Network Working Group
Request for Comments: 1374

J. Renwick
A. Nicholson
Cray Research, Inc.
October 1992

IP and ARP on HIPPI

Status of this Memo

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

Abstract

The ANSI X3T9.3 committee has drafted a proposal for the encapsulation of IEEE 802.2 LLC PDUs and, by implication, IP on HIPPI. Another X3T9.3 draft describes the operation of HIPPI physical switches. X3T9.3 chose to leave HIPPI networking issues largely outside the scope of their standards; this document discusses methods of using of ANSI standard HIPPI hardware and protocols in the context of the Internet, including the use of HIPPI switches as LANs and interoperation with other networks.

Table of Contents

Introduction

The ANSI High-Performance Parallel Interface (HIPPI) is a simplex data channel. Configured in pairs, HIPPI can send and receive data simultaneously at nearly 800 megabits per second. (HIPPI has an equally applicable 1600 megabit/second option.) Between 1987 and 1991, the ANSI X3T9.3 HIPPI working group drafted four 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 octets (HIPPI-FP [2]), encapsulation of IEEE 802.2 LLC (HIPPI-LE [3]), and the behavior of a standard physical layer switch (HIPPI-SC [4]). HIPPI-LE also implies the encapsulation of Internet Protocol[5]. The reader should be familiar with the ANSI HIPPI documents, copies of which are archived at the site "nsco.network.com" in the directory "hippi," and may be obtained via anonymous FTP until they become published standards.

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. This memo takes up where the working group left off, using the guiding principle that except for length and hardware header, Internet datagrams sent on HIPPI should be identical to the same datagrams sent on a conventional network, and that any datagram sent on a conventional 802 network[6] should be valid on HIPPI.

Scope

This memo describes the HIPPI interface between a host and a crosspoint switch that complies with the HIPPI-SC draft standard. Issues that have no impact on host implementations are outside the scope of this memo. Host implementations that comply with this memo are believed to be interoperable on a network composed of a single HIPPI-SC switch. They are also interoperable on a simple point-to-point, two-way HIPPI connection with no switch between them. They

may as well be interoperable on more complex networks, depending on the internals of the switches and how they are interconnected; however, these details are implementation dependent and outside the scope of this memo. To the extent that a gateway acts as a host on a HIPPI-SC LAN, its behavior is within the scope of this memo.

Within the scope of this memo are:

1. Packet format and header contents, including HIPPI-FP, HIPPI-LE, IEEE 802.2 LLC[7], SNAP and ARP

2. I-Field contents

3. HIPPI switch address resolution, including self discovery

4. Rules for the use of connections.

Outside of the scope are

1. Vendor dependent solutions for multicast or third party ARP

2. Network configuration and management

3. Host internal optimizations

4. The interface between a host and an outboard protocol processor.

Definitions

Conventional
: Used with respect to networks, this refers to Ethernet, FDDI and 802 LAN types, as distinct from HIPPI-SC LANs.

Destination
: The HIPPI implementation that receives data from a HIPPI Source.

Node
: An entity consisting of one HIPPI Source/Destination pair that is connected by parallel or serial HIPPI to a HIPPI-SC switch and that transmits and receives ARP and IP datagrams. A node may be an Internet host, bridge, router or gateway. This memo uses the term node in place of the usual "host" to indicate that a host might be connected to the HIPPI LAN not directly, but through an external adaptor that does some of the protocol processing for the host.

Serial HIPPI
An implementation of HIPPI in serial fashion on coaxial cable or optical fiber, informally standardized by implementor's agreement in the Spring of 1991.

Switch Address
A value used as the address of a node on a HIPPI-SC network. It is transmitted in the I-field. HIPPI-SC switches may map Switch Addresses to physical port numbers.

Source
The HIPPI implementation that generates data to send to a HIPPI Destination.

Universal LAN Address (ULA)
A 48 bit globally unique address, administered by the IEEE, assigned to each node on an Ethernet, FDDI, 802 network or HIPPI-SC LAN.

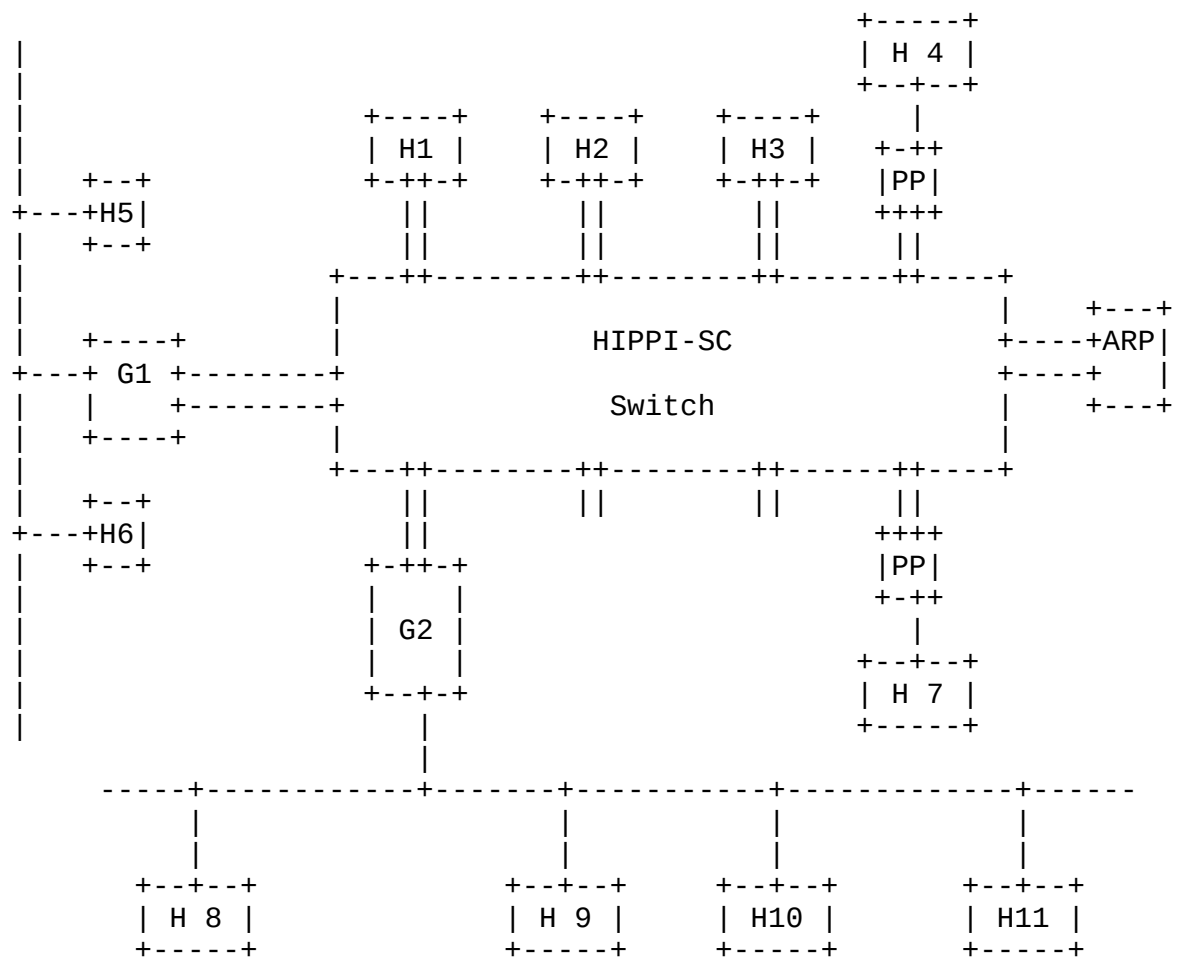
Equipment

A HIPPI network can be composed of nodes with HIPPI interfaces, HIPPI cables or serial links, HIPPI-SC switches, gateways to other networks and, possibly, proprietary equipment that multicasts or responds to ARP requests on behalf of the real nodes.

Each HIPPI interconnection between a node and a switch shall consist of a pair of HIPPI links, one in each direction.

If a link between a node and the switch is capable of the 1600 Megabit/second data rate option (i.e. Cable B installed for 64 bit wide operation) in either direction, the node's HIPPI-PH implementation shall also be capable of 32 bit operation (Cable B data suppressed) and shall be able to select or deselect the 1600Mb/s data rate option at the establishment of each new connection.

The following figure shows a sample HIPPI switch configuration.



Legend: ---+---+---+-- = 802 network, Ethernet or FDDI
|| = Paired HIPPI link
H = Host computer
PP = Outboard Protocol Processor
G = Gateway
ARP = ARP Agent

A possible HIPPI configuration

A single HIPPI-SC switch has a "non-blocking" characteristic, which means there is always a path available from any Source to any Destination. If the network consists of more than one switch, the path from a Source to a Destination may include a HIPPI link between switches. If this link is used by more than one Source/Destination pair, a "blocking" network is created: one Source may be blocked from access to a Destination because another Source is using the link it shares. Strategies for establishing connections may be more

complicated on blocking networks than on non-blocking ones.

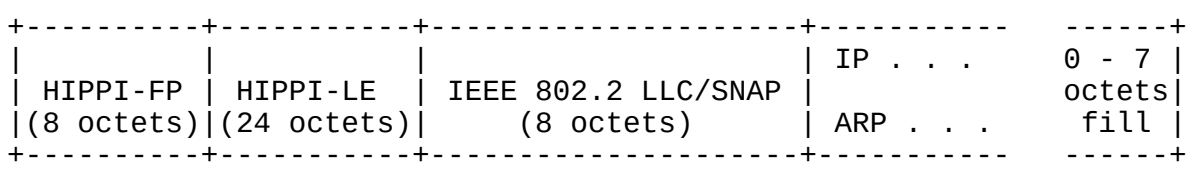
This memo ignores blocking issues, assuming that the HIPPI LAN consists of one HIPPI-SC switch or, if the network is more complex than that, it presents no additional problems that a node must be aware of.

Protocol

Packet Format

The HIPPI packet format for Internet datagrams shall conform to the HIPPI-FP and HIPPI-LE draft standards. The HIPPI-FP D1_Area shall contain the HIPPI-LE header. The HIPPI-FP D2_Area, when present, shall contain one IEEE 802.2 Type 1 LLC Unnumbered Information (UI) PDU. Support of IEEE 802.2 XID, TEST and Type 2 PDUs is not required on HIPPI, and Destinations that receive these PDUs may either ignore them or respond correctly according to IEEE 802.2 requirements.

