Network Working Group
Request for Comments: 2835
Category: Standards Track

J.-M. Pittet
Silicon Graphics Inc.

May 2000

IP and ARP over HIPPI-6400 (GSN)

Status of this Memo

Copyright Notice

Abstract

The ANSI T11.1 task force has standardized HIPPI-6400 also known as Gigabyte System Network (GSN), a physical-level, point-to-point, full-duplex, link interface for reliable, flow-controlled, transmission of user data at 6400 Mbit/s, per direction. A parallel copper cable interface for distances of up to 40 m is specified in HIPPI-6400-PH [1]. Connections to a longer-distance optical interface are standardized in HIPPI-6400-OPT [3].

HIPPI-6400-PH [1] defines the encapsulation of IEEE 802.2 LLC PDUs [10] and by implication, IP on GSN. Another T11.1 standard describes the operation of HIPPI-6400 physical switches HIPPI-6400-SC [2]. T11.1 chose to leave HIPPI-6400 networking issues largely outside the scope of their standards; this document specifies the use of HIPPI-6400 switches as IP local area networks. This document further specifies a method for resolving IP addresses to HIPPI-6400 hardware addresses (HARP) and for emulating IP broadcast in a logical IP subnet (LIS) as a direct extension of HARP. Furthermore it is the goal of this memo to define a IP and HARP that will allow interoperability for HIPPI-800 and HIPPI-6400 equipment both broadcast and non-broadcast capable networks.

Table of Contents

1. Introduction

HIPPI-6400 is a duplex data channel that can transmit and receive data simultaneously at nearly 6400 megabits per second. HIPPI-6400 data transfers are segmented into micropackets, each composed of 32 data bytes and 8 control bytes. HIPPI-6400 uses four multiplexed virtual channels. These virtual channels are allocated to control traffic, low latency traffic, and bulk traffic (see [1] for more details).

Using small packets and four virtual channels, large file transfers cannot lock out a host or switch port for interactive traffic. HIPPI-6400 guarantees in order delivery of data. It also supports link-level and end-to-end checksumming and credit-based flow control.

HIPPI-6400-PH defines a 20-bit interface for copper cables operating at 500 MBaud. This provides a user payload bandwidth of 6400 Mb/s (800,000,000 Bytes/sec) in each direction. [8]

HIPPI-6400-SC [2] defines two types of switches: bridging and non-bridging. The bridging switches are required to support hardware broadcast. Non-bridging switches are not required to support broadcast. This memo allows for a coherent implementation of IP and HARP with both types of switches.

Gigabyte System Network(TM) (GSN) is a marketing name for HIPPI-6400. It is a trademark of the High Performance Networking Forum (HNF; http://www.hnf.org) for use by its member companies that supply products complying to ANSI HIPPI-6400 standards.

2 Definitions

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 [19].

2.1 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. This document will use HIPPI-800 and HIPPI-6400 when referring to concepts that apply to one or the other technology. The term HIPPI will be used when referring to both technologies.

Values are decimal unless otherwise noted. Formatting follows IEEE 802.1A canonical bit order and HIPPI-6400-PH bit and byte ordering.

2.2 Glossary

Broadcast

A distribution mode which transmits a message to all ports. The sending port is part of "all" and will therefore also receive a copy of the sent message.

Classical/Conventional

Both terms are used with respect to networks, including 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 (HIPPI Address Resolution Protocol describes the whole set of HIPPI-6400 address resolution encodings and algorithms defined in this memo. HARP is a combination and adaptation of the Internet Address Resolution Protocol (ARP) RFC-826 [14] and Inverse ARP (InARP) [5] (see section 5). HARP also describes the HIPPI (800 and 6400) specific version of ARP (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.

HRAL

The HARP Request Address List. A list of ULAs to which HARP messages are sent when resolving names to addresses (see section 3.2).

Hardware (HW) address

The hardware address of a port; it consists of an ULA (see section 4.2). 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. A port may be an Internet host, bridge, router, or gateway.

PIBES

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

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.

3. IP Subnetwork Configuration

3.1 Background

ARP (address resolution protocol) as defined in [14] 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-6400 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-6400 port attached to the HIPPI-6400 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-6400 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-6400 connection between the two IP members over the HIPPI-6400 network. This is an consequence of using IP and choosing to have multiple LIS's on the same HIPPI-6400 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.

3.2 HIPPI LIS Requirements

The requirement for IP members (hosts, routers) operating in a HIPPI-6400 LIS configuration is:

- All members of the LIS SHALL have the same IP network/subnet address and address mask [4].

The following list identifies the set of HIPPI-6400-specific parameters that MUST be implemented in each IP station connected to the HIPPI-6400 network:

- HIPPI-6400 Hardware Address:

  The HIPPI-6400 hardware address (a ULA) of an individual IP endpoint (i.e. a network adapter within a host) MUST be unique in the LIS.

