Essential Communications Corp.
Network Driver Specification

Document Number 15-0152-00.02

Contributors
Written by Philip Cameron
Illustrated by Philip Cameron
Engineering contributions by Philip Cameron
Reviewers:

Revision History

Rev 00.02 02Dec1996 second review draft
Rev 00.01 29Mar1996 first review draft

Introduction

About this manual...

This manual is the description of the Essential Communications Corp. Network Driver. It provides information that is useful in features of the Network Driver.

What's inside...

Section 1 is an overview of the network driver.

Section 2 describes the driver interface.

Section 3 describes ARP processing.

Section 4 describes ARP table management.

Section 5 ARP proxy features.

Section 6 explains Logical Address discovery.

Section 7 presents the ARP Agent.

Section 8 describes how IP Broadcast operates.

Section 9 presents the IP Broadcast Agent.

Section 10 describes how to operate when two NICs are directly connected.

Section 11 details differences between the network driver and the standards.

Section 12 provides troubleshooting hints.

The Glossary provides definitions of technical terms used throughout the manual.

References lists the relevant standards and RFCs supported by the NIC.

Standards Conformance

The NIC conforms to the network RFCs that govern activity on networks. In particular the driver transfers IP datagrams and ARP datagrams as described in RFC 1374.

Editorial Conventions

Terms that are proper names of signals, state mnemonics, or similar terms are printed in uppercase to avoid possible confusion with other uses of the same words (e.g., REQUEST) Any lowercase uses of these words have the normal technical English meaning.

A number of conditions, sequences, parameters, Events, states, or similar terms are printed with the first letter of each word in uppercase and the rest in lowercase (e.g., Source). Any lowercase uses of these words have the normal technical English meaning.

The Glossary provides definitions for commonly used terms.

Contents

Figures

Section 1
Overview

The Network Driver permits an Essential NIC to participate in a HIPPI network. It encapsulates IP and ARP datagrams in HIPPI packets and transmits the packets over the network. It receives network packets, removes the HIPPI headers and passes the packets to the host ARP and IP stacks.

This section provides a brief overview of HIPPI hardware as it relates to network operations. It next describes the network interfaces for data transfer and configuration and extended features that relate to HIPPI. It ends with a list of utilities that are useful in managing a HIPPI network.

1.1 HIPPI Network Topography

A HIPPI network consists of a set of NICs connected to the HIPPI fabric that are configured to pass IP and ARP datagrams. A HIPPI Fabric consists of one or more interconnected HIPPI switches. NICs are connected to ports on switches that are part of the fabric. Any NIC can send packets to any other NIC in the fabric.

Not all NICs in the HIPPI fabric need to be included in the HIPPI network. Some may be used exclusively by CDI applications and dedicated equipment.

The HIPPI network uses Logical Addressing. Each NIC in the configuration is connected to a port on a switch. The switch administrator assigns a single unique "network" logical address for each port in the HIPPI network. "Network" logical addresses are in the range 0x000-0xF8F.

The network administrator must make sure that logical address 0xFC0 is not mapped to any port in the HIPPI Fabric.

The only switch ports that require a logical address are ports that will be part of the HIPPI network. Other ports may have a logical address assigned but it is not required.

Each HIPPI Fabric should have exactly 1 ARP Agent. The switch administrator maps logical address 0xFE0 to a port on one of the switches. That port is used by a host that can support an Arp Agent. (The drivers for all Essential NICs support ARP and IP broadcast agents). When multiple switches are involved 0xFE0 must be mapped to a single port on one of the switches.

In a similar fashion each HIPPI Fabric should have exactly 1 IP Broadcast Agent. The switch administrator maps logical address 0xFE1 to a port on one of the switches.

In a small HIPPI network the agents can share a NIC with other network activity. In this case a "network" logical address, 0xFE0 and 0xFE1 may be assigned to a single port.

In a large HIPPI network the Broadcast agents may generate a lot of packets. This activity can impact other work that the host system and NIC can perform. It may be desirable to use separate hosts for the agents. These hosts do not have to be part of the HIPPI network (no "network" logical address needs to be defined and the interface does not have to be "ifconfig inet up")

1.2 Network Driver Features

The Essential Network Driver provides data transport services for host system communication IP and ARP stacks. The Network Driver encapsulates datagrams with appropriate HIPPI headers before transmission and accepts received network packets, strips the HIPPI headers and passes the datagrams to the host system communication stacks.

The Network Driver and the Character Driver (as documented separately) can concurrently use the NIC.

