                   ARP and IP Broadcast over HIPPI-800

Status of this Memo

   This document specifies an Internet standards track protocol for the
   Internet community, and requests discussion and suggestions for
   improvements. Please refer to the current edition of the "Internet
   Official Protocol Standards" (STD 1) for the standardization state
   and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Abstract

   This document specifies a method for resolving IP addresses to ANSI
   High-Performance Parallel Interface (HIPPI) hardware addresses and
   for emulating IP broadcast in a logical IP subnet (LIS) as a direct
   extension of HARP. This memo defines a HARP that will interoperate
   between HIPPI-800 and HIPPI-6400 (also known as Gigabyte System
   Network, GSN). This document (when combined with RFC-2067 "IP over
   HIPPI") obsoletes RFC-1374.

Table of Contents

1. Introduction

   The ANSI High-Performance Parallel Interface (HIPPI) is a dual
   simplex data channel.   HIPPI can send and receive data
   simultaneously at 800 or 1600 megabits per second. Between 1987 and
   1997, the ANSI X3T11.1 HIPPI working group (now known as NCITS T11.1)
   Standardized five documents that bear on the use of HIPPI as a
   network interface. They cover the physical and electrical
   specification (HIPPI-PH [1]), the framing of a stream of bytes
   (HIPPI-FP [2]), encapsulation of IEEE 802.2 LLC (HIPPI-LE [3]), the
   behavior of a physical layer switch (HIPPI-SC [4]) and the physical-
   level and optical specification (HIPPI-Serial [5]). HIPPI-LE also
   implies the encapsulation of Internet Protocol[5]. The reader should
   be familiar with the ANSI HIPPI documents. Approved ANSI NCITS

standards are available from ANSI (http://www.ansi.org). The working
documents of the T11.1 working group may be obtained from the T11 web
page (http://www.t11.org/).

HIPPI switches can be used to connect a variety of computers and
peripheral equipment for many purposes, but the working group stopped
short of describing their use as Local Area Networks. RFC-2067 [15]
describes the encapsulation of IP over HIPPI-800. This memo takes up
where the working group and RFC-2067 [15] left off and defines
address resolution and LIS IP broadcast emulation for HIPPI-800
networks.

While investigating possible solutions for HARP it became evident
that IP broadcast could easily be emulated for both HIPPI-800 and
HIPPI-6400 hardware types. This is useful since HIPPI switches are
not required to implement broadcast but many standard networking
protocols rely on broadcast. This memo therefore further addresses
the emulation of LIS IP broadcast as an extension of HARP.

2 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 [18].

3. Definitions

3.1 Global concepts used

In the following discussion, the terms "requester" and "target" are
used to identify the port initiating the address resolution request
and the port whose address it wishes to discover, respectively. If
not all switches in the 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.

Values are decimal unless otherwise noted. Formatting follows IEEE
802.1A canonical bit order and and HIPPI-FP bit and byte order.

3.2 Glossary

Broadcast

A distribution mode which transmits a message to all ports.
Particularly also the port 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.

   Destination

   The HIPPI port that receives data from a HIPPI Source.

   HARP

   HARP describes the whole set of HIPPI address resolution encodings
   and algorithms defined in this memo. HARP is a combination and
   adaptation of the Internet Address Resolution Protocol (ARP) RFC-826
   [13] and Inverse ARP (InARP) [7] (see section 5). HARP also describes
   the HIPPI specific version of ARP [10] (i.e. the protocol and the
   HIPPI specific encoding).

   HARP table

   Each host has a HARP table which contains the IP to hardware address
   mapping of IP members.

   HIPPI-Serial

   An implementation of HIPPI in serial fashion on coaxial cable or
   optical fiber. (see [5])

   HRAL

   The HARP Request Address List. A list of ULAs to which HARP messages
   are sent when resolving names to addresses (see section 4.2).

   Hardware (HW) address

   The hardware address of a port consisting of an I-Field and an
   optional ULA (see section 6.2). Note: the term port as used in this
   document refers to a HIPPI port and is roughly equivalent to the term
   "interface" as commonly used in other IP documents.

   Host

   An entity, usually a computer system, that may have one or more HIPPI
   ports and which may serve as a client or a HARP server.

   Port

   An entity consisting of one HIPPI Source/Destination dual simplex
   pair that is connected by parallel or serial HIPPI to a HIPPI-SC
   switch and that transmits and receives IP datagrams.

   PIBES

   The Protocol for Internet Broadcast Emulation Server (see section 7).

   Switch Address

   A value used as the address of a port on a HIPPI-SC network. It is
   transmitted in the I-field. HIPPI-SC switches map Switch Addresses
   to physical switch port numbers. The switch address is extended with
   a mode byte to form an I-Field (see [4] and 6.2.2)

   Source

   The HIPPI port that generates data to send to a HIPPI Destination.

   Universal LAN MAC Address (ULA)

   A 48-bit globally unique address, administered by the IEEE, assigned
   to each port on an Ethernet, FDDI, 802 network, or HIPPI-SC LAN.

4. IP Subnetwork Configuration

4.1 Background

   ARP (address resolution protocol) as defined in [12] was meant to
   work on the 'local' cable. This definition gives the ARP protocol a
   local logical IP subnet (LIS) scope. In the LIS scenario, each
   separate administrative entity configures its hosts and routers
   within the LIS. Each LIS operates and communicates independently of
   other LIS's on the same HIPPI network.

   HARP has LIS scope only and serves all ports in the LIS.
   Communication to ports located outside of the local LIS is usually
   provided via an IP router. This router is a HIPPI port attached to
   the HIPPI network that is configured as a member of one or more
   LIS's. This configuration MAY result in a number of disjoint LIS's
   operating over the same HIPPI network. Using this model, ports of
   different IP subnets SHOULD communicate via an intermediate IP router
   even though it may be possible to open a direct HIPPI connection
   between the two IP members over the HIPPI network. This is a
   consequence of using IP and choosing to have multiple LIS's on the
   same HIPPI fabric.

By default, the HARP method detailed in section 5 and the classical
LIS routing model MUST be available to any IP member client in the
LIS.

4.2 HIPPI LIS Requirements

The requirement for IP members (hosts, routers) operating in a HIPPI
LIS configuration is:

o All members of the LIS SHALL have the same IP network/subnet
   address and address mask [6].

The following list identifies the set of HIPPI-specific parameters
that MUST be implemented in each IP station connected to the HIPPI
network:

o HIPPI Hardware Address:

   The HIPPI hardware address of an individual IP port MUST contain
   the port's Switch Address (see section 9). The address SHOULD also
   contain a non-zero ULA address. If there is no ULA then that field
   MUST be zero.

o HARP Request Address List (HRAL):

   The HRAL is an ordered list of two or more addresses identifying
   the address resolution service(s). All HARP clients MUST be
   configured identically, i.e. all ports MUST have the same
   addresses(es) in the HRAL.

   The HRAL MUST contain at least two HIPPI HW addresses identifying
   the individual HARP service(s) that have authoritative
   responsibility for resolving HARP requests of all IP members
   located within the LIS.

   By default the first address MUST be the reserved address for
   broadcast, i.e. the address for "IP traffic conventionally
   directed to the IEEE 802.1 broadcast address: 0xFE1" [4]. The ULA
   for this HARP service entry SHALL be FF:FF:FF:FF:FF:FF.

   It is REQUIRED that the second address be the address for
   "Messages pertaining to (the) ... address resolution requests:
   0xFE0" [4]. The ULA for this HARP server entry is
   00:00:00:00:00:00.

