channel mappings overlap, *i.e.*, the same group address is mapped to more than one channel number.

Multicast channel owners SHALL monitor MCAP advertisements in order to detect overlapped channel mappings. When an overlapped channel mapping is detected, the owner with the largest *phy_ID* (as determined by the least significant six bits of *source_ID* from the GASP header) is NOT REQUIRED to take any action. The owner(s) with smaller physical IDs SHALL cease transmission of MCAP advertisements for the overlapped channel number. As soon as these channel mapping(s) are no longer valid, their owners SHALL deallocate any unused channel numbers as described in 10.8 below.

Recipients of MCAP advertisements that detect overlapped channel mappings SHALL ignore the advertisements from multicast channel owner(s) with the smaller physical IDs. It is possible for some channel mappings in a single MCAP advertisement to be valid even if others SHALL be IGNORED as a result of overlap.

10.7 Transfer of channel ownership

The owner of a channel mapping MAY cease multicast transmission on a particular channel, in which case it SHOULD invalidate the channel mapping and in some cases deallocate the channel number. Because other multicast sources MAY be using the same channel mapping, an orderly process is defined to transfer channel ownership.

The owner of an existing channel mapping that wishes to release the mapping SHALL commence a timer to measure the time remaining before the anticipated release of the mapping and its associated channel. Until the timer counts down to zero, the owner SHALL continue to transmit MCAP advertisements for the affected channel but SHALL adjust *expiration* in each advertisement to reflect the time remaining until the channel is to be deallocated. The sequence of *expiration* times transmitted by the owner intending to release the mapping SHALL decrease with each succeeding advertisement. If other multicast source(s) are using the same channel mapping and observe an *expiration* time less than or equal to 30 seconds, they SHALL commence transmitting MCAP advertisements for the channel mapping with refreshed *expiration* times greater than or equal to 30 seconds that maintain the channel mapping. Any contention that occurs between multiple sources that attempt to claim ownership of the channel mapping SHALL be resolved as described in 10.6. If the original owner observes an MCAP advertisement for the channel to be relinquished before its own timer has expired, it SHALL NOT deallocate the channel number.

Otherwise, if the owner’s timer expires without the observation of a MCAP advertisement by another node, the owner of the channel number SHALL deallocate the channel as described below.

10.8 Expired channel mappings

A channel mapping expires when *expiration* seconds have elapsed since the most recent MCAP advertisement. At this time, multicast recipients SHALL stop reception on the expired channel number(s). The owner of the channel mapping(s) SHALL deallocate the channel number and indicate its availability in the isochronous resource manager’s CHANNELS_AVAILABLE register.

10.9 Bus reset

A bus reset SHALL invalidate all multicast channel mappings and SHALL cause all multicast recipients and senders to zero all MCAP advertisement interval timers.

Prior owners of multicast channel mappings MAY reallocate a channel number from the isochronous resource manager’s CHANNELS_AVAILABLE register and resume broadcast of MCAP advertisements as soon as a channel is allocated. If channel reallocation is attempted, the prior owner SHOULD use the same channel number allocated prior to the bus reset and MAY commence reallocation immediately upon completion of the bus reset so long as the same channel number is reused. If the prior owner elects to allocate a different channel number, it SHALL wait until at least one second has elapsed since the completion of the bus reset before attempting to allocate a new channel number.

Intended or prior recipients or transmitters of multicast on other than the default channel SHALL NOT transmit MCAP solicitation requests until at least ten seconds have elapsed since the completion of the bus reset. Multicast data on other than the default channel SHALL NOT be received or transmitted until an MCAP advertisement is observed or transmitted for the multicast group address.

Intended or prior transmitters of multicast on other than the default channel that did not own a channel mapping for the multicast group address prior to the bus reset SHALL NOT attempt to allocate a channel number from the isochronous resource manager's CHANNELS_AVAILABLE register until at least ten seconds have elapsed since the completion of the bus reset. Subsequent to this ten second delay, intended or prior transmitters of multicast MAY follow the procedures specified by 10.4 to allocate a channel number and advertise the channel mapping.

11. IANA CONSIDERATIONS

This document is likely the first that necessitates the creation and management of a new name space (registry) by IANA. The need for such a registry arises out of the method by which protocol interfaces are uniquely identified by bus standards compliant with ISO/IEC 13213:1994, CSR Architecture. This is explained in more detail in section X; the essence is that a globally unique 48-bit number SHALL identify the document that specifies the protocol interface. The 48-bit number is the concatenation of a 24-bit number granted to IANA by the IEEE Registration Authority Committee (RAC) and a second 24-bit number administered by IANA.

The IEEE RAC RECOMMENDS that the policy for management of the second 24-bit number be chosen to maximize the quantity of usable numbers with the range of possible values. By way of a concrete example, the IEEE RAC RECOMMENDS that the assignment scheme NOT apply a structure to the number since this would tend to waste large portions of the range.

In accordance with this guidance, this document suggests that the RFC number of this document be assigned as the 24-bit version number and that this same method be used to assign version numbers to future standards. Requests for version numbers other than RFC numbers should be refereed to a committee of experts selected by IANA.

Regardless of the assignment method elected by IANA, a registry of all assigned version numbers SHOULD be maintained at one or more Internet sites and should clearly identify the relevant standard identified by the version number.

12. SECURITY CONSIDERATIONS

This document specifies the use of an unsecured link layer, Serial Bus, for the transport of IPv4 datagrams. Serial Bus is vulnerable to denial of service attacks; it is also possible for devices to eavesdrop on data or present forged identities. Implementers who utilize Serial Bus for

IPv4 SHOULD consider appropriate counter-measures within application or other layers.

13. ACKNOWLEDGEMENTS

This document represents work in progress by the IP/1394 Working Group. The editor wishes to acknowledge the contributions made by all the active participants, either on the reflector or at face-to-face meetings, which have advanced the technical content.

14. REFERENCES

[1] IEEE Std 1394-1995, Standard for a High Performance Serial Bus

[2] ISO/IEC 13213:1994, Control and Status Register (CSR) Architecture for Microcomputer Buses

[3] IEEE Project P1394a, Draft Standard for a High Performance Serial Bus (Supplement)

[4] IEEE Project P1394b, Draft Standard for a High Performance Serial Bus (Supplement)

15. EDITOR’S ADDRESS

Peter Johansson
Congruent Software, Inc.
98 Colorado Avenue
Berkeley, CA 94602

(510) 527-3926
(510) 527-3856 FAX
pjohansson@aol.com