- 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. FF:FF:FF:FF:FF:FF.

The second address MUST be the standard HW address for the HARP server 00:10:3B:FF:FF:E0.

Therefore, the HRAL entries are sorted in the following order:

1st	: broadcast address	(FF:FF:FF:FF:FF:FF)	REQUIRED
2nd	: official HARP server address	(00:10:3B:FF:FF:E0)	REQUIRED
3rd & on:	any additional HARP server addresses will be sorted in decreasing order.		OPTIONAL

Manual configuration of the addresses and address lists presented in this section is implementation dependent and beyond the scope of this memo. However, prior to use by any service or operation detailed in this memo, clients MUST have HRAL address(es) configured as appropriate for their LIS.

4. Internet Protocol

4.1 Packet format

The HIPPI-6400 packet format for Internet datagrams [15] shall conform to the HIPPI-6400-PH standard [1]. The length of a HIPPI-6400-PH packet, including headers and trailing fill, shall be a multiple of 32 bytes as required by HIPPI-6400-PH.

All IP Datagrams shall be carried on HIPPI-6400-PH Virtual Channel 1 (VC1). Since HIPPI-6400-PH has a 32-byte granularity, IP Datagrams MUST be padded to a 32-byte granularity prior to sending. Added padding is transparent to IP and is not reflected in the length field of the IP header.

D_ULA Destination ULA SHALL be the ULA of the destination port.

S_ULA Source ULA SHALL be the ULA of the requesting port.

M_len Set to the IEEE 802 packet (e.g. IP or HARP message) length + 8 Bytes to account for the LLC/SNAP header length. The HIPPI-6400-PH [1] length parameter shall not include the pad.

4.1.1 IEEE 802.2 LLC

The IEEE 802.2 LLC Header SHALL begin in the first byte after M_len.

The LLC values (in hexadecimal and decimal) SHALL be

SSAP	0xAA	170	(8 bits)
DSAP	0xAA	170	(8 bits)
CTL	0x03	3	(8 bits)

for a total length of 3 bytes. The 0x03 CTL value indicates the presence of a SNAP header.

4.1.2 SNAP

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

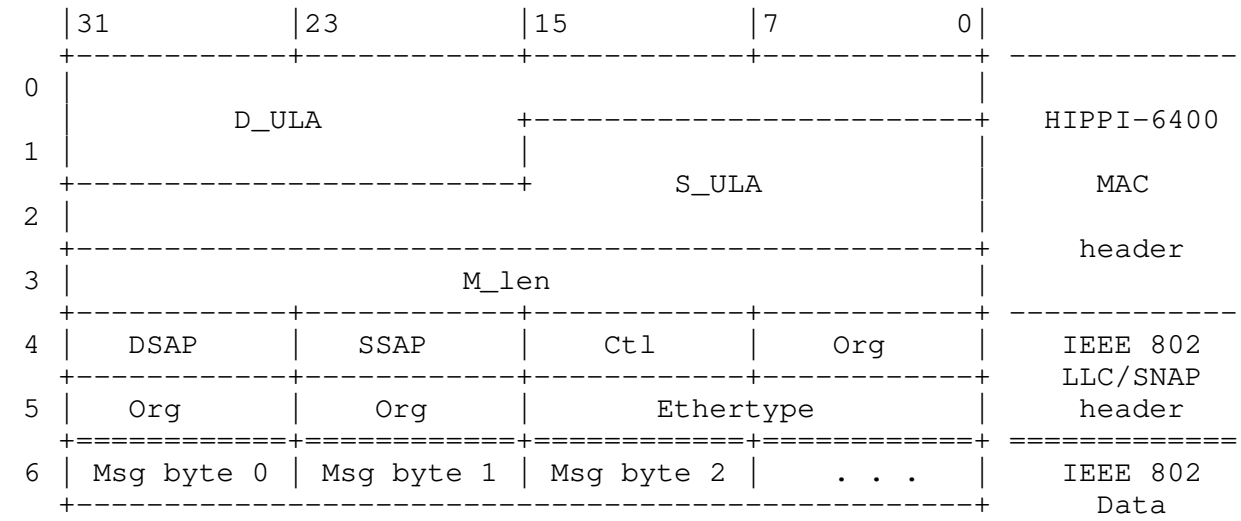
The Ethertype value SHALL be set as defined in Assigned Numbers [18]:

IP	0x0800	2048	(16 bits)
HARP = ARP =	0x0806	2054	(16 bits)