Therefore, the HRAL entries are sorted in the following order:
```
  1st **  : broadcast address            (0x07000FE1 FF:FF:FF:FF:FF:FF),
  2nd **  : official HARP server address (0x07000FE0 00:00:00:00:00:00),
  3rd & on: any additional HARP server addresses will be sorted in
            decreasing order of the 12bit destination switch
            address portion of their I-Field (see section 6.2).
  ** REQUIRED
```

Within the restrictions mentioned above and in Section 6.2.2, local administration choose address(es) for the additional HARP services which they will put into the HRAL.

```
  An example of such a list:
    1st entry: 0x07000FE1 FF:FF:FF:FF:FF:FF
    2nd entry: 0x07000FE0 00:00:00:00:00:00
    3rd entry: 0x07000001 <Alternate-HARP-server-ula>
    ...
```

Manual configuration of the addresses and address lists presented in this section is implementation dependent and beyond the scope of this memo.

5. HIPPI Address Resolution Protocol - HARP

Address resolution within the HIPPI LIS SHALL make use of the HIPPI Address Resolution Protocol (HARP) and the Inverse HIPPI Address Resolution Protocol (InHARP). HARP provides the same functionality as the Internet Address Resolution Protocol (ARP). HARP is based on ARP which is defined in RFC-826 [13]. Knowing the Internet address, conventional networks use ARP to discover another port's hardware address. HARP presented in this section further specifies the combination of the original protocol definitions to form a coherent address resolution service that is independent of the hardware's broadcast capability.

InHARP is based on the original Inverse ARP (InARP) protocol presented in [7]. Knowing its hardware address, InARP is used to discover the other party's Internet address.

This memo further REQUIRES the PIBES (see section 7 below) extension to the HARP protocol, guaranteeing broadcast service to upper layer protocols like IP.

Internet addresses are assigned independent of ULAs and switch addresses. Before using HARP, each port MUST know its IP and its hardware addresses. The ULA is optional but is RECOMMENDED if bridging to conventional networks is desired.

5.1 HARP Algorithm

This section defines the behavior and requirements for HARP
implementations on both broadcast and non-broadcast capable HIPPI-SC
networks. HARP creates a table in each port which maps the IP address
of each port to a hardware address, so that when an application
requests a connection to a remote port by its IP address, the
hardware address can be determined, a correct HIPPI-LE header can be
built, and a connection to the port can be established using the
correct Switch Address in the I-field.

HARP is a two phase protocol. The first phase is the registration
phase and the second phase is the operational phase. In the
registration phase the port detects if it is connected to broadcast
hardware or not. The InHARP protocol is used in the registration
phase. In case of non-broadcast capable hardware, the InHARP
Protocol will register and establish a table entry with the server.
The operational phase works much like conventional ARP with the
exception of the message format.

5.1.1 Selecting the authoritative HARP service

Within the HIPPI LIS, there SHALL be an authoritative HARP service.
At each point in time there is only one authoritative HARP service.

To select the authoritative HARP service, each port needs to
determine if it is connected to a broadcast network.

The port SHALL send an InHARP_REQUEST to the first address in its
HRAL (0x07000FE1 FF:FF:FF:FF:FF:FF). If the port sees its own
InHARP_REQUEST, then it is connected to a broadcast capable network.
In this case, the rest of the HRAL is ignored and the authoritative
HARP service is the broadcast entry.

If the port is connected to a non-broadcast capable network, then the
port SHALL send the InHARP_REQUEST to all of the remaining entries in
the HRAL. Every address which sends an InHARP_REPLY is considered to
be a responsive HARP server. The authoritative HARP service SHALL be
the HARP server which appears first in the HRAL.

The sequence of the HRAL is only important for deciding which address
will be the authoritative one. On a non-broadcast network, the port
is REQUIRED to keep "registered" with all HARP server addresses in
the HRAL (NOTE: not the broadcast address since it is not a HARP
server address). If for instance the authoritative HARP service is
non-responsive, then the port will consider the next address in the
HRAL as a candidate for the authoritative address and send an
InHARP_REQUEST.

The authoritative HARP server SHOULD be considered non-responsive
when it has failed to reply to: (1) one or more registration requests
by the client (see section 5.1.2 and 5.2), (2) any two HARP_REQUESTs
in the last 120 seconds or (3) if an external agent has detected
failure of the authoritative HARP server. The details of such an
external agent and its interaction with the HARP client are beyond
the scope of this document. Should an authoritative HARP server
become non-responsive, then the registration process SHOULD be
restarted. Alternative methods for choosing an authoritative HARP
service are not prohibited.

5.1.2 HARP registration phase

HARP clients SHALL initiate the registration phase by sending an
InHARP_REQUEST message using the addresses in the HRAL in order. The
client SHALL terminate the registration phase and transition into the
operational phase, either when it receives its own InHARP_REQUEST or
when it receives an InHARP_REPLY from at least one of the HARP
servers and when it has determined the authoritative HARP service as
described in section 5.1.1.

When ports are initiated they send an InHARP_REQUEST to the
authoritative address as described in section 5.1.2. The first
address to be tried will be the broadcast address "0x07000FE1
FF:FF:FF:FF:FF:FF". There are two outcomes:

1. The port sees its own InHARP_REQUEST: then the port is connected
   to a broadcast capable network. The first address becomes and
   remains the authoritative address for the HARP service.

2. The port does not receive its InHARP_REQUEST: then the port is
   connected to a non-broadcast capable network.

In the second case, the port SHALL choose the next address in the
HRAL as a candidate for a authoritative address and send an
InHARP_REQUEST to that address: (0x07000FE0 00:00:00:00:00:00).

o If the port receives its own message, then the port itself is the
   HARP server and the port is REQUIRED to provide broadcast services
   using the PIBES (see section 7).

o If the port receives an InHARP_REPLY, then it is a HARP client and
   not a HARP server.

In both cases, the current candidate address becomes the
authoritative HARP service address.

If the client determines it is connected to a non-broadcast capable
network then the client SHALL continue to retry each non-broadcast
HARP server address in the HRAL at least once every 5 seconds until
one of these two termination criteria are met for each address.

InHARP is an application of the InARP protocol for a purpose not
originally intended. The purpose is to accomplish registration of
port IP address mappings with a HARP server if one exists or detect
hardware broadcast capability.

If the HIPPI-SC LAN supports broadcast, then the client will see its
own InHARP_REQUEST message and SHALL complete the registration phase.
The client SHOULD further note that it is connected to a broadcast
capable network and use this information for aging the HARP server
entry and for IP broadcast emulation as specified in sections 5.4 and
5.6 respectively.

If the client doesn't see its own InHARP_REQUEST, then it SHALL await
an InHARP_REPLY before completing the registration phase. This will
also provide the client with the protocol 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 HIPPI-SC network.

5.1.3 HARP operational phase

Once a HARP client has completed its registration phase it enters the
operational phase. In this phase of the protocol, the HARP client
SHALL gain and refresh its own HARP table which contains the IP to HW
address mapping of IP members by sending HARP_REQUESTS to the
authoritative address in the HRAL and receiving HARP_REPLYs. The
client is fully operational during the operational phase.

In the operational phase, the client's behavior for requesting HARP
resolution is the same for broadcast or non-broadcast networks.

The target of an address resolution request updates its address
mapping tables with any new information it can find in the request.
If it is the target port it SHALL formulate and send a reply message.
A port is the target of an address resolution request if at least ONE
of the following statements is true of the request:

1. The port's IP address is in the target protocol address field
   (ar$tpa) of the HARP message.

2. The port's ULA (if non-zero), is in the ULA part of the Target
   Hardware Address field (ar$tha) of the message.

   3. The port's switch address is in the Target Switch Address field of
      Target Hardware Address field (ar$tha) of the message (see section
      6.2.2).