The Network Driver also provides HIPPI related services including:

- processing ARP datagrams,
- maintaining ULA to logical address mappings,
- discovering the NIC's logical address,
- providing ARP Agent broadcast services, and
- providing UDP/IP Broadcast Agent services.

IP packet services and ARP resolution services are described in RFC-1374. The driver implements the ARP Broadcast Agent model.

The host communication stacks process IP and ARP datagrams. Internally the datagrams are stored in chains of `mbufs` in the kernel. The Network Driver converts datagrams to HIPPI packets by applying the proper HIPPI headers as described in RFC-1374.

The driver supports up to 16 NICs in the host system and also supports SMP host systems.

1.3 HIPPI Network Hardware Overview

This section describes how the NIC is used to support network processing.

HIPPI is a switched point–to–point connection based network technology. HIPPI provides a "best effort" transport for IP and ARP datagrams. As such it is similar to UDP/IP.

A Source NIC (transmitter) connects to a Destination NIC (receiver). When the connection is established, one packet (encapsulated datagram) is passed over the connection. When the transfer has completed, the connection is terminated. When the Destination is busy (receiving another packet), the sender waits for a connection (CampON).

Each NIC provides a source (transmitter) and a separate destination (receiver). The source and destination are independent and may operate concurrently. Since HIPPI transfers 800,000,000 bits/sec independently on the source and destination, the NIC is capable of transferring 1,600,000,000 bits/sec over the HIPPI media. HIPPI provides a hardware flow control mechanism when the maximum rates cannot be sustained due to host system constraints.

HIPPI packets are routed through HIPPI switches by placing a routing control field, called the CCI field, ahead of the packet. The routing control field contains the destination address and routing control flags. There are two forms of addresses "Logical" and "Source". Logical addressing identifies a destination with a 12 bit address. The Switch is responsible for establishing the physical route through the switch. Source routing specifies the physical path from the source to the destination. The network driver supports "Logical" addressing. It does not support transmission using source "Source" addressing. It accepts packets that have been sent using "Source" routing.

A HIPPI network consists of a HIPPI fabric and a set of HIPPI end points. The fabric consists of a collection of interconnected HIPPI switches. It can be viewed as a "cloud" that handles passing packets between end points. HIPPI end points are HIPPI NICs in host computer systems and other peripherals.

The HIPPI fabric assigns (by means of administrator intervention) a unique logical address to each End Point that can pass network packets. It may also assign logical addresses to other End Points.

HIPPI media may be either copper cables or optical cables. Essential provides NICs that are based on the RoadRunner ASIC with optical media connections. Essential also provides HIPPI switches that can be configured with a combination of optical and copper media cards.

HIPPI hardware does not support broadcast or multicast. However, the HIPPI-SC switch standard reserves certain logical addresses for pseudo-broadcast mechanisms. Broadcast is simulated using the ARP Agent and the IP Broadcast Agent as described below.

1.4 Network Interface

The Network Driver is a typical network driver for the target host systems. The programming interface is the Unix standard "sockets" interface. The configuration is managed by "ifconfig", statistics are reported using "netstat". ARP and Published ARP entries and ARP server are supported and managed using an extended "arp" utility, "essarp".

An administrator who is experienced in managing Ethernet and other networks will have little difficulty in managing the HIPPI network interface.

1.5 Extended Features

The driver supports HIPPI network services that are not found in other network technologies, namely address discovery and psuedo-broadcast.

The HIPPI switch logical address is used in routing packets through the HIPPI fabric. The logical address of the NIC can be configured to match the switch configuration or automatically determined by an address discovery algorithm.

HIPPI switches do not directly support broadcast. However there are reserved logical addresses that may be configured. ARP packets may be sent to Logical address 0xFE0 for pseudo-broadcast to all hosts that are supporting network operations. The pseudo-broadcast consists of sending a copy of the packet to each known target machine. In a similar fashion UDP/IP broadcast packets are sent to logical address 0xFE1. The HIPPI fabric administrator configures the reserved addresses to specific host systems.

The Network driver supports pseudo-broadcast for both ARP and UDP/IP broadcast. The support is always present and active whether or not the NIC is configured or is assigned a logical address. Whenever a broadcast packet arrives the appropriate broadcast agent broadcasts it.

1.6 Utilities

The host system provided "arp" utility is extended to support HIPPI functions. The utility, "essarp", supports the expected "arp" functions (and calls "arp" as needed). The extensions support features that are described in the remainder of this document. See the man page for further details.

The "esshippi" utility controls the state of the NIC. RunCode must be loaded into the NIC and started, and the optical link must be ON before the Network Driver can become active. esshippi cannot change operating mode while the interface is "ifconfig up". See the man page for further details.