The length of a HIPPI packet, including trailing fill, shall be a multiple of eight octets as required by HIPPI-LE.



HIPPI Packet Structure

HIPPI-FP Header

ULP-id (8 bits) shall contain 4.

D1_Data_Set_Present (1 bit) shall be set.

Start_D2_on_Burst_Boundary (1 bit) shall be zero.

Reserved (11 bits) shall contain zero.

D1_Area_Size (8 bits) should be sent as 3. Destinations shall accept any value that HIPPI-FP defines as legal: from 3 to 127 (32 bit HIPPI) or 3 to 255 (64 bit HIPPI).

D2_Offset (3 bits) may be any value from 0 to 7.

D2_Size (32 bits) Shall contain the number of octets in the

IEEE 802.2 LLC Type 1 PDU, or zero if no PDU is present. It shall not exceed 65,288 (decimal). This value includes the IEEE 802.2 LLC/SNAP header and the IP datagram. It does not include trailing fill octets. (See "MTU," below.)

The first octet of the IEEE 802.2 LLC PDU (SSAP) shall be located at offset "n" of the packet, where

n = 8 + (D1_Area_Size*8) + D2_Offset

as specified in HIPPI-FP.

HIPPI-LE Header

FC (3 bits) shall contain zero unless otherwise defined by local administration.

Double_Wide (1 bit) shall contain one if the Destination associated with the sending Source supports 64 bit HIPPI operation. Otherwise it shall contain zero.

Message_Type (4 bits) contains a code identifying the type of HIPPI-LE PDU. Defined values (binary) are:

0 Data PDU
1 Address Resolution Request PDU (AR_Request)
2 Address Resolution Response PDU (AR_Response)
3 Self Address Resolution Request PDU (AR_S_Request)
4 Self Address Resolution Response PDU (AR_S_Response)
5-11 Reserved by the ANSI X3T9.3 committee
12-15 Locally Assigned

Destination_Switch_Address is a 24-bit field containing the Switch Address of the Destination if known, otherwise zero. If the address comprises less than 24 bits, it shall be right justified (occupying the least significant bits) in the field.

Destination_Address_Type (4 bits) and Source_Address_Type (4 bits) contain codes identifying the type of addresses in the Destination_Switch_Address and Source_Switch_Address fields respectively. Defined values (binary) are:

0 Unspecified
1 HIPPI-SC Source Route (24 bits)
2 HIPPI-SC Address (12 bits)
3-11 Reserved by the ANSI X3T9.3 committee
12-15 Locally Assigned

Source_Switch_Address is a 24-bit field containing the Switch Address of the Source. If the address comprises less than 24 bits, it shall be right justified (occupying the least significant bits) in the field.

Reserved (16 bits) shall contain zero.

Destination_IEEE_Address (48 bits) shall contain the 48 bit Universal LAN MAC Address of the Destination if known, otherwise zero.

LE_Locally_Administered (16 bits) shall contain zero unless otherwise defined by local administration.

Source_IEEE_Address (48 bits) shall contain the 48 bit Universal LAN MAC Address of the Source if known, otherwise zero.

IEEE 802.2 LLC

The IEEE 802.2 LLC Header shall begin in the first octet of the HIPPI-FP D2_Area.

SSAP (8 bits) shall contain 170 (decimal).

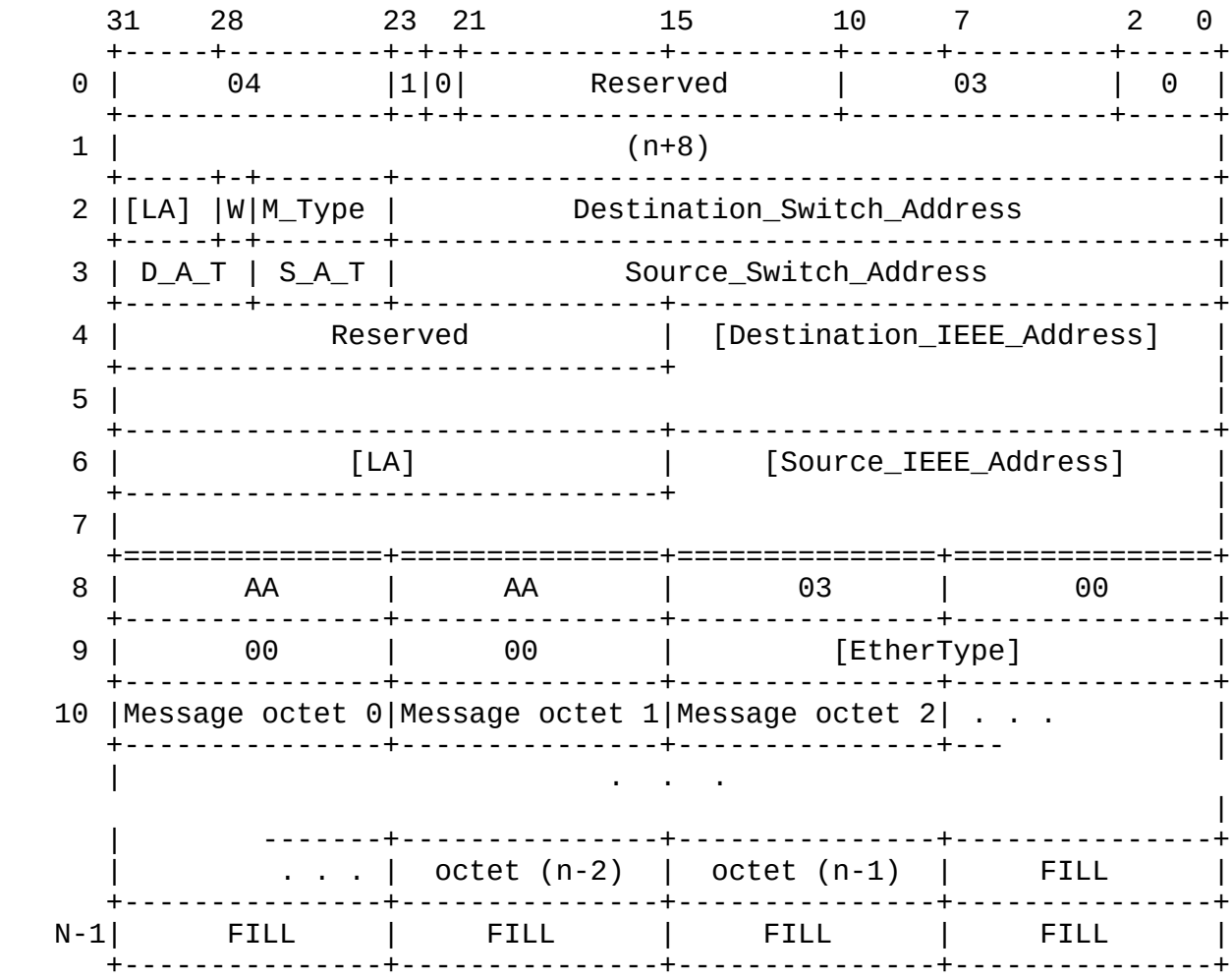
DSAP (8 bits) shall contain 170 (decimal).

CTL (8 bits) shall contain 3 (Unnumbered Information).

SNAP

Organization Code (24 bits) shall be zero.

EtherType (16 bits) shall be set as defined in Assigned Numbers [8] (IP = 2048, ARP = 2054, RARP = 32,821).



HIPPI Packet 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 (IP or ARP message)
(n) is the number of octets in the IP or ARP message.
+====+ denotes the boundary between D1 and D2 areas.
[LA] fields are zero unless used otherwise locally.
Abbreviations: "W" = Double_Wide field;
"M_Type" = Message_Type field;
"D_A_T" = Destination_Address_Type;
"S_A_T" = Source_Address_Type;
[FILL] octets complete the HIPPI packet to an even number of 32 bit words. The number of fill octets is not counted in the data length.

IEEE 802.2 Data

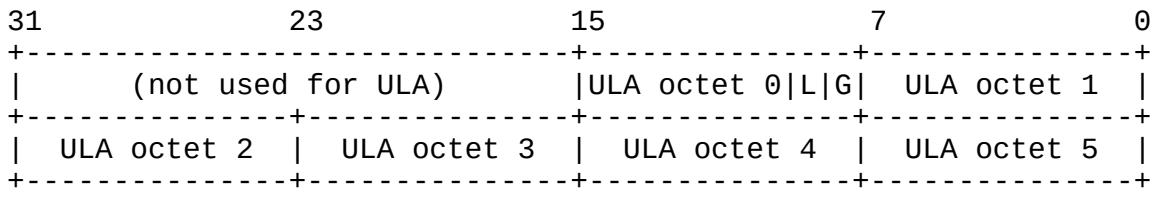
The IEEE 802.2 Data shall follow the EtherType field immediately. Fill octets shall be used following the Data as necessary to make the number of octets in the packet a multiple of 8. In accordance with HIPPI-FP, the amount of this fill is not included in the D2_Size value in the HIPPI-FP Header.

The order of the octets in the data stream is from higher numbered to lower numbered data signal (left to right) within the HIPPI word, as specified in HIPPI-FP Clause 7, "Word and byte formats." With the 1600 megabit/second data rate option (64 bit) bits 32 through 63 are on Cable B, so that the four octets on Cable B come logically before those on Cable A. Within each octet, the most significant bit is the highest numbered signal.

48 bit Universal LAN MAC Addresses

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

The format of the address within its 48 bit HIPPI-LE fields follows IEEE 802.1A canonical bit order and HIPPI-FP 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.

The use of ULAs is optional, but encouraged. Although ULAs are not used by HIPPI-SC switches, they are helpful for HIPPI Switch Address resolution, and for distinguishing between multiple logical entities that may exist within one node. They may also be used by gateway devices that replace HIPPI hardware headers with the MAC headers of other LANs. Carrying the ULAs in the HIPPI header may simplify these devices, and it may also help if HIPPI is used as an interface to some future HIPPI based LAN that uses ULAs for addressing.