   4. The port is a HARP server.

   NOTE: It is RECOMMENDED that all HARP servers run on a ports which
   each have a non-zero ULA.

5.2 HARP Client Operational Requirements

   The HARP client is responsible for contacting the HARP server(s) to
   have its own HARP information registered and to gain and refresh its
   own HARP entry/information about other IP members. This means, as
   noted above, that HARP clients MUST be configured with the hardware
   address of the HARP server(s) in the HRAL.

   HARP clients MUST:

   1. When an interface is enabled (e.g. "ifconfig <interface> up" with
      an IP address) or assigned the first or an additional IP address
      (i.e. an IP alias), the client SHALL initiate the registration
      phase.

   2. In the operational phase the client MUST respond to HARP_REQUEST
      and InHARP_REQUEST messages if it is the target port. If an
      interface has multiple IP addresses (e.g., IP aliases) then the
      client MUST cycle through all the IP addresses and generate an
      InHARP_REPLY for each such address. In that case an InHARP_REQUEST
      will have multiple replies. (Refer to Section 7, "Protocol
      Operation" in RFC-1293 [7].)

   3. React to address resolution reply messages appropriately to build
      or refresh its own client HARP table entries. All solicited and
      unsolicited HARP_REPLYs from the authoritative HARP server SHALL
      be used to update and refresh its own client HARP table entries.

      Explanation: This allows the HARP server to update the clients
      when one of server's mappings change, similar to what is
      accomplished on Ethernet with gratuitous ARP.

   4. Generate and transmit InHARP_REQUEST messages as needed and
      process InHARP_REPLY messages appropriately (see section 5.1.2 and
      5.6). All InHARP_REPLY messages SHALL be used by the client to
      build or refresh its HARP table entries. (Refer to Section 7,
      "Protocol Operation" in [7].)

If the registration phase showed that the hardware does not support
broadcast, then the client MUST refresh its own entry for the HARP
server, created during the registration phase, at least once every 15
minutes. This can be accomplished either through the exchange of a
HARP request/reply with the HARP server or by repeating step 1. To
decrease the redundant network traffic, this timeout SHOULD be reset
after each HARP_REQUEST/HARP_REPLY exchange.

Explanation: The HARP_REQUEST shows the HARP server that the client
is still alive. Receiving a HARP_REPLY indicates to the client that
the server must have seen the HARP_REQUEST.

If the registration phase shows that the underlying network supports
broadcast, then periodic InHARP_REQUEST/InHARP_REPLY operations of
step 4 are NOT REQUIRED.

5.3 Receiving Unknown HARP Messages

If a HARP client receives a HARP message with an operation code
(ar$op) that it does not support, it MUST gracefully discard the
message and continue normal operation. A HARP client is NOT REQUIRED
to return any message to the sender of the undefined message.

5.4 HARP Server Operational Requirements

A HARP server MUST accept HIPPI connections from other HIPPI ports.
The HARP server expects an InHARP_REQUEST as the first message from
the client. A server examines the IP source address, the hardware
source address of the InHARP_REQUEST and adds or updates its HARP
table entry <IP address(es), switch address, ULA> as well as the
time stamp.

A HARP server SHALL reply to HARP_REQUESTs and InHARP_REQUESTs based
on the information which it has in its HARP table. The HARP server
SHALL reply with a HARP_REPLY or a InHARP_REPLY, if it has the
requested information in its tables; otherwise it SHALL reply with a
HARP_NAK. The HARP server replies SHALL contain the hardware type and
corresponding format of the request (see also section 6).

The following table shows all possible source address combinations on
an incoming message and the actions to be taken. "linked" indicates
that an existing "IP entry" is linked to a "hardware entry". It is
possible to have an existing "IP entry" and to have an existing
"hardware entry" but neither is linked to the other.

```
+---+--------+--------+-----------+---------------------+
| # | IP entry | HW entry |   misc    |       Action        |
+---+--------+--------+-----------+---------------------+
| 1 | exists  | exists  |    linked | *                   |
| 2 | exists  | exists  | not linked | *, a, b,      e, f  |
| 3 | exists  |   new   | not linked | *, a, b,   d, e, f  |
| 4 |  new    | exists  | not linked | *,       c,   e, f  |
| 5 |  new    |   new   | not linked | *,       c, d, e, f |
+---+--------+--------+-----------+---------------------+
```
Actions:
*: update timeout value
a: break the existing IP -> hardware (HW) - old link
b: delete HW(old) -> IP link and decrement HW(old) refcount, if
   refcount = 0, delete HW(old)
c: create new IP entry
d: create new HW entry
e: add new IP -> HW link to IP entry
f: add new HW -> IP link to HW entry

Examples of when this could happen (Numbers match lines in above
table):

1: supplemental message

   Just update timer.

2: move an IP alias to an existing interface

   If the IP source address of the InHARP_REQUEST duplicates a table
   entry IP address (e.g. IPa <-> HWa) and the InHARP_REQUEST
   hardware source address matches a hardware address entry (e.g. HWb
   <-> IPb), but they are not linked together, then:
   - HWa entry needs to have its reference to the current IPa
      address removed.
   - HWb needs to have a new reference to IPa added
   - IPa needs to be linked to HWb

3: move IP address to a new interface

   If the InHARP_REQUEST requester's IP source address duplicates a
   table entry IP address and the InHARP_REQUEST hardware source
   address does not match the table entry hardware address, then a
   new HW entry SHALL be created. The requestor's IP address SHALL be
   moved from the original HW entry to the new one (see above).

   4: add IP alias to table

      If the InHARP_REQUEST requester's hardware source address
      duplicates a hardware source address entry, but there is no IP
      entry matching the received IP address, then the IP address SHALL
      be added to the hardware entries previous IP address(es). (E.g.
      adding an IP alias).

   5: fresh entry, add it

      Standard case, create both entries and link them.

   A server MUST update the HARP table entry's timeout for each
   HARP_REQUEST. Explanation: if the client is sending HARP requests to
   the server, then the server SHOULD note that the client is still
   "alive" by updating the timeout on the client's HARP table entry.

   A HARP server SHOULD use the PIBES (see section 7) to send out
   HARP_REPLYs to all hardware addresses in its table when the HARP
   server table changes mappings. This feature decreases the time of
   stale entries in the clients.

   If there are multiple addresses in the HRAL, then a server needs to
   act as a client to the other servers.

5.5 HARP and Permanent ARP Table Entries

   An IP station MUST have a mechanism (e.g. manual configuration) for
   determining what permanent entries it has. The details of the
   mechanism are beyond the scope of this memo. The permanent entries
   allow interoperability with legacy HIPPI adapters which do not yet
   implement dynamic HARP and use a table-based static ARP. Permanent
   entries are not aged.

   The HARP server SHOULD use the static entries to resolve incoming
   HARP_REQUESTS from the clients. This feature eliminates the need for
   maintaining a static HARP table on the client ports.

5.6 HARP Table Aging

   HARP table aging MUST be supported since IP addresses, especially IP
   aliases and also interfaces (with their ULA), are likely to move.
   When so doing the mapping in the clients own HARP table/cache becomes
   invalid and stale.

   o When a client's HARP table entry ages beyond 15 minutes, a HARP
      client MUST invalidate the table entry.

   o When a server's HARP table entry ages beyond 20 minutes, the HARP
      server MUST delete the table entry.