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

4.1.3 HIPPI-6400 802 Packet diagrams

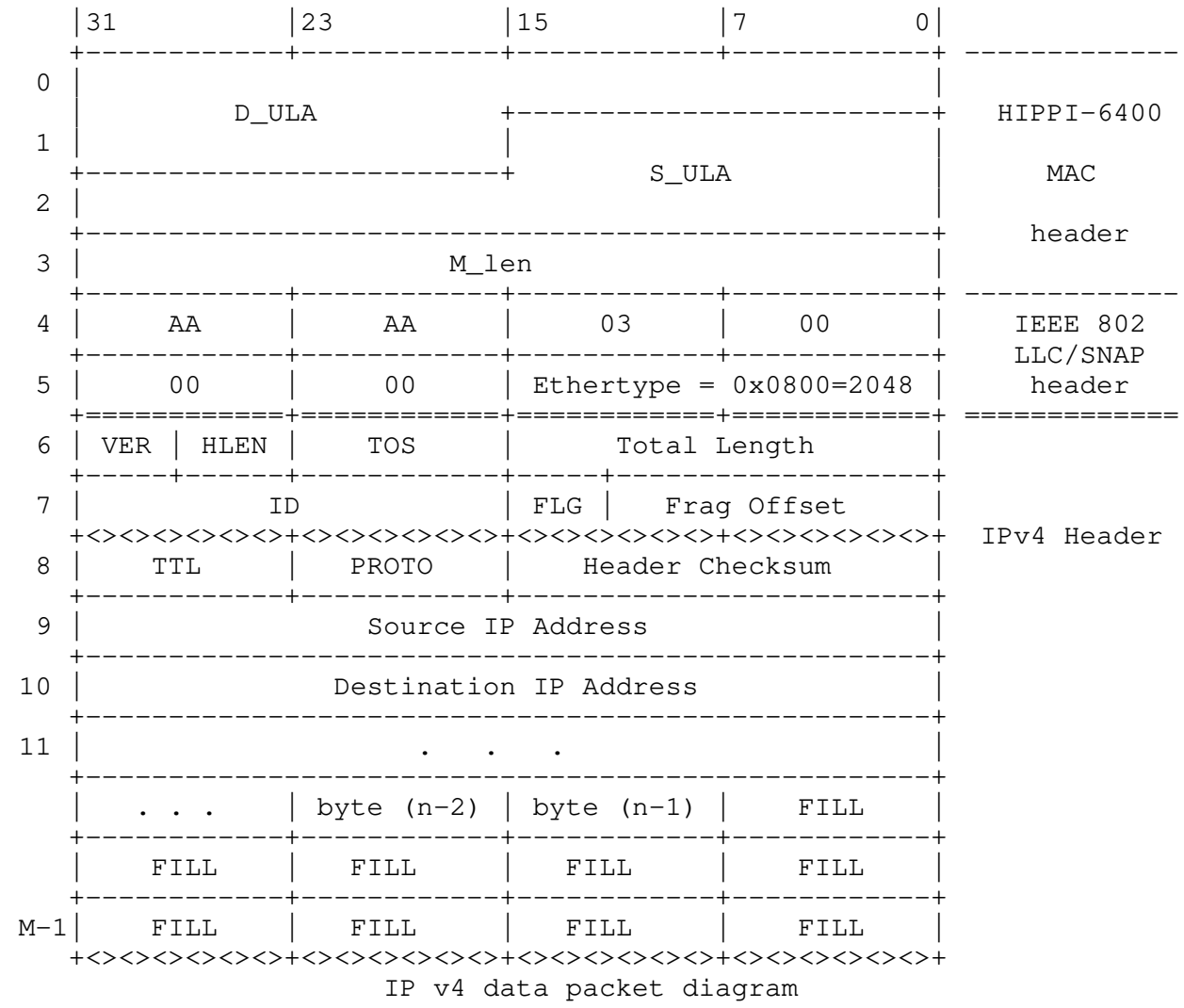
The following diagram shows a HIPPI-6400 message carrying IEEE 802 data.



Generic 802.1 data packet diagram

The following diagram shows an IP datagram of length n with the FILL bytes (value: 0x0). "<><>" indicates the micropacket separation. A HIPPI-6400-PH [1] micropacket is 32 bytes long.

All IP (v4) [15] packets will always span two or more micropackets. The first micropacket has a TYPE = header. The second and any further micropackets have a TYPE = Data (see [1]).