Recommended HIPPI-FP Options

HIPPI-FP allows some flexibility in the construction of a HIPPI packet, including placement of short bursts, optional fill and offset octets between the D1 and D2 areas and fill following the D2 data. For efficiency, Sources should limit the use of these options:

1. Send the short burst as the last burst of the packet rather than the first.

2. Do not place fill octets between the HIPPI-LE header and the start of the D2_Area.

3. Use no more than seven octets after the D2 Data, as needed to make the total packet length a multiple of 8 octets.

One HIPPI-FP option is forbidden: setting the Start_D2_on_Burst_Boundary flag to one. This places no limitation on the formation of packets into a series of bursts; a Source may segment the packet in any legal manner according to HIPPI-FP, including forcing the D2_Area to start on a burst boundary. The purpose of the Start_D2_on_Burst_Boundary flag is to help preserve the segmentation of the packet for some device-control protocols that use the first burst boundary to separate command and data areas within the packet. Requiring this flag to be clear means that when a packet arrives at the Destination its burst boundaries might not be exactly as the Source sent them. This may occur if a HIPPI packet passes over some other medium in the route between HIPPI LANs.

Notwithstanding these recommendations, each Destination shall accept any well-formed HIPPI packet within the definitions in HIPPI-FP.

Note that neither HIPPI-FP nor HIPPI-LE limits the number of fill bytes placed between the end of the IP packet and the end of the HIPPI-PH packet. Some source implementations may add fill sufficient to overflow a destination input buffer. To avoid interpreting valid packets as errors, destinations should ignore overflow conditions and verify that at least the number of bytes indicated by the IP header actually arrived.

I-Field format

The I-field bits, as defined in HIPPI-SC, shall be set as follows:

Locally Administered (bit 31) shall be zero.

Reserved (bits 30, 29) should be zero. Destinations shall accept any value for these bits.

Double wide (bit 28) shall be set when Source Cable B is connected and the Source wants a 64 bit connection. It shall be zero otherwise.

Direction (bit 27) should be sent as zero, however Destinations shall accept either zero or one and interpret the Routing Control field accordingly, per HIPPI-SC.

Path Selection (bits 26, 25) shall be 00, 01, or 11 (binary) at the Source's option. 00 (source route mode) indicates that the I-field bits 23-00 contain a 24 bit source route; 01 or 11 (logical address mode) indicate that bits 23-00 contain 12 bit Source and Destination Addresses. The value 11 is meaningful when more than one route exists from a Source to a Destination; it allows the switch to choose the route. Use of 01 forces the switch always to use the same route for the same Source/Destination pair.

Camp-on (bit 24) may be 1 or 0; however, a Source shall not make consecutive requests without Camp-on to the same Destination while the requests are being rejected. The purpose of this restriction is to prevent a node from circumventing the fair share arbitration mechanism of the switch by repeating requests at a very high rate.

If logical address mode is used:

   Source Address (bits 23-12) is not used.

   Destination Address (bits 11-0) shall contain the Switch Address of the Destination.

If source route mode is used:

   Routing control (bits 23-00) shall contain the route to the Destination.

Note: the outcome of Switch Address Resolution (see "Address Resolution" below) determines whether to use logical address mode or source route mode. If source route mode is used with multiple interconnected switches, different sources may use different addresses to reach the same destination, and multicast-based address resolution may not be possible because a target node may not know the route to itself from a given remote source. Regardless of this difficulty, it may be possible to use source route mode if the network consists of a single switch, or if address resolution is supported by an ARP agent that is able to deliver correct routes to each node. The nodes themselves need not be concerned with these problems if they use the addressing

mode suggested by the value of the Source_Address_Type field in a HIPPI-LE Address Resolution Response packet.

Rules For Connections

The following rules for connection management by Source and Destination are intended to insure frequent, fair share access to Destinations for which multiple Sources are contending. If possible, nodes should transfer data at full HIPPI speeds and hold connections no longer than necessary.

A source may hold a connection for as long as it takes to send 68 HIPPI bursts at what ever speed the two connected nodes can achieve together. The number of packets sent in one connection is not limited, except that the number of bursts over all the packets should not exceed 68. This is not a recommendation to send as many packets as possible per connection; one packet per connection is acceptable. The purpose of this limit is to give each Source an fair share of a common Destination's bandwidth. Without a limit, if there is a Destination that is constantly in demand by multiple Sources, the Source that sends the most data per connection wins the greatest share of bandwidth.

The limit of 68 bursts is not absolute. An implementation may check the burst count after transmission of a packet and end the connection if it is greater than or equal to some threshold. If this is done, the threshold should be less than 68 depending on the typical packet size, to ensure that the 68 burst limit is not normally exceeded. For instance, a Source sending 64K packets would send two per connection (130 bursts) if it checked for 68 at the end of each packet. In this situation the Source is required to check for a value small enough that it will not send a second packet in the same connection.

Destinations shall accept all packets that arrive during a connection, and may discard those that exceed its buffering capacity. A Destination shall not abort a connection (deassert CONNECT) simply because too many bursts were received; however a Destination may abort a connection whose duration has exceeded a time period of the Destination's choosing, as long as the Source is allowed ample time to transmit its quota of bursts.

The rules admonish the node to do certain things as fast as it can, however there is no absolute measure of compliance. Nodes that cannot transfer data at full HIPPI speeds can still interoperate but the faster the implementation, the better the performance of the network will be.

Assuming that bursts flow at the maximum rate, the most important factor in network throughput is the connection switching time, measured from the deassertion of REQUEST by the Source at the end of one connection to its first assertion of BURST after the establishment of the new connection. Implementations should keep this time as short as possible. For a guideline, assuming parallel HIPPI and a single HIPPI-SC switch, ten microseconds permits nearly full HIPPI throughput with full-sized packets, and at 60 microseconds the available throughput is reduced by about 10%. (See "Performance," below.)

All HIPPI electrical signaling shall comply with HIPPI-PH. In every case, the following rules go beyond what HIPPI-PH requires.

Rules for the Source

1. Do not assert REQUEST until a packet is ready to send.

2. Transmit bursts as quickly as READYs permit. Except for the required HIPPI Source Wait states, there should be no delay in the assertion of BURST whenever the Source's READY counter is nonzero.

3. Make a best effort to ensure that connection durations do not exceed 68 bursts.

4. Deassert REQUEST immediately when no packet is available for immediate transmission or the last packet of the connection has been sent.

Rules for the Destination

1. Reject all connections if unable to receive packets. This frees the requesting Source to connect to other Destinations with a minimum of delay. Inability to receive packets is not a transient condition, but is the state of the Destination when its network interface is not initialized.

2. A HIPPI node should be prepared to efficiently accept connections and process incoming data packets. While this may be best achieved by not asserting connect unless 68 bursts worth of buffers is available, it may be possible to meet this requirement with fewer buffers. This may be due to a priori agreement between nodes on packet sizes, the speed of the interface to move buffers, or other implementation dependent considerations.

3. Accept a connection immediately when buffers are available. The Destination should never delay the acceptance of a connection unnecessarily.

4. Once initialized, a Destination may reject connection requests only for one of the following reasons:

   1. The I-field was received with incorrect parity.

   2. The I-field contents are invalid, e.g. the "W" bit set when the Destination does not support the 1600 megabit data rate option, the "Locally Administered" bit is set, the Source is not permitted to send to this Destination, etc.

   Transient conditions within the Destination, such as temporary buffer shortages, must never cause rejected connections.

5. Ignore aborted connection sequences. Sources may time out and abandon attempts to connect; therefore aborted connection sequences are normal events.

MTU

Maximum Transmission Unit (MTU) is defined as the length of the IP packet, including IP header, but not including any overhead below IP. 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 has no inherent limit on packet size. The HIPPI-FP header contains a 32 bit D2_Size field that, while it may limit packets to about 4 gigabytes, imposes no practical limit for networking purposes. Even so, a HIPPI-SC switch used as a LAN needs an MTU so that Destination buffer sizes can be determined.

The MTU for HIPPI-SC LANs is 65280 (decimal) octets.

This value was selected because it allows the IP packet to fit in one 64K octet buffer with up to 256 octets of overhead. The overhead is 40 octets at the present time; there are 216 octets of room for expansion.

HIPPI-FP Header	8 octets
HIPPI-LE Header	24 octets
IEEE 802.2 LLC/SNAP Headers	8 octets
Maximum IP packet size (MTU)	65280 octets
Total	65320 octets (64K - 216)

Camp-on

When several Sources contend for a single Destination, the Camp-on feature allows the HIPPI-SC switch to arbitrate and ensure that all Sources have fair access. (HIPPI-SC does not specify the method of arbitration.) Without Camp-on, the contending Sources would simply have to retry the connection repeatedly until it was accepted, and the fastest Source would usually win. To guarantee fair share arbitration, Sources are prohibited from making repeated requests to the same Destination without Camp-on in such a way as to defeat the arbitration.

There is another important reason to use Camp-on: when a connection without Camp-on is rejected, the Source cannot determine whether the rejection came from the requested Destination or from the switch. The Source also cannot tell the reason for the rejection, which could be either that the Destination was off line or not cabled, or the I-field was erroneous or had incorrect parity. Sources should not treat a rejection of a request without Camp-on as an error. Camp-on prevents rejection due to the temporary busy case; with one exception, rejection of a Camp-on request indicates an error condition, and an error event can be recorded. The exception occurs when a 64 bit connection is attempted to a Destination that does not have Cable B connected, resulting in a reject. This case is covered in "Channel Data Rate Discovery," below.

Address Resolution

The Internet Address Resolution Protocol (ARP) is defined in RFC 826 [9]. Ethernet, FDDI and 802 networks use ARP to discover another host's ULA knowing the Internet address. Reverse ARP [10] is used to discover the Internet address, knowing the ULA. ARP can be used in the conventional way on HIPPI-SC LANs equipped with a multicast capability or third party ARP agent.

HIPPI-LE defines similar lower-level address resolution between ULAs and switches. HIPPI-LE adds a self-address resolution mechanism not defined for Internet ARP, which allows a node to discover its own switch address dynamically.

ARP for the purpose of discovering ULAs is not necessary for the operation of a HIPPI-SC LAN, but it serves as the vehicle for discovery of HIPPI-SC Switch Addresses, without which the HIPPI-SC LAN cannot function. In other words, at the same time a node is using ARP to map another node's IP address to its ULA, it is also mapping the ULA to the 12 bit HIPPI Switch Address, from which it will construct the I-field value for sending messages to that node. This additional level of hardware addressing uses the address fields in the HIPPI-LE header.