   NOTE: the client SHOULD revalidate a HARP table entry before it ages,
   thus restarting the aging time when the table entry is successfully
   revalidated. The client MAY continue sending traffic to the port
   referred to by this entry while revalidation is in progress, as long
   as the table entry has not aged. The client MUST revalidate an aged
   entry prior to transmitting any non-address-resolution traffic to the
   port referred to by this entry.

   The client revalidates the entry by querying the HARP server with a
   HARP_REQUEST. If a valid reply is received (e.g. HARP_REPLY), the
   entry is updated. If the address resolution service cannot resolve
   the entry (e.g. HARP_NAK, "host not found"), the associated table
   entry is removed. If the address resolution service is not available
   (i.e. "server failure") the client MUST attempt to revalidate the
   entry by transmitting an InHARP_REQUEST to the hardware address of
   the entry in question and updating the entry on receipt of an
   InHARP_REPLY. If the InHARP_REQUEST attempt fails to return an
   InHARP_REPLY, the associated table entry is removed.

6. HARP Message Encoding

   The HARP Message is encapsulated over HIPPI-FP and HIPPI-LE headers.
   The HARP FP header values are to be set as defined in RFC-2067 "IP
   over HIPPI" [15]. The following sections detail the HIPPI-LE field
   contents and HARP message structure and contents. In a broadcast
   capable network the client MAY also support Type 1 and 6, Ethernet
   and IEEE 802 ARP packet formats.

6.1 HIPPI-LE Header of HARP Messages

   The HIPPI message format for Internet datagrams shall conform to the
   HIPPI-FP [2] and HIPPI-LE [3] standards. The length of a HIPPI
   message, including trailing fill, shall be a multiple of eight bytes
   as required by HIPPI-LE. The HIPPI-LE header fields of HARP and
   InHARP requests and replies SHALL be:

   FC (3 bits) SHALL contain zero.

   Double-wide SHOULD be set according to HIPPI-LE [3]. This memo does
   NOT address the implications on HARP when this bit is set to 1
   indicating the possibility of a port being able to accept 64-bit
   HIPPI connections.

Message_Type SHALL contain 0 to indicate a data message. HARP messages are identified using the Ethertype and the message type in the ar$op field of the HARP message.

Destination_Switch_Address, SHALL be the Switch Address of the destination port.

Destination_IEEE_Address SHALL be the ULA of the destination port, if known, otherwise zero.

Destination_Address_Type SHALL be 2, a 12-bit logical address. The behavior with type = 1, source routing, is NOT defined in this specification.

Source_Switch_Address in requests SHALL be the sender's Switch Address.

Source_IEEE_Address SHALL be the sender's ULA if known, otherwise zero.

Source_Address_Type SHALL be 2, a 12-bit logical address. The behavior with type = 1, source routing, is NOT defined in this specification.

6.1.1 IEEE 802.2 LLC

The IEEE 802.2 LLC Header SHALL begin in the first byte of the HIPPI-FP D2_Area.

The LLC value for SSAP-DSAP-CTL SHALL be 0xAA-AA-03 (3 bytes) indicating the presence of a SNAP header.

6.1.2 SNAP

The OUI value for Organization Code SHALL be 0x00-00-00 (3 bytes) indicating that the following two-bytes is an Ethertype.

The Ethertype value SHALL be set as defined in Assigned Numbers [16]: InHARP = InARP = HARP = ARP = 2054 = 0x0806.

The total size of the LLC/SNAP header is fixed at 8-bytes.

6.1.3 HIPPI-LE header Diagram

HIPPI-LE header for HARP/InHARP PDUs:

```
     31    28        23 21           15      10   7          2   0
     +-----+---------+-+-+-----------+--------+-----+---------+-----+
   0 | 04 = IP ULP   |1|0|    000    |       03          |  0  |
     +-----+---------+-+-+-----------+--------+-----+---------+-----+
   1 |                         n + 8                               |
     +-----+-+-------+-----------+--------+--------------------------+
   2 |[LA] |W|M_Type |    000    |    Dest. Switch Addr            |
     +-----+-+-------+-----------+--------+--------------------------+
   3 | D_A_T | S_A_T |    000    |    Source Switch Addr           |
     +-------+-------+-----------+--------+--------------------------+
   4 |        00 00             |                                   |
     +---------------------------+                                   |
   5 |                   Destination ULA                            |
     +---------------------------+--------------------------------+
   6 |        [LA]              |                                   |
     +---------------------------+                                   |
   7 |                      Source ULA                              |
     +=============+=============+==============+=================+
   8 |     AA      |     AA      |     03       |      00         |
     +-------------+-------------+--------------+-----------------+
   9 |     00      |     00      |     Ethertype (2054)           |
     +-------------+-------------+--------------+-----------------+
  10 |Message byte 0 |Message byte 1 |Message byte 2 | . . .       |
     +-------------+-------------+--------------+---              |
     |                         . . .                              |
     +  -----------+-------------+--------------+-----------------+
     |      . . .  | byte (n-2)  | byte (n-1)   |      FILL        |
     +-------------+-------------+--------------+-----------------+
 N-1|    FILL      |    FILL     |    FILL      |      FILL        |
     +-------------+-------------+--------------+-----------------+
```