IP v4 data packet diagram

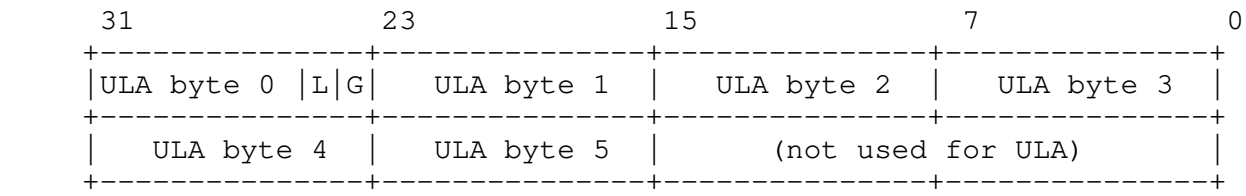
As shown in above figure the first eight bytes of the IP Datagram occupy the last eight bytes of the HIPPI-6400-PH [1] Header micropacket.

4.2 HIPPI-6400 Hardware address: Universal LAN MAC address (ULA)

HIPPI-6400 uses Universal LAN MAC Addresses specified in IEEE Standard 802.1A [10] or a subset as defined in HIPPI-6400-SC [2]. The globally unique part of the 48 bit space is administered by the IEEE. Each port on a HIPPI-6400-SC LAN MUST be assigned a ULA. Multiple ULAs may be used if a node contains more than one IEEE 802.2 LLC protocol entity.

This memo assumes the use of "Logical Addressing" as described in Annex A.2 of HIPPI-6400-SC[2].

The format of the address within its 48 bit HIPPI-6400-PH fields follows IEEE 802.1A canonical bit order and HIPPI-6400-PH bit and byte order:



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.

4.3 Maximum Transmission Unit - MTU

Maximum Transmission Unit (MTU) is defined as the maximum length of the IP packet, including IP header, but not including any overhead below IP, i.e., HIPPI-6400 MAC header and IEEE 802 LLC/SNAP header. Conventional LANs have MTU sizes determined by physical layer specification. MTUs may be required simply because the chosen medium won't work with larger packets, or they may serve to limit the amount of time a node must wait for an opportunity to send a packet.

HIPPI-6400-PH [1] limits packets to about 4 gigabytes (on VC 3) which imposes no practical limit for networking purposes. HIPPI-6400-PH VC 1, which was chosen for IP and ARP traffic, limits messages to about 128 Kbytes which is still larger than the HIPPI-800 MTU [17].

The MTU for HIPPI-6400 LANs SHALL be 65280 (decimal) bytes.

This value is backwards compatible with HIPPI-800. It allows the IP packet to fit in one 64K byte buffer with up to 256 bytes of overhead. The IP v4 overhead is 24 bytes for HIPPI-6400 and 40 bytes for HIPPI-800.

For HIPPI-6400 the byte accounting is:

HIPPI-6400-PH Header	16 bytes
IEEE 802.2 LLC/SNAP Headers	8 bytes
Maximum IP packet size (MTU)	65280 bytes
Unused expansion room	232 bytes
Total	65536 bytes (64K)

In contrast, the HIPPI-800 accounting is:

HIPPI-800-FP Header	8 bytes
HIPPI-800-LE Header	24 bytes
IEEE 802.2 LLC/SNAP Headers	8 bytes
Unused expansion room	216 bytes
Maximum IP packet size (MTU)	65280 bytes
Total	65536 bytes (64K)

5. HIPPI Address Resolution Protocol - HARP

Address resolution within the HIPPI-6400 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 [14] except the HIPPI-6400 specific packet format. Knowing the Internet address, conventional networks use ARP to discover another node'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 the same protocol as the original Inverse ARP (InARP) protocol presented in [5] except the HIPPI-6400 specific packet format. Knowing its hardware address, InARP is used to discover the other party's Internet address.

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

Internet addresses are assigned independent of ULAs. Before using HARP, each node MUST know its IP and its HW addresses. The ULA is optional but is RECOMMENDED if interoperability with conventional networks is desired.

If all switches in the LIS support broadcast, then the source address in the reply will be the target's source address. If all switches in the LIS do not support broadcast, then a HARP server MUST be used to provide the address resolution service, and the source address in the reply will be the HARP server's source address.

5.1 HARP Algorithm

This section defines the behavior and requirements for HARP implementations on both broadcast and non-broadcast capable HIPPI-6400-SC networks. HARP creates a table in each port which maps remote ports' IP addresses to ULAs, so that when an application requests a connection to a remote port by its IP address, the remote ULA can be determined, a correct HIPPI-6400-PH header can be built, and a connection to the port can be established using the ULA.

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. To select the authoritative HARP service, each port needs to determine if it is connected to a broadcast network. At each point in time there is only one authoritative HARP service.

The port SHALL send an InHARP_REQUEST to the first address in the HRAL (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 order of addresses in 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 HRAL addresses in order. The client SHALL terminate the registration phase and transition into the operational phase, when either: (1) it receives its own InHARP_REQUEST, or (2) when it receives an InHARP_REPLY from at least one of the HARP servers and it has determined the authoritative HARP service as described in 5.1.1.