In the following discussion, the terms "requester" and "target" are used to identify the node requesting address resolution and the node whose address it wishes to discover, respectively. In third party ARP (see "ARP Implementation Methods," below) the source of a reply is an ARP agent node, not the target node.

ARP and RARP Message Format

The HIPPI ARP/RARP protocol uses the same packet format as ARP for Ethernet. ARP packets shall be transmitted with a hardware type code of 1 (as for Ethernet). Furthermore, ARP packets shall be accepted if received with hardware type codes of either 1 or 6 (IEEE 802 networks).

ar$hrd (16 bits) shall contain 1.

ar$pro (16 bits) shall contain the IP protocol code 2048 (decimal).

ar$hln (8 bits) shall contain 6.

ar$pln (8 bits) shall contain 4.

ar$op (16 bits) shall contain 1 for requests, 2 for responses.

ar$sha (48 bits) in requests shall contain the requester's ULA. In replies it shall contain the target node's ULA.

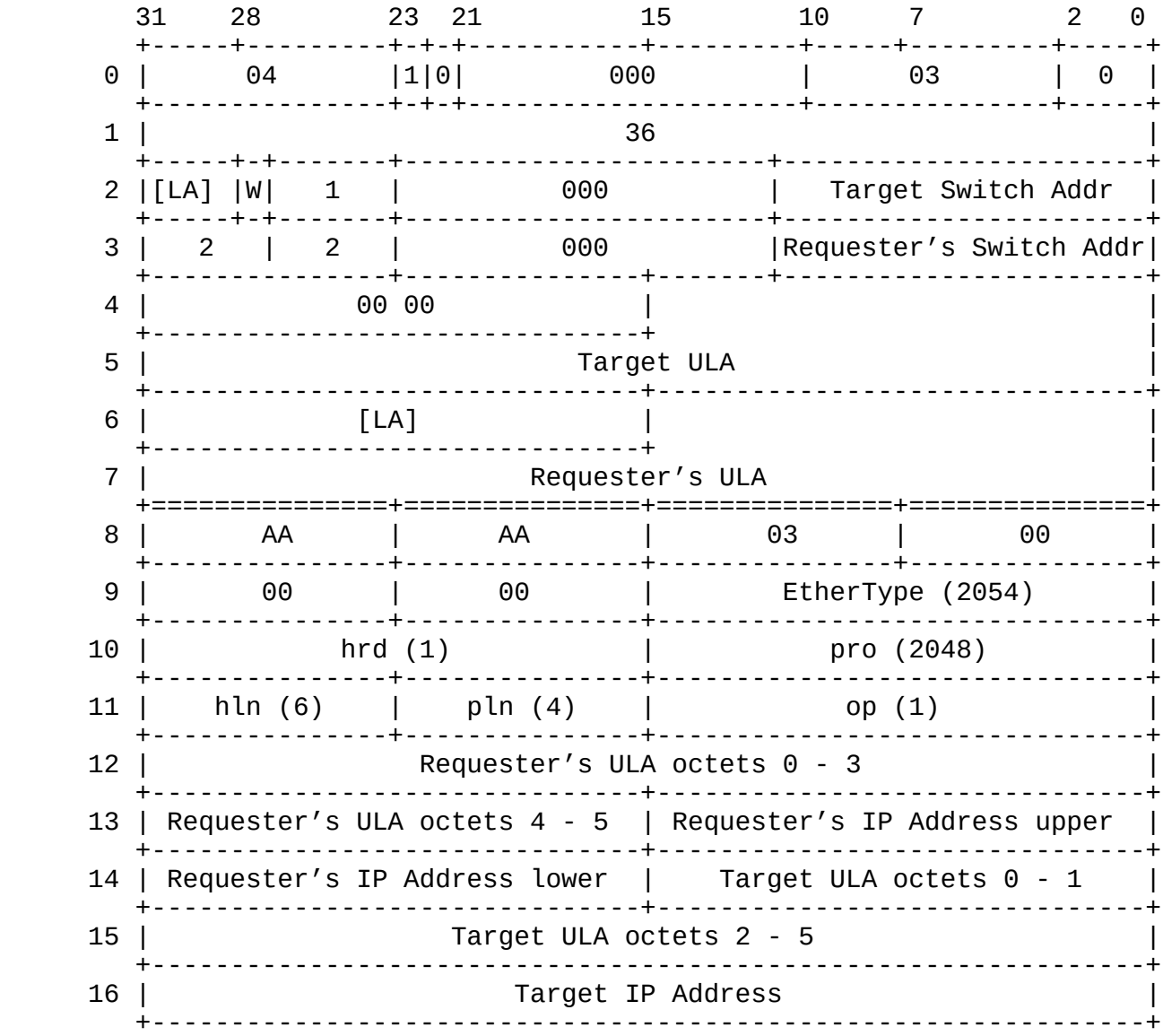
ar$spa (32 bits) in requests shall contain the requester's IP address if known, otherwise zero. In replies it shall contain the target node's IP address.

ar$tha (48 bits) in requests shall contain the target's ULA if known, otherwise zero. In replies it shall contain the requester's ULA.

ar$tpa (32 bits) in requests shall contain the target's IP address if known, otherwise zero. In replies it shall contain the

requester's IP address.

The format of the six octets of the ULA shall be the same as required in the HIPPI-LE header (see "48 bit Universal LAN MAC Addresses" above), except for the alignment of the Source ULA 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 [11].



HIPPI ARP/RARP Request (logical address mode)

All ARP requests shall be sent with the I-field bit 28 set to zero, i.e. requesting a 32 bit connection.

Unless another convention is locally defined for ARP requests, the I-field Path Selection bits may be set to binary 01 or 11 (logical address mode), and Destination Address field set to the HIPPI-SC address reserved for traffic conventionally directed to the IEEE 802.1[12] broadcast address (which HIPPI-SC defines as FE0, hex).

Reply packets shall be sent with I-field Path Selection and Routing Control fields set according to the Source_Address_Type and Source_Switch_Address fields in the request.

In the HIPPI-LE header of ARP/RARP requests and replies the following fields shall be set:

Double-Wide should be 1 if the HIPPI Destination at the sending node can accept 64 bit HIPPI connections.

Message_Type shall contain an address resolution type code as defined in HIPPI-LE. It shall be set appropriately to the value of the ARP operation code (ar$op) in piggybacked ARP messages:

ARP ar$op	HIPPI-LE Message_Type
ARP Request (1)	AR_Request (1)
ARP Reply (2)	AR_Response (2)
Reverse ARP Request (3)	AR_Request (1)
Reverse ARP Reply (4)	AR_Response (2)

There is no ARP message corresponding to HIPPI-LE self address discovery; these packets are sent without ULP data.

Destination_Switch_Address in requests shall be the Switch Address of the target node if known, otherwise zero. In replies it shall be the requesting node’s Switch Address

Destination_Address_Type shall be 1 if the Destination_Switch_Address is a source route, 2 if it is a 12 bit address.

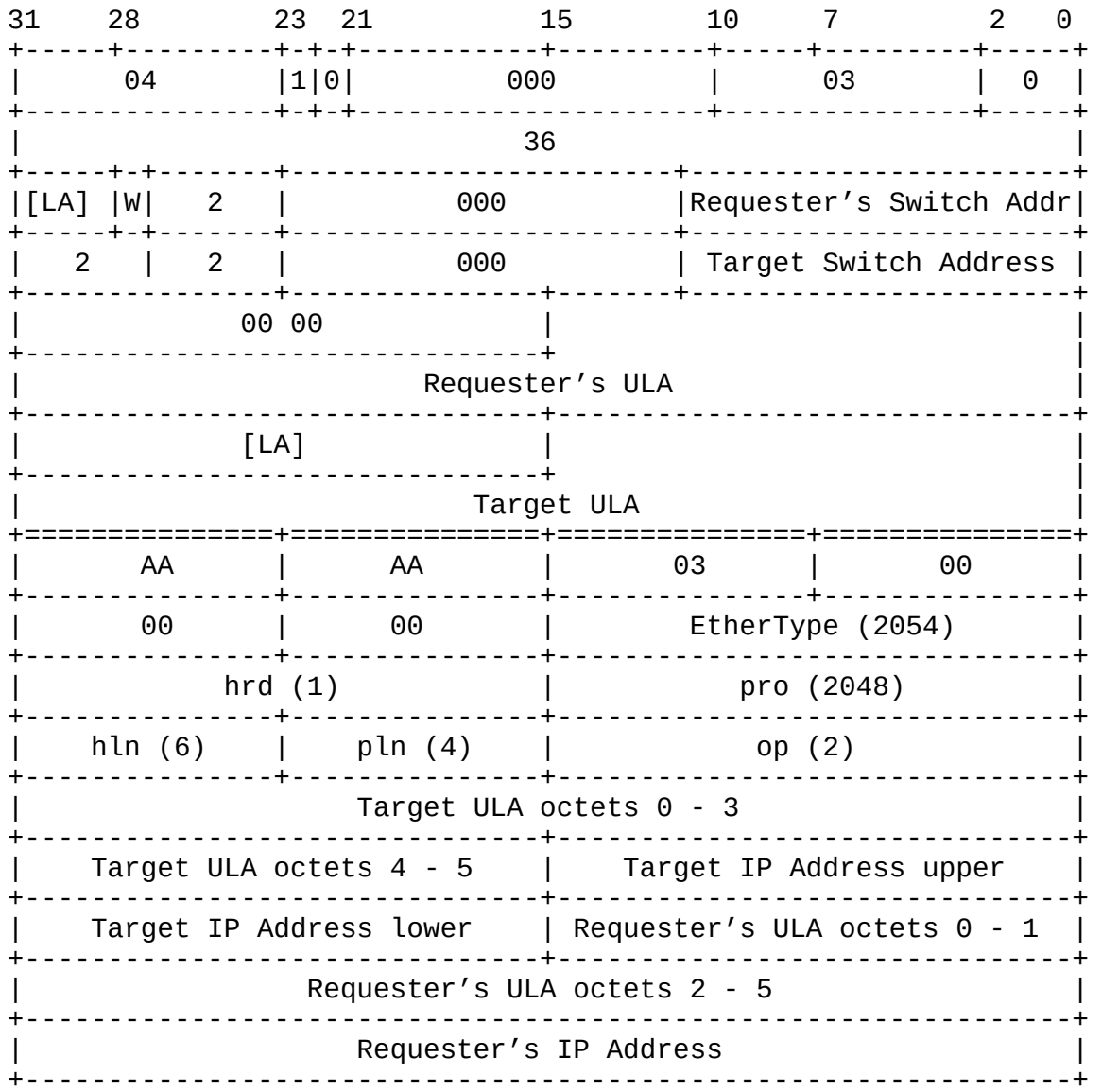
Source_Address_Type shall be 1 if the Source_Switch_Address is a source route, 2 if it is a 12 bit address.