The "essdb" utility controls verification of received packets. When verification is set, the headers on each received network packet are validated. See the man page for further details.

Section 2
Network Interface

The programming and administrator interfaces are very similar to existing network interfaces. The programming interface is the standard Unix sockets interface. The primary administrator interface is "ifconfig" and "netstat" (two standard Unix utilities).

2.1 Application Network Interface

The programming interface is the standard Unix "Sockets" interface. The interface supports networks with different capabilities. The Essential Network driver supports:

- MTU of 65288 bytes,
- UDP/IP broadcast,
- ARP broadcast, and
- AF_INET protocol family.

The Network Driver does not support:
- Multicast

2.2 System Network Interface

The "ifconfig" utility is used to configure the interface. "netstat" may be used to view the configuration and operating statistics. Each host system has a different set of configuration flags. See the following sections for details.

The NICs device name is different for the various host systems. The last character of the name is a digit which represents the card number as assigned by the host system. At present there is support for 1 to 16 NICs in a host system. "netstat -i" shows the interfaces that are configured in the host system.

The following ifconfig flags are not supported:
- PROMISC
- MULTICAST

The driver supports ARP. The NOARP flag prevents host generated ARP messages from being sent. It also discards received ARP messages.

The RUNNING flag is set when all of the following are true:
- The RunCode (firmware) is running on the NIC,
- The Optical Link is ON,
- The logical address of the NIC is known,
- The NIC is accepting connections, and
- The NIC is configured for HIPPI-FP operation.

The "esshippi" utility is used to start firmware, configure the NIC to accept connections and configure the NIC to operate in HIPPI-FP mode. The RunCode determines the state of the Optical Link. The mode can only be changed when the interface is "ifconfig down".

2.2.1 System Network Interface - Digital Equipment

The interface name is "hip" and the system assigns a unit number for each NIC in the configuration starting at 0. The full name of the first NIC is "hip0".

The following flags are set by default:
- BROADCAST
- NOTRAILERS
- SNAP
- SIMPLEX

2.2.2 System Network Interface - Hewlett Packard

The interface name is "ess" and the system assigns a unit number for each NIC in the configuration starting at 0. The full name of the first NIC is "ess0".

The following flags are set by default:
- BROADCAST
- NOTRAILERS

2.2.3 System Network Interface - Silicon Graphics

The interface name is "ess" and the system assigns a unit number for each NIC in the configuration starting at 0. The full name of the first NIC is "ess0".

The following flags are set by default:
- BROADCAST
- NOTRAILERS

2.3 getsockopt and setsockopt

The getsockopt and setsockopt system calls support different commands for different host systems as described below. Each host system supports a somewhat different set of commands.

2.3.1 getsockopt and setsockopt - Digital Equipment

SIOCSIFADDR	- Set interface address
SIOCSIFFLAGS	- Set interface control flags
SIOCENABLBACK	- Internal loopback (Not supported, use esshippi on loopback)
SIOCDISABLBACK	- Internal loopback (Not supported, use esshippi on loopback)
SIOCRPHYSADDR	- Return the NIC ULA
SIOCADDMULTI	- Not supported
SIOCDELMULTI	- Not supported
SIOCSPHYSADDR	- Not supported
SIOCRDZCTRS	- Not supported
SIOCRDCTRS	- Not supported
SIOCIFRESET	- Not supported

SIOCSIPMTU	- Supported

2.3.2 getsockopt and setsockopt - Hewlett Packard

SIOCSIFADDR	- Set interface address
SIOCSIFFLAGS	- Set interface control flags
SIOCGACFLAGS	- Get interface arpcom flags
SIOCSACFLAGS	- Set interface arpcom flags
SIOCSIFMETRIC	- Set interface metric
SIOCSIFBRDADDR	- Set broadcast address
SIOCSIFNETMASK	- Set network address mask

2.3.3 getsockopt and setsockopt - Silicon Graphics

SIOCSIFADDR	- Set interface address
SIOCSIFFLAGS	- Set interface control flags
SIOCSIFMETRIC	- Set interface metric
SIOCSIFBRDADDR	- Set broadcast address
SIOCSIFNETMASK	- Set network address mask

2.4 Network Driver Operation

Datagrams are passed between the Communication Stack (in the host system) and the driver using a chain of `mbufs`. The details vary from system to system and are described below.

The network driver accepts an outgoing datagram (`mbuf` chain) and places a HIPPI header ahead of the datagram. The resulting HIPPI packet is queued along with CDI packets for transmission.

