Network Working Group N. Droux/Brad Allen
WORKING DRAFT Essential/ODS Networks
J.-M. Pittet
SGI
June 2000

Ethernet and HIPPI-800 Bridging

REV 12 (WORKING DRAFT)

Status of this memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet- Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Abstract

This document specifies methods for bridging ANSI High Performance Parallel Interface (HIPPI) networks to Ethernet networks. In particular issues related to addresses resolution and broadcast across these networks.

TABLE OF CONTENTS

1. Introduction

2. Scope

This memo is an effort to allow for bridging between HIPPI-800 and Ethernet. It defines Bridge requirements as well as additional requirements for HIPPI Address Resolution Protocol (HARP) Servers and HARP Clients to attain the following objectives:

o Allow for dynamic hardware address resolution between media.

o Allow for HARP broadcast to interoperate with Ethernet broadcast.

o Allow for multiple Bridges to be attached to a HIPPI network while allowing Bridge protocol data units (PDUs) to be passed across these networks.

No additional requirements for Ethernet hosts are defined by this memo.

2.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119.

3. Definitions

3.1 Global concepts used

In the following discussion, the terms “requester” and “target” are used to identify the node initiating the address resolution request and the node whose address it wishes to discover, respectively. If not all switches in the Local IP Subnet (LIS) support broadcast then there will be a HARP Server providing the address resolution service and it will be the source of the reply. If on the other hand all switches support broadcast then the source address of a reply will be the target’s target address.

3.2 Glossary

Broadcast

A distribution mode which transmits a message to all nodes, including the node sending the message.

Classical/Conventional

Both terms are used to refer to networks such as Ethernet, FDDI, and other 802 LAN types, as distinct from HIPPI-SC LANs.

HARP

HARP describes the whole set of HIPPI address resolution encodings and algorithms defined in [1].

HSAL

The HARP Server Address List (see section 4.2)

HBAL

The HIPPI Bridge Address List (see section 4.2)

LIS

Logical IP subnet

Switch Address

A value used as the address of a node on a HIPPI-SC network. HIPPI-SC compliant switches map Switch Addresses to physical port numbers. The switch address is extended with a mode byte to form an 32-bit I-Field [1].

Universal LAN MAC Address (ULA)

A 48-bit address that uniquely identifies an originating source or final destination. The ULA conforms to the 48-bit MAC address specified by the IEEE 802 Overview and Architecture Standard [7].

4. Network Topology

4.1 Background

In this memo, we consider a network topology where Ethernet and HIPPI networks are connected through one or more Bridges. Ethernet and HIPPI networks implement different hardware addressing mechanisms, and conversion of packets must be implemented to allow these networks to inter-operate.

4.2 HIPPI Requirements in a Bridged Environment

This document assumes that HARP [1] be present on the HIPPI network. For HIPPI hosts that do not implement HARP, static entries SHOULD be defined on the HARP Servers to allow for dynamic address resolution of these hosts from the Ethernet.

HIPPI hosts MUST have unique non-zero ULAs. In the event they don’t, the Bridges or HIPPI hosts MAY provide ULA proxy services where non-zero ULAs are assigned to these hosts. How such a mechanism is implemented is outside the scope of this memo.

The following list identifies HIPPI-specific parameters that MUST be implemented in each HIPPI Bridge connected to the HIPPI network:

o HIPPI hardware address:

The HIPPI hardware address of an individual Bridge MUST contain the Bridge Switch Address and a non-zero ULA. If the Bridge is not connected to a HIPPI-800 switch, rather it is directly connected to a HIPPI-800 endpoint, the Bridge Switch Address can be set to an arbitrary value.

o HARP Server Address List (HSAL):

The HSAL (HARP Server Address List) is an ordered list of one or more HRALs (HARP Request Address List, see [1]). Each HRAL present in the HSAL SHOULD correspond to a defined HIPPI LIS. All HIPPI Bridges SHOULD be configured indentically, i.e. all Bridges SHOULD have the same entries in their HSAL.

Explanation: a Bridge does not implement the IP protocol, and therefore does not have an IP address. For this reason, a Bridge is not a member of any LIS. A Bridge must therefore be able to interoperate with more than one HIPPI LIS. If more than one LIS is present on the HIPPI network, a Bridge's HSAL will contain the hardware address of each HARP server (which could be one or many) in the LIS.

All HRALS of a HSAL SHALL be constructed as described by Section 4.2 of [1]. The HSAL MUST contain at least one HRAL, which will be referred to in this memo as the primary HRAL.

Within the restrictions presented above and in [1], local administrators choose addresses for the HARP Servers on the HIPPI network which one stores in the HSAL. Methods for the administration of the HSAL are beyond the scope of this memo.

Each HARP Server compliant with this memo SHALL implement a HIPPI Bridge Address List (HBAL). The HBAL is a list of zero or more addresses identifying HIPPI Bridges. All HARP Servers SHOULD be configured identically, i.e. all HARP Servers SHOULD have the same address(es) in their HBAL. Each entry of the HBAL contains the Switch Address and unique ULA of the Bridge.