When ports are initiated they send an InHARP_REQUEST to the authoritative HRAL address. The first address to be tried will be the broadcast address "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.

   The port SHALL choose the next address in the HRAL as a candidate for a HARP server and send an InHARP_REQUEST to that address: (00:10:3B:FF:FF:E0).

   The port SHALL continue to retry each non-broadcast HARP server address in the HRAL at least once every 5 seconds until one of the following termination criteria are met for each address.

   a. 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).

b. 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.

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-6400-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 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-6400-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 information about other IP members by sending of HARP_REQUESTs to the authoritative address in the HRAL and by receiving of 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 HIPPI-6400-SC switched 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, is in the ULA part of the Target Hardware Address field (ar$tha) of the message.

3. The port is a HARP server.

NOTE: It is REQUIRED to have a HARP server run on a port that has 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 [5].)

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 5.1.3 and 5.6). All InHARP_REPLY messages SHALL be used to build/refresh its client HARP table entries. (Refer to Section 7, "Protocol Operation" in [5].)

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 showed that the underlying network supports broadcast, then the refresh sequence is 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-6400 connections from other HIPPI-6400 ports. The HARP server expects an InHARP_REQUEST as the first message from the client. A server examines the IP address, the hardware address of the InHARP_REQUEST and adds or updates its HARP table entry <IP address(es), ULA> as well as the time stamp.

A HARP server replies to HARP_REQUESTs and InHARP_REQUESTs based on the information which it has in its table. The HARP server replies SHALL contain the hardware type and corresponding format of the request (see also sec. 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

The result will be a table with: IPb <-> HWa <-> IPb If IPb was the only IP address referred to by the HWb entry, then delete the HWb entry.

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 sect. 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.

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

- 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 the invalidated 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. 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 another type of IEEE 802 payload as described in section 4.1.3 above. The HIPPI-6400 HARP SHALL support two packet formats, both the generic Ethernet ARP packet and the HIPPI-800 HARP packet format defined in [13]. HARP messages SHALL be transmitted with a hardware type code of 28 on non-broadcast capable hardware or 1 in either case.

The ar$hrd field SHALL be used to differentiate between the two packet formats. The reply SHALL be in the format of the request.

6.1 Generic IEEE 802 ARP Message Format

This is the ARP packet format used by conventional IEEE 802 networks (i.e. Ethernet, etc). The packet format is described in RFC-826 [14] and is given here only for completeness purpose.

```
  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$op   16 bits  opcode (ares_op$REQUEST | ares_op$REPLY)
  ar$sha  48 bits  Hardware address of sender of this packet
  ar$spa  32 bits  Protocol address of sender of this packet
  ar$tha  48 bits  Hardware address of target of this
  ar$tpa  32 bits  Protocol address of target.
```

Where:

ar$hrd - SHALL contain 1. (Ethernet)

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

ar$hln - SHALL contain 6.

ar$pln - SHALL contain 4.