HIPPI Message Format
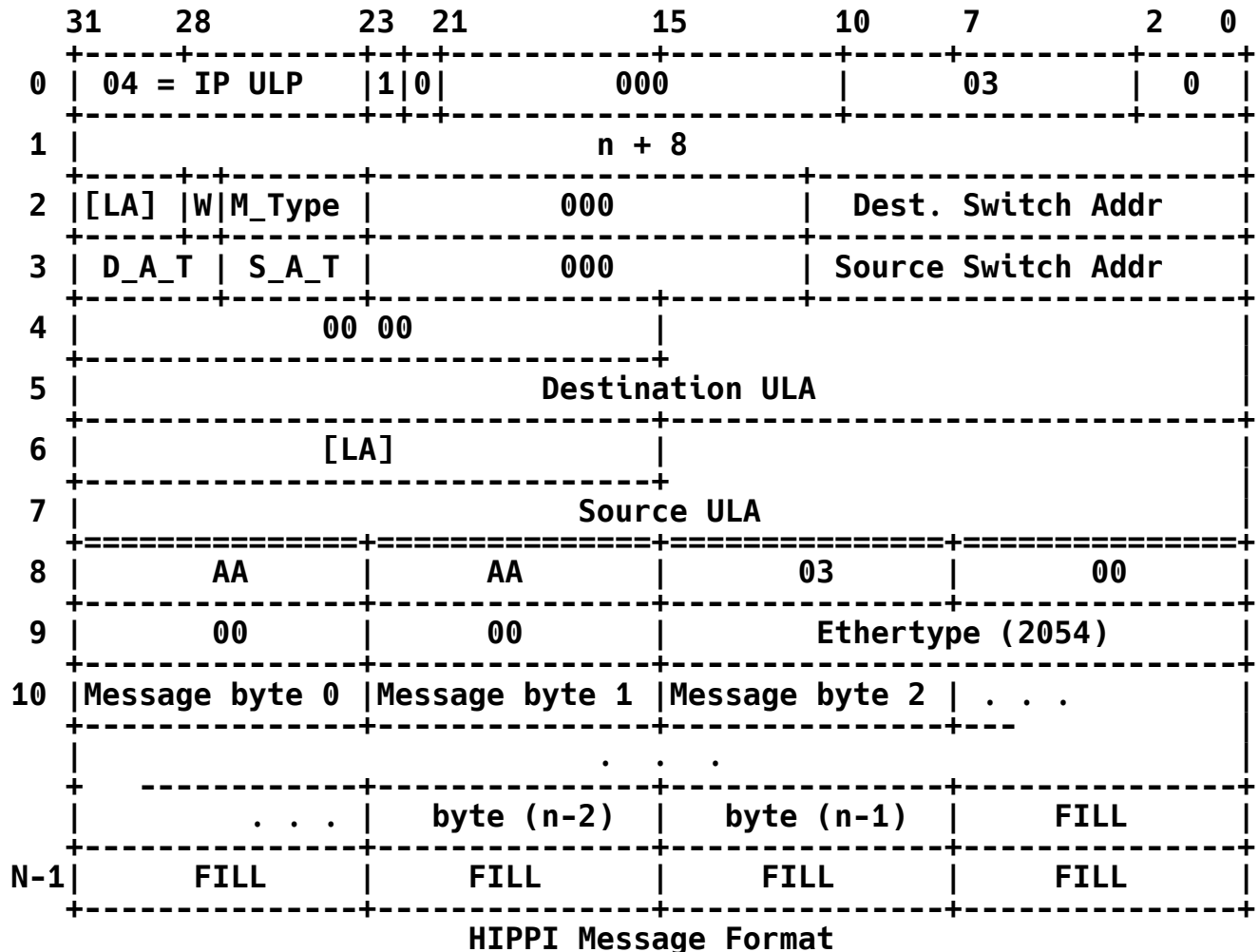
```
    Words 0-1:  HIPPI-FP Header
    Words 2-7:  D1_Area (HIPPI-LE Header)
    Words 8-9:  D2_Area (IEEE 802.2 LLC/SNAP)
    Words 10-(N-1): D2_Area          (HARP message)
    (n+8) is the nb of bytes in the  HARP message, incl. LLC/SNAP.
    +====+ denotes the boundary between D1_Area and D2_Area.
    [LA] fields are zero unless used otherwise locally.
    Abbreviations:
     "W"      = Double_Wide field       SHALL be 0
     "M_Type" = Message_Type field      SHALL be set according to
                                              HIPPI-LE
     "D_A_T"  = Destination_Address_Type SHALL be 2
```

```
        "S_A_T" = Source_Address_Type      SHALL be 2
        [FILL] bytes complete the HIPPI message to an even
        number of 32 bit words.  The number of fill bytes
        is not counted in the data length.
```

6.2 HIPPI Hardware Address Formats and Requirements

   For HIPPI-800, the Hardware Address is a 10-byte unit that SHALL
   contain the Switch Address AND the ULA. The format of a hardware
   address is:

```
31                  23                  15            7             0
+----------------+----------------+-------+-------+----------------+
|   Mode Byte    |      00        |   0   |   X   |      XX        |
+----------------+----------------+-------+-------+----------------+
|   ULA byte 0   |   ULA byte 1   |  ULA byte 2   |   ULA byte 3   |
+----------------+----------------+-------+-------+----------------+
|   ULA byte 4   |   ULA byte 5   |
+----------------+----------------+
```

   Where "XXX" is the 12 bit HIPPI logical address defined in HIPPI-SC
   [4]. Details on ULA see next section.

   Two switch addresses are considered to be the same when they have the
   same 12 bit destination HIPPI logical address.

   NOTE: In the case of HIPPI-6400, the hardware address is ONLY the 6-
   byte ULA. Therefore the length of the hardware address clearly
   defines which version of HIPPI is being used.

6.2.1 48-bit Universal LAN MAC Addresses

   IEEE Standard 802.1A [11] specifies the Universal LAN MAC Address
   format. The globally unique part of the 48-bit space is administered
   by the IEEE. Each port on a HIPPI-SC LAN SHOULD be assigned a ULA.
   Multiple ULAs may be used if a port contains more than one IEEE 802.2
   LLC protocol entity.

   The format of the HIPPI hardware address within its HARP message
   follows IEEE 802.1A canonical bit order and HIPPI-FP bit and byte
   order. For example the requester's ULA part of the HIPPI hardware
   address would decompose to:

```
   31              23              15               7               0
   +---------------+---------------+---------------+---------------+
   |ULA byte 0 |L|G|  ULA byte 1   |  ULA byte 2   |  ULA byte 3   |
   +---------------+---------------+---------------+---------------+
   |   ULA byte 4  |  ULA byte 5   |
   +---------------+---------------+
```
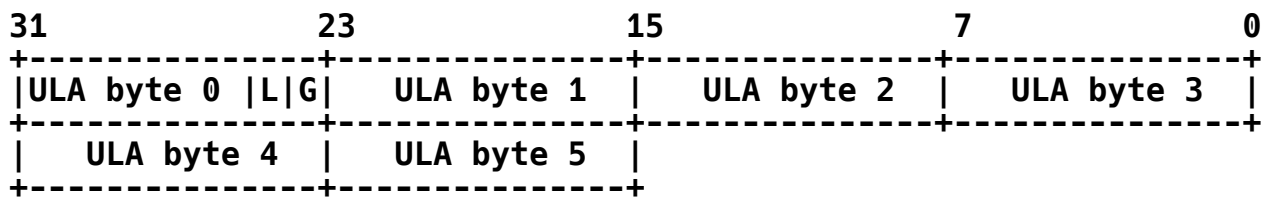
                 Universal LAN MAC Address Format

     L (U/L bit) = 1 for Locally administered addresses,
                   0 for Universal.
     G (I/G bit) = 1 for Group addresses,
                   0 for Individual.

   The use of ULAs is OPTIONAL, but RECOMMENDED. The use of ULAs is
   REQUIRED if a port wishes to interoperate with a conventional
   network.

   ULAs may also be used by bridging devices that replace HIPPI hardware
   headers with the MAC headers of other LANs.

6.3 HARP and InHARP Message Formats

   The HARP protocols use the HIPARP hardware type (ar$hrd) [16],
   protocol type (ar$pro), and operation code (ar$op) data formats as
   the ARP, and InARP protocols [15,7]. In addition, HARP makes use of
   an additional operation code for ARP_NAK introduced with [12]. The
   remainder of the HARP/InHARP message format is different than the
   ARP/InARP message format defined in [15,7,10] and it is also
   different from the format defined in the first "IP and ARP on HIPPI"
   RFC-1374 [14].

   HARP messages SHALL be transmitted with the HIPARP hardware type code
   of 28 (decimal). Furthermore, HARP messages SHALL be accepted if
   received with hardware type codes of either 28, 1 or 6 (decimal).

   The HARP message has several fields that have the following format
   and values:

   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
```

   Where :
     ar$hrd - SHALL contain 28. (HIPARP)

     ar$pro - SHALL contain the IP protocol code 2048 (decimal).

     ar$op   - SHALL contain the operational value (decimal):
                  1 for   HARP_REQUESTs
                  2 for   HARP_REPLYs
                  8 for InHARP_REQUESTs
                  9 for InHARP_REPLYs
                  10 for   HARP_NAK

     ar$pln - SHALL contain 4.

     ar$rln - SHALL contain 10 IF this is a HIPPI-800 HW address
               ELSE, for HIPPI-6400, it SHALL contain 6.

     ar$thl - SHALL contain 10 IF this is a HIPPI-800 HW address
               ELSE, for HIPPI-6400, it SHALL contain 6.

     ar$rha - in requests and NAKs it SHALL contain the requester's
               HW address. In replies it SHALL contain the target
               port's HW address.

     ar$rpa - in requests and NAKs it SHALL contain the requester's IP
               address if known, otherwise zero.
               In other replies it SHALL contain the target
               port's IP address.

     ar$tha - in requests and NAKs it SHALL contain the target's
               HW address if known, otherwise zero.
               In other replies it SHALL contain the requester's
               HW addressA.

     ar$tpa - in requests and NAKs it SHALL contain the
               target's IP address if known, otherwise zero.
               In other replies it SHALL contain the requester's
               IP address.

   The format of the six bytes of the ULA SHALL be the same as required
   in the HIPPI-LE header (see section 6.2), except for the alignment of
   the ULAs with respect to the 32-bit HIPPI word, which is different
   between ARP and HIPPI-LE. No bit reversal is necessary as is
   required with FDDI.