4.3 Bridge Registration

In a non-broadcast capable HIPPI environment, HARP Servers must be able to identify messages received from, and send specific messages to HIPPI Bridges in order to provide support for dynamic address resolution of Ethernet MAC addresses, as well as to provide the other services described by this memo. For this purpose, HARP Servers conforming to this memo SHALL implement a HBAL (HIPPI Bridge Address List). The HBAL allows HARP Servers to identify the Bridges present on the HIPPI network. HARP Servers gain knowledge of the presence of HIPPI Bridges during the Bridge registration phase.

The Bridge registration phase consists, for the Bridges, of sending InHarp_REQUEST messages to the HARP Servers identified by the Bridge HSAL (HARP Server Address List). This registration process is presented in this section.

4.3.1 Bridge Registration Phase

Bridges SHALL initiate the registration phase by sending an InHARP_REQUEST message to the addresses described by the entries of the HSAL. The Bridge SHALL terminate the registration phase and transition into the operational phase, either when it receives its own InHarp_REQUEST message or when it receives an InHarp_REQUEST message from at least one of the HARP Servers.

The format of a InHARP_REQUEST message sent by a Bridge for registration with a HARP Server is identical to that of a HARP Client with the exception that the Requestor and Target protocol address is always set to zero.

```
HIPPI-LE Destination_IEEE_Address   = HSAL_ULA
HIPPI-LE Destination_Switch_Address = HSAL_SW
HIPPI-LE Source_IEEE_Address        = bridge_ULA
HIPPI-LE Source_Switch_Address      = bridge_SW
LLC                                 = 0xAAAA0300
SNAP                                = 0x00000806
```

```
HARP ar$hrd                          = HIPARP – 28
HARP ar$pro                          = IP protocol code - 2048
HARP ar$op                          = InHARP_REQUEST
HARP ar$pln                          = 4
HARP ar$rhl                          = 10
HARP_ar$thl                          = 10
HARP ar$rpa                          = 0
HARP ar$tpa                          = 0
HARP ar$rha                          = bridge_SW, bridge_ULA
HARP ar$tha                          = HSAL_SW, HSAL_ULA
```

HSAL_ULA and HSAL_SW are the ULA and Switch Address of the HARP Server. bridge_ULA and bridge_SW are the ULA and Switch Address of the Bridge. The ar$pln (protocol address length) field of the HARP message SHALL contain 4 for Bridge registration requests.

HARP Servers must recognize that InHARP_REQUEST messages with Requestor and Target protocol addresses of zero as Bridge registration messages.

When Bridges are initiated they send an InHARP_REQUEST to the selected address. The first address to be tried will be the first address of the primary HRAL, i.e. the broadcast address "0x07000FE1 FF:FF:FF:FF:FF:FF". There are two outcomes:

1. The Bridge sees its own InHARP REQUEST: then the Bridge is connected to a broadcast capable network. The first address becomes and remains the selected address for the HARP service.

2. The node does not receive its InHARP REQUEST: then the node is connected to a non-broadcast capable network.

In the second case, the Bridge SHALL continue to send registration messages to each non-broadcast entry in the HSAL at least once every 5 seconds until a reply is received from each address. A InHARP_REPLY sent by a HARP Server in reply to a Bridge registration request SHALL have the following values:

```
HIPPI-LE Destination_IEEE_Address   = bridge_ULA
HIPPI-LE Destination_Switch_Address = bridge_SW
HIPPI-LE Source_IEEE_Address        = HSAL_ULA
HIPPI-LE Source_Switch_Address      = HSAL_SW
HARP ar$op                          = InHARP_REPLY
HARP ar$pln                         = 4
HARP ar$rha                         = HSAL_SW, HSAL_ULA
HARP ar$tha                         = bridge_SW, bridge_ULA
```

Each HSAL entry for which the Bridge receives a registration reply message SHALL be marked by the Bridge as a responsive HARP Server.

If the HIPPI LAN supports broadcast, then the Bridge will see its own InHARP_REQUEST message and SHALL complete the registration phase. The Bridge SHOULD further note that it is connected to a broadcast capable network and use this information for the processing of packets received from the Ethernet as specified in section 5.4.

If the Bridge doesn't see its own InHARP_REQUEST message, then it SHALL await an InHARP_REPLY before completing the registration phase. This will also provide the client with the ULA and Switch Address by which the HARP Server is addressable. This will be the case when the client happens to be connected to a non-broadcast capable network.

When a Bridge receives a InHARP_REPLY   from a HARP Server, it SHOULD mark this HARP Server as responsive and SHALL send InHARP_REQUEST registration requests to this server at least every 5 minutes.

4.3.2 Bridge Operational Phase

Once a Bridge has completed its registration phase, it enters the operational phase. In this phase of the protocol, the Bridge SHALL provide services as specified by [4] and convert address resolution messages between the two media as specified by this memo. In the operational phase, the Bridge behavior for providing these services is the same for broadcast or non-broadcast networks.