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

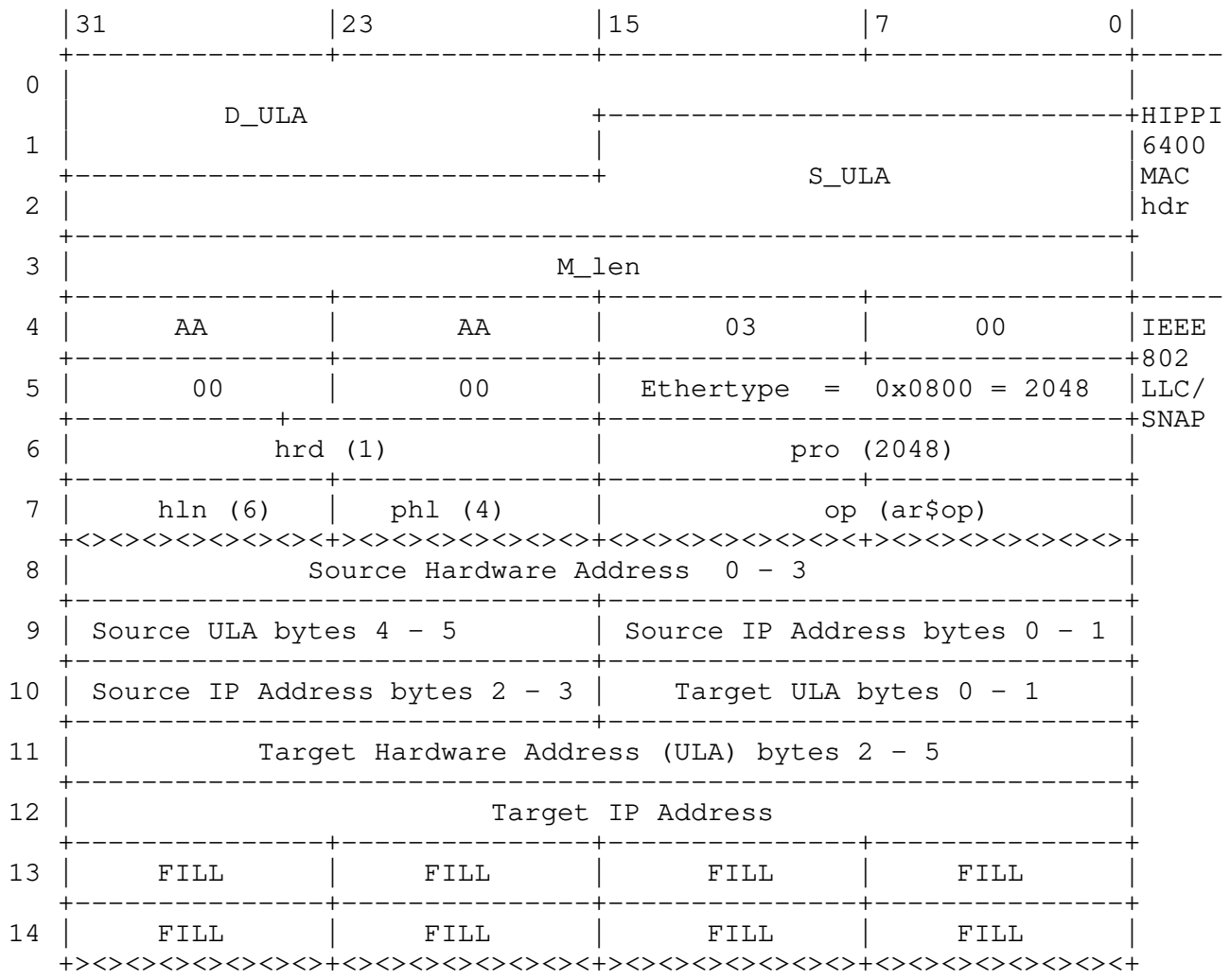
ar$rpa - in requests and NAKs it SHALL contain the requester's IP address if known, otherwise zero.
In other replies it SHALL contain the target port's IP address.

ar$sha - in requests and NAKs it SHALL contain the requester's ULA
In replies it SHALL contain the target port's ULA.

ar$spa - 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 ULA if known, otherwise zero.
In other replies it SHALL contain the requester's ULA.

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.



6.2 HIPARP Message Formats

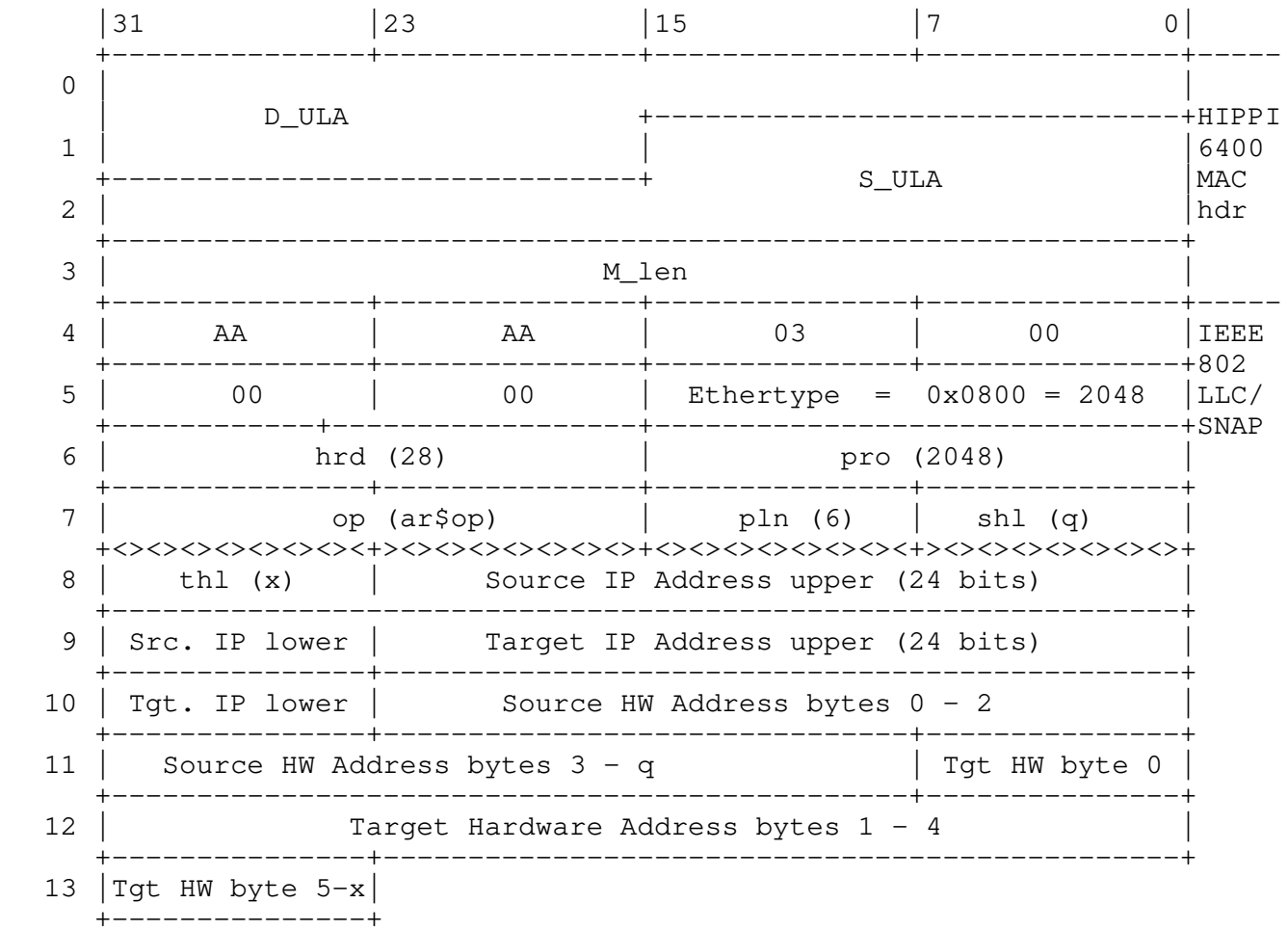
The HARP protocols further SHALL support the HIPARP hardware type (ar$hrd) = 28 (dec) [18], protocol type (ar$pro), and operation code (ar$op) data formats as the ARP, and InARP protocols [14,7]. In addition, HARP makes use of an additional operation code for ARP_NAK introduced with [11]. The remainder of the HIPARP message format (defined in [13]) is different than the ARP/InARP message format defined in [14,7,10] and it is also different from the format defined in the first "IP and ARP on HIPPI" RFC-1374 [16].