```
       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 ULA                          |
       +=============+===============+===============+===============+
     8 |    AA       |     AA        |      03       |      00       |
       +-------------+---------------+---------------+---------------+
     9 |    00       |     00        |    Ethertype (2054)           |
       +-------------+---------------+---------------+---------------+
    10 |        hrd (28)             |        pro (2048)             |
       +-------------+---------------+---------------+---------------+
    11 |        op (ar$op)           |    pln (6)    |    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 - q      | Tgt 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
```

6.3.1 Example Message encodings:

    HARP_REQUEST message
        HARP ar$op   = 1 (HARP_REQUEST)
        HARP ar$rpa = IPy   HARP ar$tpa = IPa
        HARP ar$rha = SWy ULAy   HARP ar$tha = 0 **
        ** is what we would like to find out

    HARP_REPLY message format
        HARP ar$op   = 2 (HARP_REPLY)
        HARP ar$rpa = IPa   HARP ar$tpa = IPy
        HARP ar$rha = SWa ULAa *   HARP ar$tha = SWy ULAy
        * answer we were looking for

    InHARP_REQUEST message format
        HARP ar$op   = 8 (InHARP_REQUEST)
        HARP ar$rpa = IPy   HARP ar$tpa   = 0 **
        HARP ar$rha = SWy ULAy   HARP ar$tha   = SWa ULAa
        ** is what we would like to find out

    InHARP_REPLY message format
        HARP ar$op   = 9 (InHARP_REPLY)
        HARP ar$rpa = IPs *   HARP ar$tpa   = IPy
        HARP ar$rha = SWa ULAa   HARP ar$tha   = SWy ULAy
        * answer we were looking for

6.3.2 HARP_NAK message format

    The HARP_NAK 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_NAK value.
    HARP makes use of an additional operation code for HARP_NAK. Hence,
    HARP_NAK MUST be implemented.

6.3.3 Combined HIPPI-LE and HARP message addresses

    The combined HIPPI-LE/HARP message contains ten addresses, two for
    the destination and two for the source of the message, three for the
    requester and three for the target:

        Destination Switch Address (HIPPI-LE)
        Destination ULA   (HIPPI_LE)

        Source Switch Address   (HIPPI-LE)
        Source ULA   (HIPPI-LE)

```
    Requester IP Address        (HARP)
    Requester ULA               (HARP)
    Requester Switch Address    (HARP)

    Target IP Address           (HARP)
    Target ULA                  (HARP)
    Target Switch Address       (HARP)
```

Examples:

The following relations are true for a HARP_REQUEST and
InHARP_REQUESTs.