5. Address Resolution in the Presence of Bridges

HARP [1] defines the behavior of HIPPI nodes to allow the implementation of dynamic ARP services and IP broadcast emulation within a LIS. In the presence of a Bridge, HARP Servers must implement additional processing to allow for dynamic address resolution transparently across different media. In this section, we present the address resolution algorithm in the presence of HIPPI to Ethernet Bridges.

5.1 HARP Algorithm in the Presence of Bridges

For the following discussion, we assume that a simple network is composed of a HIPPI-800 host, a HARP client, which has registered itself with a HARP Server. The HARP client and server are connected to a HIPPI switch. A HIPPI-800 to Ethernet Bridge is connected to both the HIPPI switch and the Ethernet. Finally, an Ethernet host is present and is connected to the Bridge. This topology is depicted by Figure 1.

```
                         +--------+
                         |  HARP  |
                         | Server |                    |    +------+
                         +--------+                    +----|Ether |
                             |                         |    |Node  |
                             |                         |E   +------+
                             |                         |t
           +------+      +========+      /------\      |h
           |HIPPI |------| HIPPI  |------|Bridge|------|e
           |Node  |      | Switch |      |      |      |r
           +------+      +========+      \------/      |n
                                                       |e
                                                       |t
```

|

Figure 1. Topology in Presence of a Bridge

We assume that the HIPPI-800 client host is registered with the HARP Server, i.e. the IP address, ULA, and Switch Addresses of the client are known by the HARP Server. We also assume that the Bridge is registered with the HARP Server.

5.1.1 Resolution of Ethernet MAC Addresses from HIPPI

The HARP algorithm [1] is extended by this memo to take into account the presence of Bridges on the network. In the presence of one or more Bridges, the following additional steps are REQUIRED to allow for dynamic Ethernet address resolution from HIPPI nodes:

1. Upon reception of a HARP_REQUEST for a target protocol address that is not known by the HARP Server, the HARP Server SHALL send a corresponding HARP_REQUEST to every entry in the HARP SERVER HBAL, and send a HARP_PENDING message (see Section 7.1.1) back to the requesting HIPPI node.

2. The HARP_REQUEST is translated by each Bridge into a corresponding Ethernet ARP_REQUEST (see Section 5.4.1) which is broadcast on the Ethernet.

3. The ARP_REQUEST sent by the Bridge is eventually received by the target Ethernet node which replies with a ARP_REPLY message which is sent to the requestor.

4. The ARP_REPLY is received by the Ethernet port of the Bridge. It is translated by the Bridge into a corresponding HARP_REPLY message (see Section 5.4.2) which is sent to all responsive server entries in the Bridge HSAL.

5. The HARP_REPLY is received by the HARP Server, and is processed as specified by HARP [1]. This mapping will become known by the HIPPI nodes by either gratutious HARP or next time a HARP_REQUEST is received by the HARP Server for this entry.

5.1.2 Resolution of HIPPI HW Address from Ethernet

The following additional steps are REQUIRED to allow for dynamic resolution of HIPPI hardware addresses from an Ethernet node:

1. Upon reception of a ARP_REQUEST, a Bridge SHALL convert this message into a corresponding HARP_REQUEST (see Section 5.4.2) A Bridge SHALL then send this HARP_REQUEST to every authoritative HARP Server in the HSAL.

2. The HARP_REQUEST is processed by the HARP Server as specified by [1] and a HARP_REPLY is sent to the requestor through the Bridge.

3. The HARP_REPLY message is received by the Bridge and converted into a corresponding Ethernet ARP_REPLY (see Section 5.4.1) which is sent on the Ethernet.

4. The ARP_REPLY is received by the original Ethernet requestor which updates its table.

5.1.3 Multiple HIPPI LIS

As defined by [1], each LIS is served by one or more distinct HARP Servers. When more than one LIS are present on the HIPPI network, a Bridge SHALL maintain one HRAL per LIS in its HSAL (HARP Server Address List). ARP messages received by a Bridge from Ethernet SHALL be converted by the Bridge into corresponding HARP messages (see Section 5.4.2) and SHALL be sent to all responsive HARP Server entries of the HSAL.

Each HARP Server MUST accept HARP messages coming from Bridges and SHALL discard those messages that refer to IP addresses that are not part of the LIS of the HARP Server. HARP messages pertaining to the LIS of the HARP Server SHALL be processed as specified by [1] and this memo.

5.1.4 Multiple Bridges

When multiple Bridges are connected to the HIPPI network, every HARP Server, when present in one of the HSAL of the Bridges, will have learned about the presence of every Bridge during the Bridge’s registration phase. HARP Servers SHALL create and maintain entries for every Bridge in their HBAL (HIPPI Bridge Address List).

All HARP_REQUEST's that cannot be resolved by a HARP Server SHALL be forwarded to every entry in the HBAL, as defined in Section 5.2.2 of this memo.