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 address.
```

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.

Payload Format for HARP/InHARP PDUs:



HARP - InHARP Message

6.2.1 Example Message encodings:

Assume for the following example that the HARP server is in the HIPPI-6400 side and the clients, X and Y are on the HIPPI-800 side of the non-broadcast capable network.

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

HARP_REPLY message format
      HARP ar$op   = 2 (HARP_REPLY)
      HARP ar$rpa  = IPx                HARP ar$tpa  = IPy
      HARP ar$rha  = SWx ULAx *         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   = SWx ULAx
      ** is what we would like to find out

InHARP_REPLY message format
      HARP ar$op    = 9 (InHARP_REPLY)
      HARP ar$rpa   = IPx *             HARP ar$tpa   = IPy
      HARP ar$rha   = SWx ULAx          HARP ar$tha   = SWy ULAy
      * answer we were looking for
```

6.2.2 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 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 and MUST be implemented.

7 Broadcast and Multicast

HIPPI-6400-SC requires compliant systems to support broadcast. Initial HIPPI-6400-SC systems MAY defer broadcast capability to a broadcast server rather than support it directly in the switching mechanism. A centralized HARP server architecture meets two of the three major duties of a broadcast server.

A central entity serving the whole LIS solves the coordination problem of a distributed approach. The registration requirement solves the second problem of determining which addresses make up the set loosely called "everyone". The last duty of a broadcast server is to replicate an incoming packet and send it to "everyone".

During its registration phase, every port , including HARP server(s), discover if the underlying medium is capable of broadcast (see section 5.1.1). 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.

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-6400 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 [7]) 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:

(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-6400 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

This RFC also applies for resolving addresses used with Scheduled Transfer (ST) over HIPPI-6400 instead of IP. This RFC's message types and algorithms can be used for ST (since ST uses Internet Addresses) as long as there is also an IP over HIPPI-6400 implementation on all the ports.

9 Security Consierations

There are known security issues relating to port impersonation via the address resolution protocols used in the Internet [6]. 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-6400 are clearly understood at this time, due to the fluid state of HIPPI-6400 specifications, newness of the technology, and other factors. However, given the security hole ARP allows, other concerns are probably minor.

10 Open Issues

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

11 HARP Examples

Assume a HIPPI-6400-SC switch is installed with three connected ports: x, y, and a. Each port has a unique hardware address that consists unique ULA (ULAx, ULAy and UlAa, respectively). There is a HARP server connected to a switch port that is mapped to the address HWa, 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. Both ports X and Y have their interfaces configured DOWN.