Source_Switch_Address in requests shall be the Switch Address of the requesting node if known, otherwise zero. In replies it shall be the

target node’s Switch Address.

Destination_IEEE_Address shall be the same as the ar$tha field in the ARP message.

Source_IEEE_Address shall be the same as the ar$sha field in the ARP message.



HIPPI ARP/RARP Reply (logical address mode)

ARP procedure

The combined HIPPI-LE/ARP packet contains six addresses, three each for the requester and the target:

Requester's IP Address	(ARP)
Requester's ULA	(ARP and HIPPI-LE)
Requester's Switch Address	(HIPPI-LE)
Target's IP Address	(ARP)
Target's ULA	(ARP and HIPPI-LE)
Target's Switch Address	(HIPPI-LE)

Internet ARP concerns the IP Address and ULA; HIPPI-LE address resolution concerns the ULA and Switch Address. Thus the ULA appears in both parts of the packet.

Successful ARP results in tables in each node that map remote nodes' IP addresses to ULAs and ULAs to Switch Addresses, so that when an application requests a connection to a remote node by its IP address, both the remote ULA and Switch Address can be determined, a correct HIPPI-LE header can be built, and a connection to the node can be established using the correct Switch Address in the I-field. Any recipient of an ARP request or reply may use information in the packet to augment its tables, even if it is neither the target node nor the requester.

Note that the use of ULAs with HIPPI is not required. In both the HIPPI-LE header and the Internet ARP message, the fields that contain ULAs should be set to zero when the ULA is not known. Address resolution consists of two separate protocols, HIPPI-LE address resolution and Internet ARP, neither of which can function independently without ULAs. However HIPPI Switch Address resolution can work without ULAs if the two protocols are piggybacked and treated as one operation in which Internet addresses are mapped directly to switch addresses. With the exception of the optional self-address resolution request, which has no analogous Internet protocol, HIPPI-LE address resolution and Internet ARP messages should be sent together as a single HIPPI packet.

If ULAs are used, the HIPPI-LE address resolution request can be sent without a piggybacked 802.2 LLC PDU, so it is possible to map ULAs to HIPPI Switch Addresses without using ARP. Nodes shall accept both piggybacked and non-piggybacked forms of HIPPI-LE address resolution messages.

The recipient of an address resolution request, having first updated its address mapping tables with any new information it can find in

the request, checks to see if it is the target node. If it is, it generates a reply by filling in the unknown target address fields according to the HIPPI-LE message type and the ARP operation code, and swapping the four pairs of source/target address fields. Then it connects to the requesting node with the Source Switch Address from the request, and sends the reply packet.

A node is the target of an address resolution request if the request contains one of the following:

1. the node's ULA in the Destination_IEEE_Address field of a HIPPI-LE AR_Request message

2. the node's IP address in the target protocol address field (ar$tpa) of a piggybacked Internet ARP message

If two target fields are known but are not mapped together in the recipient's address mapping tables, it may do one of three things:

1. treat the request as having two targets, and send correct replies for both to the requester.

2. assume its own tables are invalid and ignore the request.

3. assume one of the "known" target fields is correct and respond as if the other had been unknown.

The best choice depends on which fields conflict and the nature of the implementation. Choice 3 is probably best for ordinary nodes, but third party ARP agents may have reason to use one of the other two. Future experience may shed light on this.

ARP Implementation Methods

The requirements for nodes to handle address resolution messages depend on the means by which address resolution is implemented on the LAN.

In conventional networks ARP is a distributed function. ARP requests are broadcast; each host may update its address mappings with any ARP request or reply it sees, and responds to ARP requests that contain its own address in the target protocol address field. HIPPI-SC switches are not required to provide multicast service, although some do. Even if the switches do not multicast, one or more nodes can act as multicast servers, receiving packets sent to the HIPPI-SC broadcast address and repeating them to each other node in turn. Either way, if multicast exists on a HIPPI-SC LAN, ARP can be a distributed function. In this situation each node is required to

respond correctly to address resolution requests for which it is the target.

Third party ARP is a second method that does not depend on multicast. The switches can map the HIPPI ARP multicast address to a node that acts as an ARP agent, replying to ARP requests on behalf of the real target nodes. Ordinary nodes never receive ARP requests or generate replies and never have the opportunity to update mapping tables based on ARP requests from other nodes, as usually happens on conventional networks. Each node must request any address information it needs, but never has to process ARP information it doesn't need. Under third party ARP a node should not receive address resolution requests, and each node that is not an ARP agent should ignore those that it does receive.

As a third possibility, one can omit the implementation of ARP entirely, choosing instead to build address mapping tables in each node from information available to a network administrator. Such a technique is implementation dependent and outside the scope of this memo. It may be helpful in prototype networks without multicast where no ARP agent is yet implemented. In this case, nodes are not required to respond to address resolution requests, but must accept any they may receive.

ARP Example

Assume a HIPPI-SC switch is installed with two nodes, X and Y, connected. Each node has a unique Switch Address. Both nodes have access to the host data base (e.g. /etc/hosts) in which the network administrator has configured the network and given the two nodes IP addresses. There is an ARP agent connected to a switch port that is mapped to the address FE0 (hex). The ARP agent contains no mappings of any IP, IEEE or Switch addresses. Both nodes know their own ULAs and Switch Addresses. They want to talk to each other; each knows the other's IP address (from the host data base) but neither knows the other's ULA or Switch Address. Node X starts:

1. Node X connects to FE0 and sends a piggyback ARP Request requesting addresses for Y:

```
HIPPI-LE Message_Type is 1, AR_Request
HIPPI-LE Destination_Switch_Address = 0
HIPPI-LE Source_Switch_Address = X's Switch Address
HIPPI-LE Destination_IEEE_Address = 0
HIPPI-LE Source_IEEE_Address = X's ULA
ARP ar$op = 1 (request)
ARP ar$sha = X's ULA
ARP ar$spa = X's IP Address
```

```
ARP ar$tha = 0
ARP ar$tpa = Y's IP Address
```

2. The ARP agent receives the ARP request and adds an entry for X to its address mapping table. It does not know about Y, so it does not generate a reply.

3. Node X waits for a reply. It may set a timer to retransmit the request periodically, but its requests will be ignored until node Y sends an ARP request.

4. Node Y connects to FE0 and sends an ARP request requesting addresses for X:

```
HIPPI-LE Message_Type is 1, AR_Request
HIPPI-LE Destination_Switch_Address = 0
HIPPI-LE Source_Switch_Address = Y's Switch Address
HIPPI-LE Destination_IEEE_Address = 0
HIPPI-LE Source_IEEE_Address = Y's ULA
ARP ar$op = 1 (request)
ARP ar$sha = Y's ULA
ARP ar$spa = Y's IP Address
ARP ar$tha = 0
ARP ar$tpa = X's IP Address
```

5. The ARP agent receives Y's request and adds an entry for Y to its address mapping table. It knows about the target node, X, so it connects to Y (using the Source_Switch_Address given in the request) and sends an ARP Reply:

```
HIPPI-LE Message_Type is 2, AR_Reply
HIPPI-LE Destination_Switch_Address = Y's Switch Address
HIPPI-LE Source_Switch_Address = X's Switch Address
HIPPI-LE Destination_IEEE_Address = Y's ULA
HIPPI-LE Source_IEEE_Address = X's ULA
ARP ar$op = 2 (reply)
ARP ar$sha = X's ULA
ARP ar$spa = X's IP Address
ARP ar$tha = Y's ULA
ARP ar$tpa = Y's IP Address
```

6. Node Y receives the ARP reply and builds its address mapping table entry for Node X.

7. Node Y connects to node X and transmits an IP packet with the following information in the HIPPI-LE header:

```
HIPPI-LE Message_Type is 0, AR_Data
```

HIPPI-LE Destination_Switch_Address = X's Switch Address
HIPPI-LE Source_Switch_Address = Y's Switch Address
HIPPI-LE Destination_IEEE_Address = X's ULA
HIPPI-LE Source_IEEE_Address = Y's ULA

8. Node X receives the IP packet. Since the ARP agent now knows about node Y, node X can retransmit its ARP request (repeating step 1) and receive an ARP reply:

HIPPI-LE Message_Type is 2, AR_Reply
HIPPI-LE Destination_Switch_Address = X's Switch Address
HIPPI-LE Source_Switch_Address = Y's Switch Address
HIPPI-LE Destination_IEEE_Address = X's ULA
HIPPI-LE Source_IEEE_Address = Y's ULA
ARP ar$op = 2 (reply)
ARP ar$sha = Y's ULA
ARP ar$spa = Y's IP Address
ARP ar$tha = X's ULA
ARP ar$tpa = X's IP Address

Address resolution is now complete for both nodes.

If there had been a multicast facility instead of the ARP agent in the configuration, the target nodes themselves would have received the requests and responded to them in the same way the ARP agent did.

Discovery of One's Own Switch Address

The ARP example above assumed that each node had prior knowledge of its own switch address. This may be manually configured, by means that are outside the scope of this memo, when each node is connected to the switch. If a multicast capability exists, the node may discover its own address automatically when it starts up, using a protocol defined in HIPPI-LE.

In the self-address discovery protocol, a node connects to a multicast address and sends a HIPPI-LE message containing its own ULA. It receives a multicast copy of its own message, and learns its own switch address from the destination address field of the received I-field.

HIPPI-LE self address resolution uses the same HIPPI-LE message format described in "ARP and RARP Message Format," above, with the AR_S_Request and AR_S_Response message type codes and no piggybacked ULP data. The HIPPI-LE header contents for the request are:

Message_Type is 3, AR_S_Request
Destination_Address_Type = 0 (undefined)

```
Destination_Switch_Address = 0 (unknown)
Source_Address_Type = 0 (undefined)
Source_Switch_Address = 0 (unknown)
Destination_IEEE_Address = my ULA
Source_IEEE_Address = my ULA
```

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

The node that wants to discover its address connects to the multicast address for this purpose (hex FE0 in HIPPI-SC) and transmits the request packet. What happens next depends on the particular network:

With multicast:

> The node receives its own request and can learn its own switch address from the I-field it receives. This is the only time a node should use an address from a received I-field.