Similarly, all IP broadcast messages received by a HARP Server SHALL be processed as specified by [1] and SHALL be forwarded to every entry of the HBAL. Non-IP broadcast or multicast packets received by a HARP Server from a Bridge SHALL be forwarded to all entries in the HARP Server HBAL, thus allowing multicast or broadcast Bridge PDUs [4] received by one Bridge to be sent to the other Bridges connected to the HIPPI network (see Section 6.1).

5.2 Additional HARP Server Operational Requirements

When the HBAL of the HARP Server contains no entry, i.e. no Bridges are present on the network the HARP Server SHALL process HARP messages as described by [1] otherwise the HARP Server must also implement the additional requirements described in this section.

5.2.1 HIPPI Bridge Address List

The HARP Server SHALL maintain a table of known Bridges, the HIPPI Bridge Address List (HBAL), as described in Section 4.2. Each element of this table contains the HIPPI hardware address, i.e. Switch Address and unique ULA, of a Bridge which has previously registered itself with the HARP Server (see Section 4.3.)

When a InHARP_REQUEST message is received by a HARP Server, then the HARP Server SHALL proceed as follows:

1. If the requestor protocol address (ar$rpa) is zero, the HARP Server SHALL add an entry in the HBAL using the requester ULA and Switch Address found in the InHARP_REQUEST (ar$rha), and SHALL send a InHARP_REPLY to the requester of the InHARP_REQUEST.
2. If the requester protocol address (ar$rpa) is non-zero, the HARP Server SHALL process the InHARP REQUEST as specified by [1].

When a Bridge table entry ages beyond at most 20 minutes, the HARP Server MUST delete the entry from the HBAL table. The entry expiration timer must be reset each time a Bridge registration request is received from this Bridge.

5.2.2 Additional HARP Message Processing

Upon receiving a HARP_REQUEST message, a HARP Server SHALL process the request as defined by [1]. If an entry is not found for the target IP address specified in the HARP message (ar$tpa), the HARP Server SHALL proceed as follows:

1. If the HARP_REQUEST was sent by a Bridge, then the HARP Server SHALL forward the HARP_REQUEST to all Bridges except the Bridge which sent the HARP_REQUEST.

2. If the HARP_REQUEST was sent by a HARP client, then the HARP Server SHALL:

   (a) send a HARP_PENDING (see Section 7.1.1) message to the client, and
   (b) forward the HARP_REQUEST to all registered Bridges using the HBAL.

Upon receiving a HIPPI HARP_REPLY message from a Bridge, a HARP Server SHALL process this message as described by [1], and in addition, SHALL forward this HARP_REPLY to all Bridges in the HBAL except the Bridge from which the HARP_REPLY was received.

5.3 Additional HARP Client Operational Requirements

HARP clients complying to this memo SHALL implement HARP as defined by [1]. In addition, HARP clients SHALL accept and process

HARP_PENDING messages, the format of which is described in Section 7.1.1.

5.4 Bridge Operational Requirements

Bridges SHALL provide services as specified by [4]. In addition, Bridges SHALL convert HARP messages received from the HIPPI network into corresponding Ethernet ARP messages, and similarly ARP messages received from the Ethernet into HARP messages, before transmission on Ethernet or HIPPI, respectively. These translations SHALL be performed by Bridges as specified by the following subsection.

5.4.1 HARP Messages from HIPPI-800 to Ethernet

Upon receiving a HARP_REQUEST or a HARP_REPLY from HIPPI, the Bridge SHALL construct an Ethernet ARP_REQUEST or ARP_REPLY, respectively, and send the resulting packet on the Ethernet.

A Ethernet ARP_REQUEST message SHALL contain the following values, where the prefix harp_ indicates a field of the HARP_REQUEST received by the Bridge from the HIPPI network. See Section 7.2 and [3] for a description of the Ethernet ARP message format.

```
Ethernet MAC header fields:
  Destination MAC Address - SHALL contain ff:ff:ff:ff:ff:ff
  Source MAC Address      - SHALL contain the ULA part of harp_ar$rha

ARP_REQUEST fields:
  ar$hrd                  - SHALL contain 1 (Ethernet)
  ar$pro                  - SHALL contain 0x0806 (ARP)
  ar$pln                  - SHALL contain 4
  ar$hln                  - SHALL contain 6
  ar$op                   - SHALL contain ARP_REQUEST
  ar$sha                  - SHALL contain the ULA part of harp_ar$rha
  ar$spa                  - SHALL contain harp_ar$rpa
  ar$tha                  - SHALL contain 0
  ar$tpa                  - SHALL contain harp_ar$tpa
```

An Ethernet ARP_REPLY message SHALL contain the following values, where the prefix harp_ indicates a field of the HARP_REPLY received by the Bridge from the HIPPI network.Ethernet MAC header fields:

```
  Destination MAC Address - SHALL contain the ULA part of harp_ar$tha
  Source MAC Address      - SHALL contain the ULA part of harp_ar$rha

ARP_REPLY fields:
  ar$hrd                  - SHALL contain 1 (Ethernet)
  ar$pro                  - SHALL contain 0x0806 (ARP)
  ar$pln                  - SHALL contain 4
  ar$hln                  - SHALL contain 6
  ar$op                   - SHALL contain ARP_REPLY
  ar$sha                  - SHALL contain the ULA part of harp_ar$rha
  ar$spa                  - SHALL contain harp_ar$rpa
  ar$tha                  - SHALL contain the ULA part of harp_ar$tha
```

```
     ar$tpa                  - SHALL contain harp_ar$tpa
```

5.4.2 ARP Messages from Ethernet to HIPPI-800

Upon receiving an Ethernet ARP_REPLY, the Bridge SHALL form a corresponding HIPPI HARP_REPLY and send this message to all responsive servers of the HSAL.