```
    LIS without broadcast -  Dest SW Addr   = HARP server SW Addr
    (with HARP server)       Dest ULA       = HARP server ULA
                             Source SW Addr = Requester's SW Addr
                             Source ULA     = Requester's ULA
```

7 Broadcast and Multicast

HIPPI-SC does not require switches to support broadcast. Broadcast
support has therefore been absent from many HIPPI networks.

During its registration phase, every port, including HARP server(s),
discover if the underlying medium is capable of broadcast (see
section 5.1.2). Should this not be the case, then the HARP server(s)
MUST emulate broadcast through an IP broadcast emulation server.

A HIPPI IP broadcast server (PIBES) is an extension to the HARP
server and only makes sense when the LIS does not inherently support
broadcast. The PIBES allows common upper layer networking protocols
(RIP, TCP, UDP, etc.) to access IP LIS broadcast.

7.1 Protocol for an IP Broadcast Emulation Server - PIBES

To emulate broadcast within an LIS, a PIBES SHALL use the currently
valid HARP table of the HARP server as a list of addresses called the
target list. The broadcast server SHALL validate that all incoming
messages have a source address which corresponds to an address in the
target list. Only messages addressed to the IP LIS broadcast
addresses, multicast address or 255.255.255.255 are considered valid
messages for broadcasting. Invalid messages MUST be dropped. All
valid incoming messages shall be forwarded to all addresses in the
target list.

It is RECOMMENDED that the broadcast server run on the same port as
the HARP server since this memo does not define the protocol for
exchanging the valid HARP table. The default address to use for the
broadcast address is the operational HARP server address.

7.2 IP Broadcast Address

This memo only defines IP broadcast. It is independent of the
underlying hardware addressing and broadcast capabilities. Any port
can differentiate between IP traffic directed to itself and a
broadcast message sent to it by looking at the IP address. All IP
broadcast messages SHALL use the IP LIS broadcast address or.

It is RECOMMENDED that the PIBES run on the same port as the HARP
server. In that case, the PIBES SHALL use the same address as the
HARP server.

7.3 IP Multicast Address

HIPPI does not directly support multicast address, therefore there
are no mappings available from IP multicast addresses to HIPPI
multicast services. Current IP multicast implementations (i.e. MBONE
and IP tunneling, see [9]) will continue to operate over HIPPI-based
logical IP subnets if all IP multicast packets are sent using the
same algorithm as if the packet were being sent to 255.255.255.255.

7.4 A Note on Broadcast Emulation Performance

It is obvious that a broadcast emulation service (as defined in
section 7.1) has an inherent performance limit. In an LIS with n
ports, the upper bound on the bandwidth that such a service can
broadcast is:

$$\text{(total bandwidth)}/(n+1)$$

since each message must first enter the broadcast server, accounting
for the additional 1, and then be sent to all n ports. The broadcast
server could forward the message destined to the port on which it
runs internally, thus reducing (n+1) to (n) in a first optimization.

This service is adequate for the standard networking protocols such
as RIP, OSPF, NIS, etc. since they usually use a small fraction of
the network bandwidth for broadcast. For these purposes, the
broadcast emulation server as defined in this memo allows the HIPPI
network to look similar to an Ethernet network to the higher layers.

It is further obvious that such an emulation cannot be used to
broadcast high bandwidth traffic. For such a solution, hardware
support for true broadcast is required.

8 HARP for Scheduled Transfer Protocol[17]

   This RFC also applies for resolving addresses used with Scheduled
   Transfer (STP) over HIPPI-800 instead of IP. This RFC's message
   types and algorithms can be used for STP (since STP uses Internet
   Addresses) as long as there is also an IP over HIPPI implementation
   on all of the ports.

9 Discovery of One's Own Switch Address

   This HARP specification assumes that each port has prior knowledge of
   its own hardware address. This address may be manually configured,
   by means outside the scope of this memo or a port may discover its
   own logical address through the algorithm described below.

   Ports are NOT REQUIRED to implement this switch address discovery
   protocol but are encouraged to do so since it reduces the
   administrative overhead. The algorithm presented in this section is
   based on John Renwick's work as detailed in RFC-1374 [14]. The
   concept of the discovery process is to scan all possible switch
   addresses. The messages that are received will be the ones containing
   one of our switch addresses.

   If a port implements this algorithm it SHALL form a HIPPI-LE message
   as defined in HIPPI-LE: containing an Self_Address_Resolution_Request
   (see [3]) PDU Type, a Source_IEEE_Address and
   Destination_IEEE_Address (set to the correct ULA for the sender), and
   the Source_Switch_Address and Destination_Switch_Address.

   This self address resolution message uses the same HIPPI-LE message
   format as described in HIPPI-SC and HIPPI-LE: the Self Address
   Resolution Request PDU and Self Address Resolution Response PDU type
   codes and no piggybacked ULP data. The HIPPI-LE header contents for
   the request are:

       HIPPI-LE Message_Type is   = 3, Self Addr. Resolution Request
       HIPPI-LE Destination_Address_Type   = 0 (undefined)
       HIPPI-LE Destination_Switch_Address = X (X element scan range)
       HIPPI-LE Source_Address_Type   = 0 (undefined)
       HIPPI-LE Source_Switch_Address   = 0 (unknown)
       HIPPI-LE Destination_IEEE_Address   = 0
       HIPPI-LE Source_IEEE_Address   = my ULA

   There is no D2 data; the message contains only the HIPPI-FP header
   and D1_Area with the HIPPI-LE header.

Ports SHALL start the scan with a configurable logical address
(default 0x000) and increment the value for by one for each
subsequent try. The port SHALL continue until it sees its own self
address resolution request or it has reached the end, which may be
another configurable value (default 0xFFF). It is RECOMMENDED that
the range of addresses to scan be configurable since some networks
have equipment that does not gracefully handle HIPPI-LE messages.

After a port sends the[se] request[s], two positive outcomes are
possible:

o   the port receives its own request(s), and obtains one of its own
    Switch Address, or

o   the port receives an AR_S_Response with the
    Destination_Switch_Address filled in.

10 Security Considerations

HARP messages are not authenticated which is a potentially flaw that
could allow corrupt information to be introduced into the server
system.

There are other known security issues relating to port impersonation
via the address resolution protocols used in the Internet [8]. No
special security mechanisms have been added to the address resolution
mechanism defined here for use with networks using HARP.

Not all of the security issues relating to ARP over HIPPI are clearly
understood at this time. However, given the security hole ARP allows,
other concerns are probably minor.

11 Open Issues

Synchronization and coordination of multiple HARP servers and
multiple broadcast servers are left for further study.

12 HARP Examples

Assume a HIPPI-SC switch is installed with three connected ports: x,
y, and a. Each port has a unique hardware address that consists of
Switch Address (e.g. SWx, SWy, SWa) and unique ULA (ULAx, ULAy and
ULAa, respectively). There is a HARP server connected to a switch
port that is mapped to the address HWa (SWa, ULAa), this address is
the authoritative HIPPI hardware address in the HRAL (HARP Request
Address List).

The HARP server's table is empty. Ports X and Y each know their own
hardware address. Eventually they want to talk to each other; each
knows the other's IP address (from the port database) but neither
knows the other's ULA or Switch Address. Both ports X and Y have
their interfaces configured DOWN.

NOTE: The LLC, SNAP, Ethertype, HIPPI-LE Message Type, ar$hrd,
arpro, arpln fields are left out from the examples below since they
are constant. Likewise, ar$rhl = ar$thl = 9 are omitted since these
are all HIPPI-800 examples.

12.1 Registration Phase of Client Y on Non-broadcast Hardware

Port Y starts: its HARP table entry state for the server: PENDING

1. Port Y initiates its interface and sends an InHARP_REQUEST to HWa
   after starting a table entry for HWa.

   ```
   HIPPI-LE Destination_Switch_Address = SWa
   HIPPI-LE Source_Switch_Address      = SWy
   HIPPI-LE Destination_IEEE_Address   = ULAa
   HIPPI-LE Source_IEEE_Address        = ULAy
   HARP ar$op                          = 8 (InHARP_REQUEST)
   HARP ar$rpa                         = IPy
   HARP ar$tpa                         = 0 **
   HARP ar$rha                         = SWy ULAy
   HARP ar$tha                         = SWa ULAa
   ** is what we would like to find out
   ```

2. HARP server receives Y's InHARP_REQUEST, it examines the source
   addresses and scans its tables for a match. Since this is the
   first time Y connects to this server there is no entry and one
   will be created and time stamped with the information from the
   InHARP_REQUEST. The HARP server will then send a InHARP_REPLY
   including its IP address.

   ```
   HIPPI-LE Destination_Switch_Address = SWy
   HIPPI-LE Source_Switch_Address      = SWa
   HIPPI-LE Destination_IEEE_Address   = ULAy
   HIPPI-LE Source_IEEE_Address        = ULAa
   HARP ar$op                          = 9 (InHARP_REPLY)
   HARP ar$rpa                         = IPs *