With an ARP agent:

> The node may receive an AR_S_Response message with its own ULA in the Destination_IEEE_Address field and its own switch address in the Destination_Switch_Address field. This address may be different from the address contained in the I-field, and should be used instead.

The ARP agent response alternative requires that the agent have prior knowledge of the node's location and ULA through some process not specified by this memo. The node may receive both its request and the agent's response if both an ARP agent and multicast are active. In this case the address it learns from the I-field is later replaced by the address given by the ARP agent in the response. Agents may assign new addresses to nodes and inform them by sending unsolicited AR_S_Response messages. Any node whose switch address is updated in this way should invalidate the switch addresses it has saved for other nodes, and use ARP to rediscover them.

If the node reacts correctly to either the multicast request or agent-generated response, it can discover its address without having to know whether or not an ARP agent is active. The full procedure is:

1. Transmit the AR_S_Request

2. When a connection arrives, accept it and save the I-field for later analysis.

3. Receive the message and look at the HIPPI-LE header. If the message is Message Type AR_S_Request, analyze the I-field to discover the node's own switch address. HIPPI-SC I-field formats suggest the following:

```
if bit 25 == 1
        Address type is HIPPI-SC logical address.
        if bit 27 == 1
                take address from bits 23-12
        else
                take address from bits 11-00
        endif
else
        Address is unusable (source route)
endif
```

   This is a one-time operation. Once the node knows an address for itself, it should not take any new address from a received I-field.

4. If a message of type AR_S_Response arrives and the Destination_IEEE_Address field contains the node's own ULA, take the new switch address from the Destination_Switch_Address field and its type from the Destination_Address_Type field.

5. The node should invalidate its ARP tables when an AR_S_Response changes its own switch address, to force retransmission of ARP requests containing its new address to all the remote nodes with which it communicates.

Path MTU Discovery

RFC 1191 [13] describes the method of determining MTU restrictions on an arbitrary network path between two hosts. HIPPI nodes may use this method without modification to discover restrictions on paths between HIPPI-SC LANs and other networks. Gateways between HIPPI-SC LANs and other types of networks should implement RFC 1191.

Channel Data Rate Discovery

HIPPI exists in two data rate options (800 megabit/second and 1600 megabit/second). The higher data rate is achieved by making the HIPPI 64 bits parallel instead of 32, using an extra cable containing 32 additional data bits and four parity bits. HIPPI-SC switches can

be designed to attach to both. Source and Destination HIPPI implementations can be designed to operate at either rate, selectable at the time a connection is established. The "W" bit (bit 28) of the I-field controls the width of the connection through the switch. Sources with both cables A and B attached to the switch may set the "W" bit to request a 1600 megabit/second connection. If the requested destination also has both cables attached, the switch can connect Source to Destination on both cables. If the requested Destination has only Cable A, the switch rejects the request. Sixty-four bit Sources can connect to 32 bit Destinations by requesting with the "W" bit clear and not using Cable B. Sixty-four bit Destinations must examine the "W" bit in the received I-field and use or ignore Cable B accordingly. Note that both INTERCONNECT signals stay active while a 64 bit HIPPI is used in 32 bit mode.

The following table summarizes the possible combinations, the switch's action for each, and the width of the resulting connection.

			Destination: 32	Destination: 64
Source: 32	W=0		Accept 32	Accept 32
Source: 32	W=1		N/A	N/A
Source: 64	W=0		Accept 32	Accept 32
Source: 64	W=1		Reject	Accept 64

HIPPI Connection Combinations

If the path between a 64 bit Source and a 64 bit Destination includes more than one switch, and the route between switches uses a link that is only 32 bits wide, the switch rejects 64 bit connection requests as if the Destination did not have 64 bit capability.

In a mixed LAN of 32 bit and 64 bit HIPPIs, a 64 bit Source needs to know the data rates available at each Destination and on the path to it. This can be known a priori by manual configuration, or it can be discovered dynamically. The only reliable method of discovery is simply to attempt a 64 bit connection with Camp-on. As long as 64 bit connections succeed, the Source knows the Destination and path are double width. If a 64 bit connection is rejected, the Source tries to connect for 32 bits. If the 32 bit connection succeeds, the Source assumes that the Destination or path is not capable of double width operation, and uses only 32 bit requests after that. If the 32

bit request is rejected, the Source assumes that the Destination or path is down and makes no determination of its capability.

The Double_Wide bit in the HIPPI-LE header, if nonzero, gives the node that receives it a hint that the 64 bit connection attempt may be worthwhile when sending on the return path.

Note that Camp-on must be used at least in the 64 bit attempt, because it removes some ambiguity from the meaning of rejects. If the request is made with the "W" bit and no Camp-on, a reject could mean either that the Destination has no Cable B or that it is simply busy, and no conclusion can be drawn as to its status for 64 bit connections.

Performance

The HIPPI connection rules are designed to permit best utilization of the available HIPPI throughput under the constraint that each Destination must be made available frequently to receive packets from different Sources. This discipline asks both Sources and Destinations to minimize connection setup overhead to deliver high performance. Low connection setup times are easily achieved by hardware implementations, but overhead may be too high if software is required to execute between the initial request of a connection and the beginning of data transfer. Hardware implementations in which connection setup and data transfer proceed from a single software action are very desirable.

HIPPI connections are controlled by HIPPI Sources; a Destination, being unable to initiate a disconnect without the possibility of data loss, is a slave to the Source once it has accepted a connection. Optimizations of connection strategy are therefore the province of the HIPPI Source, and several optimizations are permitted.

If the rate of available message traffic is less than the available HIPPI throughput and Destinations are seldom busy when a connection is requested, connection optimizations do not pay off and the simplest strategy of waiting indefinitely for each connection to be made and sending messages strictly in the order queued cannot be improved upon. However if some nodes are slow, or network applications can send or receive messages at a higher aggregate rate than the available HIPPI bandwidth, Sources may frequently encounter a busy Destination. In these cases, certain host output queuing strategies may enhance channel utilization. Sources may maintain separate output queues for different HIPPI Destinations, and abandon one Destination in favor of another if a connection attempt without Camp-on is rejected or a connection request with Camp-on is not accepted within a predetermined interval. Such a strategy results in

aborted connection sequences (defined in HIPPI-PH: REQUEST is deasserted before any data is sent). Destinations must treat these as normal events, perhaps counting them but otherwise ignoring them.

Two components of connection setup time are out of the control of both Source and Destination. One is the time required for the switch to connect Source to Destination, currently less than four microseconds in the largest commercially available (32 port) switch. The second component is the round trip propagation time of the REQUEST and CONNECT signals, negligible on a standard 25 meter copper HIPPI cable, but contributing a total of about 10 microseconds per kilometer on fiber optic links. HIPPI-SC LANs spanning more than a few kilometers will have reduced throughput. Limited span networks with buffered gateways or bridges between them may perform better than long serial HIPPI links.

A Source is required to drop its connection after the transmission of 68 HIPPI bursts. This number was chosen to allow the transmission of one maximum sized packet or a reasonable number of smaller sized packets. The following table lists some possibilities, with calculated maximum burst and throughput rates in millions (10**6) of bytes per second:

Maximum HIPPI Throughput Rates

User Data	Number of Packets	Number of Bursts	Hold Time (usec)	Burst Rate MB/sec	Max throughput MB/sec Connection Setup Overhead (usec) 10	30	60	90	120	150
63K	1	64	654	98.7	97.2	94.4	90.4	86.8	83.4	80.3
32K	2	66	665	98.6	97.1	94.3	90.4	86.8	83.5	80.4
16K	4	68	667	98.3	96.8	94.1	90.2	86.6	83.3	80.2
8K	7	63	587	97.8	96.1	93.0	88.7	84.8	81.2	77.8
4K	13	65	551	96.7	95.0	91.7	87.2	83.1	79.4	76.0
2K	22	66	476	94.6	92.7	89.0	84.0	79.6	75.6	72.0
1K	34	68	384	90.8	88.5	84.2	78.5	73.5	75.8	65.3

These calculations are based 259 40 ns clock periods to transmit a full burst and 23 clock periods for a short burst. (HIPPI-PH specifies three clock periods of overhead per burst.) A packet of "n" kilobytes of user data consists of "n" full bursts and one short burst equal in length to the number of bytes in the HIPPI, LLC, IP and TCP headers. "Hold Time" is the minimum connection duration needed to send the packets. "Burst Rate" is the effective transfer rate for the duration of the connection, not counting connection switching time. Throughput rates are in megabytes/second, accounting for connection switching times of 10, 30, 60, 90, 120 and 150 microseconds. These calculations ignore any limit on the rate at

which a Source or Destination can process small packets; such limits may further reduce the available throughput if small packets are used.

Sharing the Switch

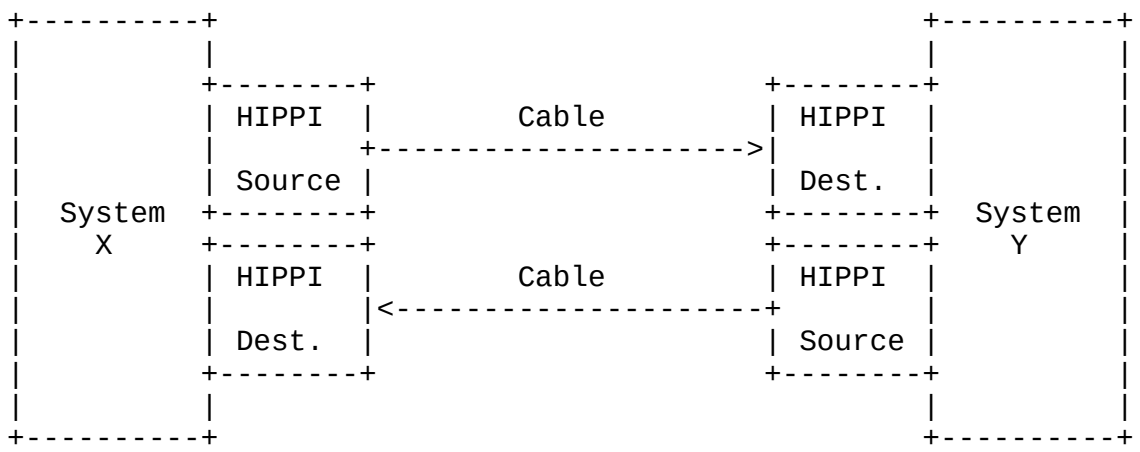
Network interconnection is only one potential application of HIPPI and HIPPI-SC switches. While network applications need very frequent transient connections, other applications may favor longer term or even permanent connections between Source and Destination. Since the switch can serve each Source or Destination with hardware paths totally separate from every other, it is quite feasible to use the same switch to support LAN interconnects and computer/peripheral applications simultaneously.