A HARP_REPLY SHALL contain the following values, where ar$ fields denote Ethernet ARP_REPLY message fields and harp_ar$ denote HARP_REPLY message fields.

```
   HIPPI-LE fields:
     Dest Switch Addr   - SHALL contain the switch addr of the
                          HARP Server
     Source Switch Addr - SHALL contain the switch addr of the Bridge
     Destination ULA    - SHALL contain the ULA of the HARP Server
     Source ULA         - SHALL contain ar$sha

   HARP message fields:
     harp_ar$op         - SHALL contain HARP_REPLY
     harp_ar$rpa        - SHALL contain ar$spa
     harp_ar$tpa        - SHALL contain ar$tpa
     harp_ar$rha        - SHALL contain the switch address of the Bridge
                          and ar$sha
     harp_ar$tha        - SHALL contain the hardware address of ar$tpa
```

Explanation: the HARP_REPLY message above will cause all registered HIPPI-800 registered HARP clients to add a new entry for the source Ethernet host with the IP address and ULA of this host and the logical address of the Bridge which originally received the Ethernet ARP_REPLY.
Upon receiving an Ethernet ARP_REQUEST, the Bridge SHALL form a corresponding HIPPI HARP_REQUEST message and send this request to the authoritative HARP Server in the Bridge HSAL.

A HARP_REQUEST SHALL contain the following values, where ar$ fields denote ARP_REQUEST message fields and harp_ar$ denote HARP_REQUEST fields.

```
   HIPPI-LE fields:
     Dest Switch Addr   - SHALL contain the switch addr of the
                          HARP Server
     Source Switch Addr - SHALL contain the switch addr of the Bridge
     Destination ULA    - SHALL contain the ULA of the Server
     Source ULA         - SHALL contain ar$sha

   HARP message fields:
     harp_ar$op         - SHALL contain HARP_REQUEST
     harp_ar$rpa        - SHALL contain ar$spa
     harp_ar$tpa        - SHALL contain ar$tpa
     harp_ar$rha        - SHALL contain the Switch Address of the Bridge
                          and ar$sha
     harp_ar$tha        - SHALL contain 0
```

5.4.3 Processing of IP Traffic

The MTU used by HIPPI networks for IP packets [5] is larger than the MTU used by Ethernet networks. These MTU sizes can be handled by a Bridge using the following methods:

1. A Bridge can drop received HIPPI IP packets that are too large to be encoded on Ethernet. In this case the participating hosts are responsible to negotiate a packet size that is compatible for the HIPPI and Ethernet networks.

2. A Bridge can provide support for Path MTU Discovery [6].

This memo does not mandate the implementation of either one of these methods by a Bridge.

6. Broadcast and Multicast in the Presence of a Bridge

HARP [1] defines IP broadcast emulation within a LIS in the presence of a non-broadcast capable HIPPI network. In this section we define additional HARP Server requirements that allow for:

1. extending LIS broadcast emulation to other media in the presence of Bridges, and
2. allowing for non-IP broadcast and multicast messages received from a Bridge, such as Bridge PDUs (BPDUs) as defined by [4], to be broadcasted to the other Bridges present on the HIPPI network, if any.

6.1 Additional HARP Server Requirements

When no Bridges are known by a HARP Server, i.e. the HBAL of the HARP server is empty and the HARP Server SHALL provide PIBES services as described by [1].

If one or more Bridges are present on the network, a IP broadcast message is considered valid if one of the following two conditions is satisfied:

1. The message satisfies the requirements for a valid message as per [1], Section 7.1, or

2. The message was sent to the HARP Server by one of the Bridges in the HBAL, and the source IP address is part of the logical IP subnet of the HARP Server.

In the presence of one or more Bridges, i.e. when the HBAL is non empty, a HARP Server SHALL perform the additional processing:

1. Valid IP broadcast messages received by the broadcast server SHALL be processed as specified by [1], Section 7.1. In addition to forwarding IP broadcast messages to all entries in the HARP server table (the target list), the HARP Server SHALL proceed as follows:

   (a) If the message was received from a client, the HARP Server SHALL forward the message to all Bridges in the HBAL.

   (b) If the message was received from a Bridge, the HARP Server SHALL forward the message to all Bridges except the Bridge from which it was received from.

2. If a non-IP broadcast message with a destination ULA corresponding to the broadcast address ff:ff:ff:ff:ff or a multicast address is received by the HARP Server from a Bridge, the message SHALL be forwarded to all entries in the HBAL except the Bridge from which the message was received. In this case the destination HIPPI-LE destination IEEE ULA SHALL NOT be changed by the HARP Server.