   HARP ar$tpa                         = IPy
   HARP ar$rha                         = SWa ULAa
   HARP ar$tha                         = SWy ULAy
   * answer we were looking for
   ```

   3. Port Y examines the incoming InHARP_REPLY, completes its table
      entry for the HARP server. The client's HARP table entry for the
      server now passes into the VALID state and is usable for regular
      HARP traffic. Receiving this reply ensures that the HARP server
      has properly registered the client.

12.2 Registration Phase of Client Y on Broadcast Capable Hardware

   If there is a broadcast capable network then the authoritative
   address in the HRAL would be mapped to the broadcast address, HWb =
   SWb, ULAb (likely 0xFE1 and FF:FF:FF:FF:FF:FF).

   Port Y starts: its HARP table entry state for HWa: PENDING

   1. Port Y initiates its interface and sends an InHARP_REQUEST to HWa,
      in this example the broadcast address, after starting a table
      entry.

      HIPPI-LE Destination_Switch_Address = SWb
      HIPPI-LE Source_Switch_Address   = SWy
      HIPPI-LE Destination_IEEE_Address   = ULAb
      HIPPI-LE Source_IEEE_Address   = ULAy
      HARP ar$op   = 8 (InHARP_REQUEST)
      HARP ar$rpa   = IPy
      HARP ar$tpa   = 0 **
      HARP ar$rha   = SWy ULAy
      HARP ar$tha   = SWb ULAb
      ** is what we would like to find out

   2. Since the network is a broadcast network, client Y will receive a
      copy of its InHARP_REQUEST. Client Y examines the source
      addresses. Since they are the same as what Y filled in the
      InHARP_REQUEST, Y can deduce that it is connected to a broadcast
      medium. Port Y completes its table entry for HWa. This entry will
      not timeout since it is considered unlikely for a particular
      underlying hardware type to change between broadcast and non-
      broadcast; therefore this mapping will never change.

12.3 Operational Phase (phase II)

   The Operational Phase of the HARP protocol as specified in this memo
   is the same for both broadcast and non-broadcast capable HIPPI
   hardware. The authoritative address in the HRAL for this example will
   be HWa: <SWa, ULAa> and IPs for simplicity reasons.

12.3.1 Standard successful HARP_Resolve example

   Assume the same process (steps 1-3 of section 10.1) happened for port
   X. Then the state of X and Y's tables is: the HARP server table entry
   is in the VALID state. So lets look at the message traffic when X
   tries to send a message to Y. Since X doesn't have an entry for Y,

   1. Port X connects to the authoritative address of the HRAL and sends
      a HARP_REQUEST for Y's hardware address:

      HIPPI-LE Destination_Switch_Address = SWa
      HIPPI-LE Source_Switch_Address   = SWx
      HIPPI-LE Destination_IEEE_Address   = ULAa
      HIPPI-LE Source_IEEE_Address   = ULAx
      HARP ar$op   = 1 (HARP_REQUEST)
      HARP ar$rpa   = IPx
      HARP ar$tpa   = IPy
      HARP ar$rha   = SWx ULAx
      HARP ar$tha   = 0 **
      ** is what we would like to find out

   2. The HARP server receives the HARP request and updates its entry
      for X if necessary. It then generates a HARP_REPLY with Y's
      hardware address information.

      HIPPI-LE Destination_Switch_Address = SWx
      HIPPI-LE Source_Switch_Address   = SWa
      HIPPI-LE Destination_IEEE_Address   = ULAx
      HIPPI-LE Source_IEEE_Address   = ULAa
      HARP ar$op   = 2 (HARP_Reply)
      HARP ar$rpa   = IPy
      HARP ar$tpa   = IPx
      HARP ar$rha   = SWy ULAy *
      HARP ar$tha   = SWx ULAx
      * answer we were looking for

   3. Port X connects to port Y and transmits an IP message with the
      following information in the HIPPI-LE header:

      HIPPI-LE Destination_Switch_Address = SWy
      HIPPI-LE Source_Switch_Address   = SWx
      HIPPI-LE Destination_IEEE_Address   = ULAy
      HIPPI-LE Source_IEEE_Address   = ULAx

   If there had been a broadcast capable HIPPI network, the target ports
   would themselves have received the HARP_REQUEST of step 2 above and
   responded to them in the same way the HARP server did.

12.3.2 Standard non-successful HARP_Resolve example

    Like in 12.3.1, assume that X and Y are fully registered with the
    HARP server. Then the state of X and Y's HARP server table entry is:
    VALID. So lets look at the message traffic when X tries to send a
    message to Q. Further assume that interface Q is NOT configured UP,
    i.e. it is DOWN. Since X doesn't have an entry for Q,

    1. Port X connects to the HARP server switch address and sends a
       HARP_REQUEST for Q's hardware address:

       HIPPI-LE Destination_Switch_Address = SWa
       HIPPI-LE Source_Switch_Address   = SWx
       HIPPI-LE Destination_IEEE_Address   = ULAa
       HIPPI-LE Source_IEEE_Address   = ULAx
       HARP ar$op   = 1   (HARP_REQUEST)
       HARP ar$rpa   = IPx
       HARP ar$tpa   = IPq
       HARP ar$rha   = SWx ULAx
       HARP ar$tha   = 0 **
       ** is what we would like to find out

    2. The HARP server receives the HARP request and updates its entry
       for X if necessary. It then looks up IPq in its tables and doesn't
       find it. The HARP server then generates a HARP_NAK reply message.

       HIPPI-LE Destination_Switch_Address = SWx
       HIPPI-LE Source_Switch_Address   = SWa
       HIPPI-LE Destination_IEEE_Address   = ULAx
       HIPPI-LE Source_IEEE_Address   = ULAa
       HARP ar$op   = 10 (HARP_NAK)
       HARP ar$rpa   = IPx
       HARP ar$tpa   = IPq
       HARP ar$rha   = SWx ULAx
       HARP ar$tha   = 0 ***
       *** No Answer, and notice that the fields do not get swapped,
           i.e. the HARP message is the same as the HARP_REQUEST
           except for the operation code.

    If there had been a broadcast capable HIPPI network, then there would
    not have been a reply.

13 References

[1]   ANSI X3.183-1991(R1996), Information Technology - High-
      Performance Parallel Interface - Mechanical, Electrical and
      Signaling Protocol Specification; (HIPPI-PH).

[2]   ANSI X3.210-1998, Information Technology - High-Performance
      Parallel Interface - Framing Protocol; (HIPPI-FP).

[3]   ANSI X3.218-1993, Information Technology - High-Performance
      Parallel Interface - Encapsulation of ISO 8802-2 (IEEE Std
      802.2) Logical Link Control Protocol Data Units; (HIPPI-LE).

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

[5]   ANSI X3.300-1997, Information Technology - High-Performance
      Parallel Interface - Serial Specification; (HIPPI-Serial).

[6]   Braden, R., "Requirements for Internet Hosts -- Communication
      Layers", STD 3, RFC 1122, October 1989.

[7]   Bradely, T. and C. Brown, "Inverse Address Resolution Protocol",
      RFC 2390, September 1998.

[8]   Bellovin, Steven M., "Security Problems in the TCP/IP Protocol
      Suite", ACM Computer Communications Review, Vol. 19, Issue 2,
      pp. 32-48, 1989.

[9]   Deering, S, "Host Extensions for IP Multicasting", STD 5, RFC
      1112, August 1989.

[10] Finlayson, R., Mann, T., Mogul, J. and M. Theimer, "A Reverse
      Address Resolution Protocol", RFC 903, June 1984.

[11] ANSI/IEEE Std. 802.2-1989, Information Processing Systems -
      Local Area Networks - Logical Link Control, "IEEE Standards for
      Local Area Networks: Logical Link Control", IEEE, New York, New
      York, 1985.

[12] Laubach, Mark., "Classical IP and ARP over ATM", RFC 2225, April
      1998.

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

    [14] Renwick, J. and A. Nicholson, "IP and ARP on HIPPI", RFC 1374,
         October 1992.

    [15] Renwick, J., "IP over HIPPI", RFC 2067, January 1997.

    [16] Reynolds, J. and J. Postel, "Assigned Numbers", STD 2, RFC 1700,
         October 1994.

    [17] ANSI NCITS xxx.199x, Project 1245-D, Scheduled Transfer Protocol
         ANSI NCITS, Scheduled Transfer Protocol draft standard.

    [18] Bradner, S., "Key words for use in RFCs to Indicate Requirement
         Levels", BCP 14, RFC 2119, March 1997.

14 Acknowledgments

   This memo could not have come into being without the critical review
   from Greg Chesson, Carlin Otto, the high performance interconnect
   group of Silicon Graphics (specifically Jim Pinkerton, Brad Strand
   and Jeff Young) and the expertise of the ANSI T11.1 Task Group
   responsible for the HIPPI standards work.

   This memo is based on the second part of [14], written by John
   Renwick. ARP [13] written by Dave Plummer and Inverse ARP [7] written
   by T. Bradley and C. Brown provide the fundamental algorithms of HARP
   as presented in this memo. Further, the HARP server is based on
   concepts and models presented in [12], written by Mark Laubach who
   laid the structural groundwork for the HARP server.

15 Changes from RFC-1374 [14]

   RFC-2067 obsoletes RFC-1374 but left ARP outside of its scope because
   there was not enough implementation experience. This memo is an
   effort to clarify and expand the definition of ARP over HIPPI as
   found in RFC-1374 such that implementations will be more readily
   possible, especially considering forward interoperability with
   HIPPI-6400.

   The changes from RFC-1374 [14] are:

   o A new message format to acknowledge the HIPPI hardware address
      format and to eliminate the requirement of HIPPI-LE ARP for HARP
      to function.

   o Explicit registration phase.

   o Additional message formats: InHARP requests and replies as well as
      HARP_NAKs.

   o Details about the IP subnetwork configuration.

   o Details about table aging.

   o IP broadcast emulation.

16 Author's Address

   Jean-Michel Pittet
   Silicon Graphics Inc
   1600 Amphitheatre Parkway
   Mountain View, CA 94043

   Phone: 650-933-6149
   Fax: 650-933-3542
   EMail: jmp@sgi.com, jmp@acm.org

17 Full Copyright Statement

Acknowledgement