Switch sharing is no problem when unlike applications do not share a HIPPI cable on any path. However if a host must use a single input or output cable for network as well as other kinds of traffic, or if a link between switches must be shared, care must be taken to ensure that all applications are compatible with the connection discipline described in this memo. Applications that hold connections too long on links shared with network traffic may cause loss of network packets or serious degradation of network service.

Appendix A -- HIPPI Basics

This section is included as an aid to readers who are not completely familiar with the HIPPI standards.

HIPPI-PH describes a parallel copper data channel between a Source and a Destination. HIPPI transmits data in one direction only, so that two sets are required for bidirectional flow. The following figure shows a simple point-to-point link between two computer systems:



A Simple HIPPI Duplex Link

Parallel copper cables may be up to 25 meters in length.

In this document, all HIPPI connections are assumed to be paired HIPPI channels.

HIPPI-PH has a single optional feature: it can use a single cable in each direction for a 32 bit parallel channel with a maximum data rate of 800 megabit/second, or two cables for 64 bits and 1600 megabit/second. Cable A carries bits 0-31 and is used in both modes; Cable B carries bits 32-63 and is use only with the 1600 megabit/second data rate option.

HIPPI Signal Hierarchy

HIPPI has the following hardware signals:

Source to Destination

INTERCONNECT A
INTERCONNECT B (64 bit only)
CLOCK (25 MHz)
REQUEST
PACKET
BURST
DATA (32 or 64 signals)
PARITY (4 or 8 signals)

Destination to Source

INTERCONNECT A
INTERCONNECT B (64 bit only)

CONNECT
READY

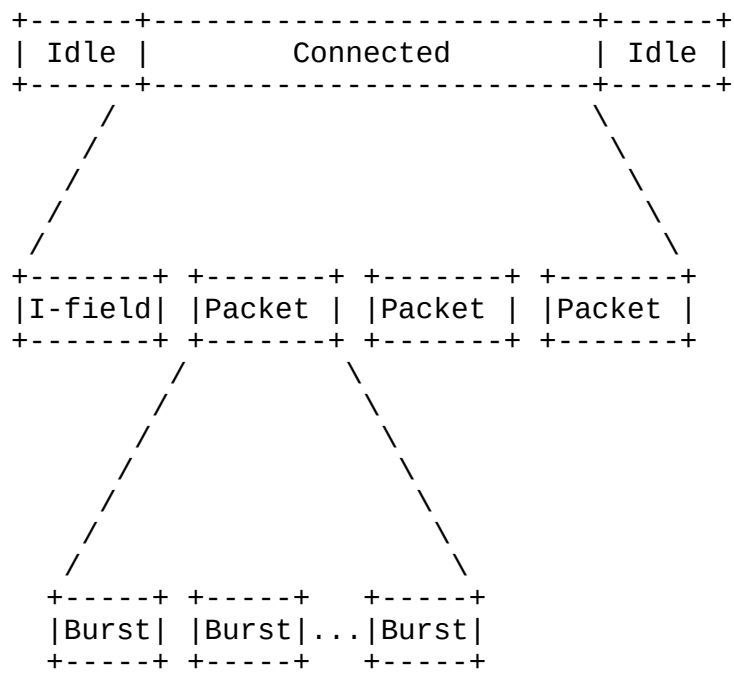
The INTERCONNECT lines carry DC voltages that indicate that the cable is connected and that the remote interface has power. INTERCONNECT is not used for signaling.

The CLOCK signal is a continuous 25 MHz (40 ns period) square wave. All Source-to-Destination signals are synchronized to the clock.

The REQUEST and CONNECT lines are used to establish logical connections. A connection is always initiated by a Source as it asserts REQUEST. At the same time it puts 32 bits of data on DATA lines 0-31, called the I-field. The Destination samples the DATA lines and can complete a connection by asserting CONNECT. Packets can be transmitted only while both REQUEST and CONNECT are asserted.

A Destination can also reject a connection by asserting CONNECT for only a short interval between 4 and 16 HIPPI clock periods (160-640 nanoseconds). The Source knows a connection has been accepted when CONNECT is asserted for more than 16 clocks or it receives a READY pulse.

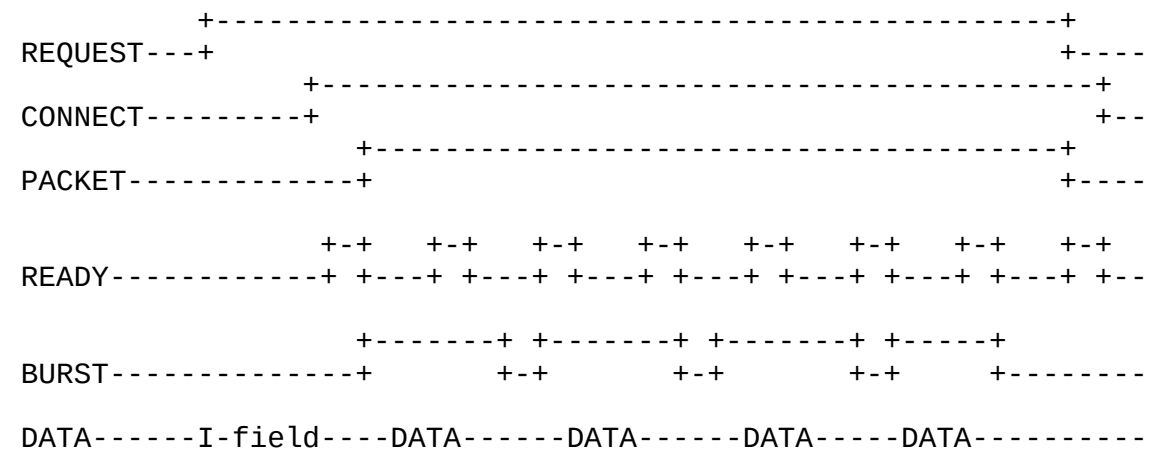
Either Source or Destination can terminate a connection by deasserting REQUEST or CONNECT, respectively. Normally connections are terminated by the Source after its last Packet has been sent. A Destination cannot terminate a connection without potential loss of data.



HIPPI Logical Framing Hierarchy

The Source asserts PACKET for the duration of a Packet transmission, deasserting it to indicate the end of a Packet. A sequence of Bursts comprise a Packet. To send a burst, a Source asserts the BURST signal for 256 clock periods, during which it places 256 words of data on the DATA lines. The first or last Burst of a Packet may be less than 256 clock periods, allowing the transmission of any integral number of 32 or 64 bit words in a Packet.

The READY signal is a pulse four or more clock periods long. Each pulse signals the Source that the Destination can receive one Burst. The Destination need not wait for a burst before sending another READY if it has burst buffers available; up to 63 unanswered READYs may be sent, allowing HIPPI to operate at full speed over distances of many kilometers. If a Source must wait for flow control, it inserts idle periods between Bursts.



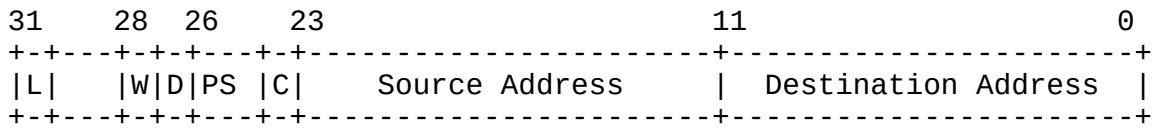
HIPPI Signal Timing Diagram

Serial HIPPI

There is no ANSI standard for HIPPI other than the parallel copper cable version. However an implementors' agreement exists, specifying a serial protocol to extend HIPPI signals on optical fiber or coaxial copper cable. Serial links may be used interchangeably with parallel links to overcome HIPPI distance limitations; they are transparent to the Source and Destination, except for the possibility of longer propagation delays.

I-Field and Switch Control

The REQUEST, CONNECT and I-field features of HIPPI-PH were designed for the control of switches as described in HIPPI-SC. A switch is a hub with a number of input and output HIPPI ports. HIPPI Sources are cabled to switch input ports, and switch output ports are cabled to HIPPI Destinations. When a HIPPI Source requests a connection, the switch interprets the I-field to select an output port and electrically connects the HIPPI Source to the HIPPI Destination on that port. Once connected, the switch does not interact with the HIPPIs in any way until REQUEST or CONNECT is deasserted, at which time it breaks the physical connection and deasserts its output signals to both sides. Some existing switch implementations can switch connections in less than one microsecond. There is the potential for as many simultaneous connections, each transferring data at HIPPI speeds, as there are input or output ports on the switch. A switch offers much greater total throughput capacity than broadcast or ring media.



HIPPI-SC I-field Format (Logical Address Mode)

L = Locally defined (1 => entire I-field is locally defined)
W = Width (1 => 64 bit connection)
D = Direction (1 => swap Source and Destination Address)
PS = Path Selection (01 => Logical Address Mode)
C = Camp-on (1 => wait until Destination is free)

HIPPI-SC defines I-field formats for two different addressing modes. The first, called Source Routing, encodes a string of port numbers in the lower 24 bits. This string specifies a route over a number of switches. A Destination's address may differ from one Source to another if multiple switches are used.

The second format, called Logical Address Mode, defines two 12 bit fields, Source Address and Destination Address. A Destination's 12 bit Switch Address is the same for all Sources. Switches commonly have address lookup tables to map 12 bit logical addresses to physical ports. This mode is used for networking.

Control fields in the I-field are:

L The "Locally Defined" bit, when set, indicates that the I-field is not in the standard format. The meaning of bits 30-0 are locally defined.