7. Message Encoding

This RFC refers to two different message formats:

1. HIPPI-800 HARP messages: defined by [1], these messages are used by the HIPPI-800 network to resolve HIPPI-800 hardware addresses. See Section 7.1.

2. Ethernet ARP messages: defined by [3], these messages are used by Ethernet host to resolve Ethernet MAC addresses. See Section 7.2

In this section, we summarize the format of HIPPI-800 HARP messages, Ethernet ARP messages, and any additional HARP messages required by this memo.

7.1 HARP and InHARP Message Formats

HARP and InHARP messages are described in [1], portions of which are repeated here as an aid to the reader.

```
    31    28        23  21          15        10     7         2   0
    +-----+---------+-+-+-----------+---------+-----+---------+-----+
  0 |      04       |1|0|         000         |      03       |  0  |
    +---------------+-+-+---------------------+---------------+-----+
  1 |                              45                               |
    +-----+-+-------+-----------------------+-----------------------+
  2 |[LA] |W|MsgT= 0|          000          |   Dest. Switch Addr   |
    +-----+-+-------+-----------------------+-----------------------+
  3 |   2   |   2   |          000          |  Source Switch Addr   |
    +---------------+---------------+-------+-----------------------+
  4 |             00 00             |                               |
    +-------------------------------+                               |
  5 |                      Destination ULA                          |
    +-------------------------------+-------------------------------+
  6 |             [LA]              |                               |
    +-------------------------------+                               |
  7 |                          Source's ULA                         |
    +===============+===============+===============+===============+
  8 |       AA      |      AA       |       03      |       00      |
    +---------------+---------------+---------------+---------------+
  9 |       00      |      00       |        Ethertype (2054)       |
    +---------------+---------------+-------------------------------+
 10 |              hrd (28)         |           pro (2048)          |
    +---------------+---------------+---------------+---------------+
 11 |             op (ar$op)        |     pln (4)   |   rhl (q)     |
    +---------------+---------------+---------------+---------------+
 12 |    thl = (x)  |    Requester IP Address upper  (24 bits)      |
    +---------------------------------------------------------------+
 13 | Req. IP lower |       Target IP Address upper  (24 bits)      |
    +---------------+-----------------------------------------------+
 14 | Tgt. IP lower |  Requester HIPPI Hardware Address bytes 0 - 2 |
    +---------------+-----------------------------------------------+
 15 |          Requester HIPPI Hardware Address bytes 3 - 6         |
    +-----------------------------------------------+---------------+
 16 |          Requester HW Address bytes 7 - 9     |Tgt's HW byte 0|
    +---------------+---------------+---------------+---------------+
 17 |           Target  HIPPI Hardware Address bytes 1 - 4          |
    +---------------------------------------------------------------+
 18 |           Target  HIPPI Hardware Address bytes 5 - 8          |
    +---------------+---------------+---------------+---------------+
 19 |Tgt HW byte 9-x|     FILL      |     FILL      |     FILL      |
    +---------------+---------------+---------------+---------------+
                             HARP - InHARP Message
```

Data sizes and field meaning:

```
  ar$hrd  16 bits  Hardware type
  ar$pro  16 bits  Protocol type of the protocol fields below
  ar$op   16 bits  Operation code (request, reply, or NAK)
  ar$pln   8 bits  byte length of each protocol address
  ar$rhl   8 bits  requester's HIPPI hardware address length (q)
  ar$thl   8 bits  target's HIPPI hardware address length (x)
  ar$rpa  32 bits  requester's protocol address
  ar$tpa  32 bits  target's protocol address
  ar$rha  qbytes   requester's HIPPI Hardware address
  ar$tha  xbytes   target's HIPPI Hardware address
```

For a detailed description of these fields refer to [1].

7.1.1 HARP_PENDING Message Format

This memo defines a new HARP message operation, HARP_PENDING. HARP clients conforming to this memo SHALL accept and process HARP_PENDING HARP messages. The ar$op operational value field of a HARP_PENDING HARP message SHALL be set to 11.

The HARP_PENDING message format is the same as the received HARP_REQUEST message format with the operation code set to HARP_NAK; i.e. the HARP_REQUEST message data is copied byte for byte for transmission with the HARP_REQUEST operation code changed to the HARP_PENDING value.