NOTE: The LLC, SNAP, Ethertype, arhrd, arpro, ar$pln fields are left out from the examples below since they are constant. As well as ar$rhl = ar$thl = 6 since these are all HIPPI-6400 examples.

11.1 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 the HWa after starting a table entry for the HWa.

```
HIPPI-6400-PH D_ULA                = ULAa
HIPPI-6400-PH S_ULA                = ULAy
HARP ar$op                         = 8 (InHARP_REQUEST)
HARP ar$rpa                        = IPy
HARP ar$tpa                        = 0 **
HARP ar$rha                        = ULAy
HARP ar$tha                        = 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-6400-PH D_ULA                 = ULAy
HIPPI-6400-PH S_ULA                 = ULAa
HARP ar$op                          = 9 (InHARP_REPLY)
HARP ar$rpa                         = IPs *
HARP ar$tpa                         = IPy
HARP ar$rha                         = ULAa
HARP ar$tha                         = ULAy
* answer we were looking for
```

3. Port Y examines the incoming InHARP_REPLY and 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.

11.2 Registration Phase of Client Y on Broadcast Capable Hardware

If port Y is connected to a broadcast-capable network then the authoritative address is the broadcast address, HWb = SWb, ULAb (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-6400-PH D_ULA                 = ULAb
HIPPI-6400-PH S_ULA                 = ULAy
HARP ar$op                          = 8 (InHARP_REQUEST)
HARP ar$rpa                         = IPy
HARP ar$tpa                         = 0 **
HARP ar$rha                         = ULAy
HARP ar$tha                         = 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.

11.3 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-6400 hardware. The authoritative address in the HRAL for this example will be HWa: <SWa, ULAa> and IPs for simplicity reasons.

11.3.1 Successful HARP_Resolve example

Assume the same process (steps 1-3 of section 11.1) 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-6400-PH D_ULA                = ULAa
   HIPPI-6400-PH S_ULA                = ULAx
   HARP ar$op                         = 1  (HARP_REQUEST)
   HARP ar$rpa                        = IPx
   HARP ar$tpa                        = IPy
   HARP ar$rha                        = 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-6400-PH D_ULA                = ULAx
   HIPPI-6400-PH S_ULA                = ULAa
   HARP ar$op                         = 2  (HARP_Reply)
   HARP ar$rpa                        = IPy
   HARP ar$tpa                        = IPx
   HARP ar$rha                        = ULAy *
   HARP ar$tha                        = 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-6400-PH D_ULA                = ULAy
   HIPPI-6400-PH S_ULA                = ULAx
   <data>
   ```

If the network had been broadcast-capable, 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.

11.3.2 Non-successful HARP_Resolve example

As in 11.3.1, 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-6400-PH D_ULA                = ULAa
HIPPI-6400-PH S_ULA                = ULAx
HARP ar$op                         = 1  (HARP_REQUEST)
HARP ar$rpa                        = IPx
HARP ar$tpa                        = IPq
HARP ar$rha                        = 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-6400-PH D_ULA                = ULAx
HIPPI-6400-PH S_ULA                = ULAa
HARP ar$op                         = 10  (HARP_NAK)
HARP ar$rpa                        = IPx
HARP ar$tpa                        = IPq
HARP ar$rha                        = 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 the network had been broadcast-capable, then there would not have been a reply.

12 References

[1] ANSI NCITS 323-1998, Information Technology - High-Performance Parallel Interface - 6400 Mbit/s Physical Layer (HIPPI-6400-PH).

[2] ANSI NCITS 324-199x, Information Technology - High-Performance Parallel Interface - 6400 Mbit/s Physical Switch Control (HIPPI-6400-SC).

[3] ANSI NCITS Project Number 1249-D, Information Technology - High-Performance Parallel Interface - 6400 Mbit/s Optical Specification (HIPPI-6400-OPT).

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

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

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

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

[8] Chesson, Greg, "HIPPI-6400 Overview", IEEE Hot Interconnects 1996, Stanford University.

[10] ANSI/IEEE Std. 802.2-1989, Information Processing Systems - Local Area Networks - Logical Link Control IEEE, IEEE, New York, New York, 1989.

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

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

[13] Pittet, J.-M., "ARP and IP Broadcast over HIPPI-800", RFC 2834, May 2000.

[14] 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.

[15] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

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

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

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

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

13 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 [17], written by John Renwick. ARP [14] written by Dave Plummer and Inverse ARP [7] written by Terry Bradley and Caralyn Brown provide the fundamental algorithms of HARP as presented in this memo. Further, the HARP server is based on concepts and models presented in [13], written by Mark Laubach who laid the structural groundwork for the HARP server.

14 Author's Address

Jean-Michel Pittet
Silicon Graphics Inc
1600 Amphitheatre Parkway
Mountain View, CA 94040

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

15 Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.