W The Width bit, when set, requests a 64 bit connection through the switch. It is meaningless if Cable B is not installed at the Source. If W is set and either the Source or the requested Destination has no Cable B to the switch, the switch rejects the connection. Otherwise the switch connects both Cable A and Cable B if W is set, or Cable A only if W is clear. This feature is useful if both Source and Destination implementations can selectively disable or enable Cable B on each new connection.

D The Direction bit, when set, reverses the sense of the Source Address and Destination Address fields. In other words, D=1 means that the Source Address is in bits 0-11 and the Destination Address is in bits 12-23. This bit was defined to give devices a simple way to route return messages. It is not useful for LAN operations.

PS The Path Selection field determines whether the I-field contains Source Route or Address information, and in Logical Address mode, whether the switch may select from multiple possible routes to the destination. The value "01" selects Logical Address mode and fixed routes.

C The Camp-on bit requests the switch not to reject the connection if the selected Destination is busy (connected to another Source) but wait and make the connection when the Destination is free.

Appendix B -- How to Build a Practical HIPPI LAN

"IP and ARP on HIPPI" describes the network host's view of a HIPPI local area network without providing much information on the architecture of the network itself. Here we describe a network constructed from available HIPPI components, having the following characteristics:

1. A tree structure with a central HIPPI-SC compliant hub and optional satellite switches

2. Each satellite is connected to the hub by just one bidirectional HIPPI link.

3. Serial HIPPI or transparent fiber optic HIPPI extender devices may be used in any link.

4. Some satellites may be a particular switch product which is not HIPPI-SC compliant.

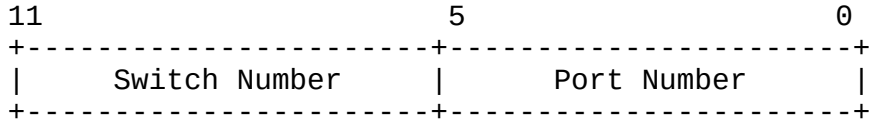
5. Host systems are attached either directly to the hub or to satellites, by single bidirectional links in which both HIPPI cables go to the same numbered switch port.

6. A server system is attached to the hub. It provides multicast and ARP services.

7. All options of the Internet Draft are supported. Hosts may use any form of address resolution: manual configuration, ARP with multicast, or ARP with a server.

Switch Address Management

Switch addresses use a flat address space. The 12-bit address is subdivided into 6 bits of switch number and 6 bits of port number.



Logical Address Construction

Switches may be numbered arbitrarily. A given host's address consists of the number of the switch it is directly attached to and the physical port number on that switch to which its input channel is attached.

In the singly-connected tree structure, there is exactly one path between any pair of hosts. Since each satellite must be connected directly to the hub, the maximum length of this path is three hops, and the minimum length is one. Each HIPPI-SC compliant switch is programmed to map each of the host switch addresses to the appropriate output port: either the port to which the host is directly attached or a port that is linked to another switch in the path to it.

Special Treatment of Nonstandard Switches

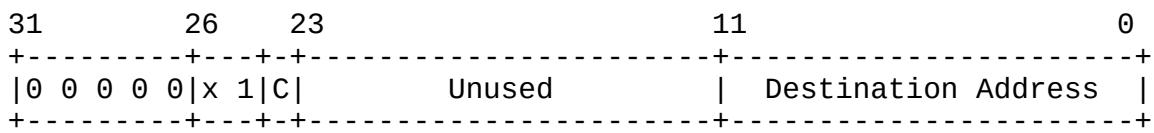
There is one commercially available switch that was designed before the drafting of HIPPI-SC and is not fully compliant. It is in common use, so it is worth making some special provisions to allow its use in a HIPPI LAN. This switch supports only the Source Route mode of addressing with a four bit right shift that can be disabled by a hardware switch on each input port. Addresses cannot be mapped. The switch does not support the "W," "D," or "PS" fields of the I-field; it ignores their contents. Use of this switch as a satellite will require a slight deviation from normal I-field usage by the hosts that are directly attached to it. Hosts attached to standard switches are not affected.

For a destination connected to a non compliant satellite, the satellite uses only the least significant four bits of the I-field as the address. Since the address contains the destination's physical port number in the least significant bits, its port will be selected. Nonstandard switches should be set to disable I-field shifting at the input from the hub, so that the destination host will see its correct switch address in the I-field when performing self-address discovery. I-field shifting must be enabled on the satellite for each input port to which a host is attached.

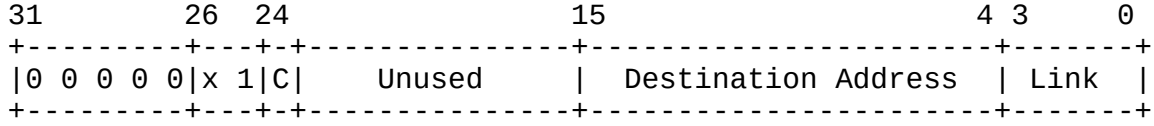
Hosts attached to nonstandard satellites must deviate from the normal I-field usage when connecting to hosts on another switch.

It is suggested that all host implementations have this capability as long as the nonstandard switches remain in use. The host must know, by some manual configuration method, that it is connected to a nonstandard switch, and it must have its "link port" number; that is, the number of the port on the satellite that is connected to the hub.

The normal I-field format for a 32-bit connection, per the Internet Draft, is this:



The special I-field format is:



This I-field is altered by shifting the lower 24 bits left by four and adding the link port number. Camp-on is optional, and the PS field is set to 01 or 11 (the host's option) as if the switch supported logical address mode. All other I-field bits are set to zero. When the host requests a connection with this I-field, the switch selects a connection through the link port to the hub, and shifts the lower 24 bits of the I-field right by four bits. The link port number is discarded and the I-field passed through to the hub is a proper HIPPI-SC I-field selecting logical address mode.

A host on a nonstandard satellite may use the special I-field format for all connection requests. If connecting to another host on the same satellite, this will cause the connection to take an unnecessarily long path through the hub and back. If an optimization is desired, the host can be given additional information to allow it to use the standard I-field format when connecting to another host on the same switch. This information could consist of a list of the other hosts on the same switch, or the details of address formation, along with the switch number of the local satellite, which would allow the host to analyze the switch address to determine whether or not the destination is on the local switch. This optimization is fairly complicated and may not always be worthwhile.

Server Algorithm

Different host implementations may take any of three approaches to address resolution:

1. Manual configuration, no ARP

2. Send ARP requests but expect a server to reply on its behalf

3. Send ARP requests and expect to receive them via multicast.

If the network includes a server that is capable of both multicast and ARP service, and that knows the services expected by each host, all can coexist on the same net.

The HIPPI-SC compliant switches are programmed to route the HIPPI-SC "broadcast" address FE0 (hex) to the server's port. It is initially given the following information by a human network administrator:

1. The list of all addresses eligible to be used by network hosts

2. The list of addresses that should not receive multicast messages (a subset of list 1). This is also the list of all hosts that either do manual configuration or expect a server to answer ARP requests.

3. The list of addresses of hosts that do manual configuration and do not send ARP requests (a subset of list 2) with the IP address corresponding to each one.

The server maintains an address resolution cache that it initializes from list 3 (the manually configured hosts). It will add to its cache as other hosts send ARP requests.

When the server receives a message sent to the broadcast address FE0, it

1. Repeats the message to all addresses in list 1 but not in list 2

2. If the message is a HIPPI-LE AR_Request with a piggybacked ARP Request, update the cache with information about the sender.

3. If the message is a HIPPI-LE AR_Request with a piggybacked ARP Request, the target system has an entry in the cache and the target is in list 2, respond to the ARP request.

Server Optimizations

1. The server could be given a topological map of the hub and satellites from which it could construct list 1.

2. If all the hosts in list 2 ignore ARP messages as required in the Internet Draft, list 2 may be eliminated and the server can respond to all ARP requests (redundant replies may be sent).

Sharing Switch Hardware With Other Devices

Some host channels and peripheral devices that are connected to the switches may use protocols other than IP, and not participate in the LAN. Since connections in a switch are independent, these applications can share switch hardware with no effect on LAN operation. To ensure success:

The server's lists of addresses should not include addresses for ports that are not used by LAN links or hosts.

If non-LAN applications use paths between switches, separate links should be installed for them so that they do not use the same inter-switch links the LAN does.

References

[1] ANSI X3.183-1991, High-Performance Parallel Interface - Mechanical, Electrical and Signalling Protocol Specification (HIPPI-PH).

[2] ANSI X3.210-199X, High-Performance Parallel Interface - Framing Protocol (HIPPI-FP).

[3] ANSI X3.218-199X, High-Performance Parallel Interface - Encapsulation of IEEE 802.2 (IEEE Std 802.2) Logical Link Control Protocol Data Units (802.2 Link Encapsulation) (HIPPI-LE).

[4] ANSI X3.222-199X, High-Performance Parallel Interface - Physical Switch Control (HIPPI-SC).

[5] Postel, J., "Internet Protocol", RFC 791, USC/Information Sciences Institute, September 1981.

[6] Postel, J., and Reynolds, J., "A Standard for the Transmission of IP Datagrams over IEEE 802 Networks", RFC 1042, USC/Information Sciences Institute, February 1988.

[7] IEEE, "IEEE Standards for Local Area Networks: Logical Link Control", IEEE, New York, New York, 1985.

[8] Reynolds, J.K., and Postel, J., "Assigned Numbers", RFC 1340, USC/Information Sciences Institute, July 1992.

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

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

[11] Katz, D., "A Proposed Standard for the Transmission of IP Datagrams over FDDI Networks", RFC 1188, Merit/NSFNET, October, 1990.

[12] IEEE, "Draft Standard P802.1A--Overview and Architecture", 1989.

[13] Mogul, J.C., and Deering, S.E., "Path MTU discovery", RFC 1191, Stanford University, November, 1990.

Security Considerations

Security issues are not discussed in this memo.

Authors' Addresses

John K. Renwick
Cray Research, Inc.
655F Lone Oak Drive
Eagan, MN 55121

Phone: (612) 683-5573

Mailing List: (none)
EMail: jkr@CRAY.COM

Andy Nicholson
Cray Research, Inc.
655F Lone Oak Drive
Eagan, MN 55121

Phone: (612) 683-5473

Mailing List: (none)
EMail: droid@CRAY.COM