7.2 Ethernet ARP Message Format

Ethernet ARP messages are described in details in [3]. For convenience, we list here the various fields of these messages. The Ethernet ARP packet data has the following fields:

```
    |31             |23             |15             |7             0|
    +---------------+---------------+---------------+---------------+
  0 |                          D_ULA (0-3)                          |
    +---------------------------------------------------------------+
  1 |           D_ULA (4-5)         |           S_ULA (0-1)         |
    +---------------------------------------------------------------+
  2 |                          S_ULA (2-5)                          |
    +---------------------------------------------------------------+
  3 |           ftype = 0x0806      |        ar$hrd = 1             |
    +---------------------------------------------------------------+
  4 |          ar$pro = 0x0800      |  ar$hln = 6   |  ar$pln = 4   |
    +---------------------------------------------------------------+
  5 |             ar$op             |         ar$sha (0-1)          |
    +---------------------------------------------------------------+
  6 |                           ar$sha (2-5)                        |
    +---------------------------------------------------------------+
  7 |                           ar$spa                              |
    +---------------------------------------------------------------+
  8 |                           ar$tha (0-3)                        |
    +---------------------------------------------------------------+
  9 |          ar$tha (4-5)         |         ar$tpa (0-1)          |
    +---------------------------------------------------------------+
 10 |          ar$tpa (2-3)         |
    +-------------------------------+
```

Ethernet ARP Message Format [3]

```
      Data sizes and field meaning:
        D_ULA   48 bits  Destination ULA
        S_ULA   48 bits  Source ULA
        ftype   16 bits  Frame type

        ar$hrd  16 bits  Hardware type
        ar$pro  16 bits  Protocol type of the protocol fields below
        ar$hln   8 bits  byte length of each hardware address
        ar$pln   8 bits  byte length of each protocol address
        ar$op   16 bits  Operation code
        ar$sha  nbytes   sender hardware address
        ar$spa  mbytes   sender protocol address
        ar$tha  nbytes   target hardware address
        ar$tpa  mbytes   target protocol address
```

8. Open Issues

9. Examples

Assume a network is composed of a HIPPI-800 Server S with a hardware address HWs (ULAs, SWs), a HIPPI-800 Client C with hardware address HWc (ULAc, SWc), a Ethernet host E with hardware address ULAe, and a Bridge B with hardware address HWb (ULAb, SWb). Nodes S, C, and E each have a unique IP address, respectively IPs, IPc, and IPe.

```
Node S: HARP server     IPs, ULAs, SWs
Node C: HARP client     IPc, ULAc, SWc
Node E: Ethernet host   IPe, ULAe
Node B: Bridge          ULAb, SWb
```

We assume that the HIPPI-800 client C is registered with the HIPPI Server S, and that the HIPPI Server S knows that the Bridge B is present on the network at switch address SWb with ULAb. We also assume that HIPPI client C and HIPPI server S have initially no knowledge of the hardware address of Ethernet host E.

9.1 Resolution of a Ethernet MAC Address from HIPPI-800

1. HIPPI Node C starts and would like to obtain the hardware address (ULAe, SWe) corresponding to the IP address IPe of the Ethernet host.

```
HIPPI-LE Destination_IEEE_Address   = ULAs
HIPPI-LE Destination_Switch_Address = SWs
HIPPI-LE Source_IEEE_Address        = ULAc
HIPPI-LE Source_Switch_Address      = SWc
HARP ar$op                          = HARP_REQUEST
HARP ar$rpa                         = IPc
HARP ar$rha                         = SWc ULAc
HARP ar$tpa                         = IPe
HARP ar$tha                         = 0 **
** is what we are looking for
```

2. Server S receives the request and look in its tables, but does not find an entry for Ipe. Server S sends a HARP_PEND message to HIPPI-800 node C and sends a HARP_REQUEST to Bridge B:

```
HIPPI-LE Destination_IEEE_Address   = FF:FF:FF:FF:FF:FF
HIPPI-LE Destination_Switch_Address = SWb
HIPPI-LE Source_IEEE_Address        = ULAs
HIPPI-LE Source_Switch_Address      = SWs
HARP ar$op                          = HARP_REQUEST
HARP ar$rpa                         = IPc
HARP ar$rha                         = SWc ULAc
HARP ar$tpa                         = IPe
HARP ar$tha                         = 0 **

** is what we are looking for
```

3. Bridge B receives the HARP_REQUEST from Server S and converts the HARP message into a Ethernet ARP_REQUEST message:

```
Ethernet Dst_IEEE_Address  = FF:FF:FF:FF:FF:FF
Ethernet Src_IEEE_Address  = ULAc
ARP ar$op                  = ARP_REQUEST
ARP ar$rpa                 = IPc
ARP ar$rha                 = ULAc
ARP ar$tpa                 = IPe
ARP ar$tha                 = 0 **
** is what we are looking for
```

4. The Ethernet ARP_REQUEST is eventually resolved by Node E which sends back a Ethernet ARP_REPLY:

```
Ethernet Dst_IEEE_Address  = ULAc
Ethernet Src_IEEE_Address  = ULAe
ARP ar$op                  = ARP_REPLY
ARP ar$rpa                 = IPe
ARP ar$rha                 = ULAe *
ARP ar$tpa                 = IPc
ARP ar$tha                 = ULAc
* what we were looking for
```

5. The Bridge B will receive this ARP_REPLY and converts it into a HARP_REPLY message which is sent to the HARP Server S:

```
HIPPI-LE Destination_IEEE_Address   = ULAs
HIPPI-LE Destination_Switch_Address = SWs
HIPPI-LE Source_IEEE_Address        = ULAe
HIPPI-LE Source_Switch_Address      = SWb
HARP ar$op                          = HARP_REPLY
HARP ar$rpa                         = IPe
HARP ar$rha                         = SWb ULAe
HARP ar$tpa                         = IPc
HARP ar$tha                         = SWc ULAc
```

6. The HARP Server S receives the HIPPI HARP_REPLY message. The HARP Server S updates its table and sends this HARP_REPLY to all known Clients.

9.2 Resolution of a HIPPI-800 Hardware Address from Ethernet

Ethernet Node E starts and would like to obtain the hardware address ULAc corresponding to the protocol address IPc of HIPPI-800 Node C.

1. Node E broadcasts a Ethernet ARP_REQUEST

```
Ethernet Destination_Address        = FF:FF:FF:FF:FF:FF
Ethernet Source_Address             = ULAe
ARP ar$op                           = ARP_REQUEST
ARP ar$rpa                          = IPe
ARP ar$rha                          = ULAe
ARP ar$tpa                          = IPc
ARP ar$tha                          = 0 *
* is what we are looking for
```

2. Bridge B receives the Ethernet ARP_REQUEST which it converts into a HARP_REQUEST message which is sent to the HARP Server S:

```
HIPPI-LE Destination_IEEE_Address   = ULAs
HIPPI-LE Destination_Switch_Address = SWs
HIPPI-LE Source_IEEE_Address        = ULAe
HIPPI-LE Source_Switch_Address      = SWb
HARP ar$op                          = HARP_REQUEST
HARP ar$rpa                         = IPe
HARP ar$rha                         = SWb ULAe
HARP ar$tpa                         = IPc
HARP ar$tha                         = 0 *
* is what we are looking for
```

3. The HARP Server S receives the HARP_REQUEST. It looks for an entry with IP address Ipc. This entry exists since the HARP Server knows about client C. It generates a HIPPI HARP_ REPLY which it sends to the Bridge.

```
HIPPI-LE Destination_IEEE_Address   = ULAe
HIPPI-LE Destination_Switch_Address = SWb
HIPPI-LE Source_IEEE_Address        = ULAs
HIPPI-LE Source_Switch_Address      = SWs
HARP ar$op                          = HARP_REPLY
HARP ar$rpa                         = IPc
HARP ar$rha                         = SWc ULAc **
HARP ar$tpa                         = IPe
HARP ar$tha                         = SWb ULAe
** is what we were looking for
```

4. Bridge B receives the HARP_REPLY from the HARP Server S and converts this message into a ARP_REPLy which is sent on the Ethernet:

```
Ethernet Dst_IEEE_Address  = ULAe
Ethernet Src_IEEE_Address  = ULAc
ARP ar$op                  = ARP_REPLY
ARP ar$rpa                 = IPc     (from HARP ar$rpa)
ARP ar$rha                 = ULAc    (from HARP ar$rha)
ARP ar$tpa                 = IPe     (from HARP ar$tpa)
ARP ar$tha                 = ULAe ** (from HARP ar$tha)
** is what we were looking for
```

5. The Ethernet ARP_REPLY is received by Node E which can update its local ARP table. Node E will be able to send packets to Node C using ULAc. When a packet is sent through the Bridge from Node E to Node C, the Bridge will find the switch address SWc corresponding to ULAc from the mapping learned during step 4. (b) and forwards the resulting packet to Node C.

10. References

[1] Pittet, J.-M., "ARP and IP Broadcast over HIPPI-800", RFC-????, Silicon Graphics, Inc., December 1998.

[2] ANSI X3.222-1997, Information Technology - High-Performance Parallel Interface - Physical Switch Control (HIPPI-SC).

[3] Plummer, D., "An Ethernet Address Resolution Protocol - or - Converting Network Addresses to 48-bit Ethernet Address for Transmission on Ethernet Hardware", RFC-826, MIT, November 1982.

[4] ANSI/IEEE Std 802.1D-1993, Information technology - Telecommunication and information exchange between systems - Local area networks - Media access control (MAC) Bridges, IEEE, New York, 1993.

[5] Renwick, J., "IP over HIPPI", RFC-2067, NetStar, Inc., January 1997.

[6] Mogul, J. and Deering S., "Path MTU Discovery", RFC-1191, November 1990.

[7] ANSI/IEEE Std 802-1990, IEEE Local and Metropolitan Area Networks : Overview and Architecture.

11. Acknowledgments

12. Author's Address

Brad Allen
Essential/ODS Networks
1551 Mercantile Ave. NE, Suite A
Albuquerque, NM 87107-7001 USA

Phone: 505-344-0080
Fax: 505-344-0408
Email: ballen@esscom.com

Nicolas Droux
Essential/ODS Networks
1551 Mercantile Ave. NE, Suite A
Albuquerque, NM 87107-7001 USA

Phone: 505-344-0080
Fax: 505-344-0408
EMail: droux@esscom.com

Jean-Michel Pittet
SGI
2011 N. Shoreline Ave
Mountain View, CA 94040

Phone: 605-933-6149
Fax: 605-933-3542
EMail: jmp@sgi.com