The HIPPI header consists of an CCI field, a HIPPI-FP header, a HIPPI-LE header and a SNAP/LLC header.

The HIPPI-FP specification reserves a target ULP (upper layer protocol), 4, for processing network packets. All transmitted network packets target ULP 4. The RunCode places received network packets in `mbufs` that are associated with ULP 4.

The received datagram is analyzed. Packets that are directed to the host are handed to either the IP stack or ARP stack. Packets that are directed to logical address 0xFE0 are handed to the ARP broadcast agent and packets that are directed to logical address 0xFE1 are handed to the IP broadcast agent. All other network packets are discarded.

2.4.1 Network Driver Operation -Digital Equipment

Datagrams are sent by passing the `mbuf` chain that contains the datagram to the driver output function. The driver output function prepends an `mbuf` to the chain. It next formats the HIPPI header and places the packet on the output queue. The low level driver takes the packet from the queue and transmits it.

The Digital Unix system is a little endian system so the HIPPI headers are converted to network order which is big endian. The data is not converted to big endian.

The low level driver places packets directed to ULP 4 into "cluster `mbufs`" which are a page long (8192 bytes). When a whole packet is received, the HIPPI header is stripped from the first `mbuf` and datagram is passed to the network driver.

The network driver processes the datagram as described in section 2.4.

2.4.2 Network Driver Operation - Hewlett Packard

Datagrams are sent by passing the `mbuf` chain that contains the datagram to the driver output function. The driver output function prepends an `mbuf` to the chain. It next formats the HIPPI header and places the packet on the output queue. The low level driver takes the packet from the queue and transmits it.

The RunCode computes the needed send checksums for the IP header and data before transmitting the the packet. This performance optimization relieves the host system of having to compute checksums.

The low level driver places packets directed to ULP 4 into cluster `mbufs`. The header is placed in the first `mbuf` in the chain and the data is placed in subsequent `mbufs`. The separation of the header from the data permits the HP-UX page flip feature to place the data in user space simply by adjusting kernel memory mapping tables (no copying is necessary). The RunCode also calculates and validates the IP and data checksums which relieves the host of having to compute checksums. When a whole packet is received, the HIPPI header is stripped from the first `mbuf` and datagram is passed to the network driver.

The network driver processes the datagram as described in section 2.4.

2.4.3 Network Driver Operation - Silicon Graphics

Datagrams are sent by passing the `mbuf` chain that contains the datagram to the driver output function. The driver output function prepends an `mbuf` to the chain. It next formats the HIPPI header and places the packet on the output queue. The low level driver takes the packet from the queue and transmits it.

The low level driver places packets directed to ULP 4 into cluster `mbufs`. When a whole packet is received, the HIPPI header is stripped from the first `mbuf` and datagram is passed to the network driver.

The network driver processes the datagram as described in section 2.4.

Section 3
ARP Operations

ARP is resolved as follows:

When the ARP stack needs to find the address (ULA) of a host that is not currently in its cache, it generates an ARP datagram and passes it to the Network Driver.

The Network Driver places a HIPPI network header ahead of the datagram. The HIPPI header has a HIPPI-LE M_Type of AR_Request. The Source Address is the logical address of the NIC interface and the Destination logical address is 0xFE0. The Source ULA field is filled in with the NIC ULA. The Destination ULA is the broadcast ULA FF:FF:FF:FF:FF:FF.

When an Essential Network Driver is serving as the ARP Agent the following occurs.

The ARP Request Packet arrives at the ARP Agent. The ARP Agent maintains a cache of all logical addresses that can accept an ARP message. It sends a copy of the ARP message to all logical addresses in its table using CampON in the HIPPI-SC CCI field. When a connection is rejected and the table entry is type "temporary", the table entry is deleted.

The ARP Agent replaces the Destination switch logical address (which is either 0xFE0 or undefined) with the target logical address from the table. If the ULA of the destination is known it is placed in the header, otherwise the ULA is set to 0:0:0:0:0:0.

When a host receives an ARP packet (either a request from the ARP Agent or a reply from a remote host), the source logical address and source ULA are added to the driver cache. When the message is AR_Request or AR_Response, the source IP address is also added to the table. The Network Driver strips the HIPPI header and passes the ARP datagram to the host ARP stack.

There are two classes of ARP response datagrams as described below.

Many hosts broadcast an ARP Response when the interface is configured, and also when the logical address of the NIC changes. This ARP Response has a source and destination that is the originating host. Since ARP is broadcast to all hosts this is a way of notifying all hosts that the interface has been configured. The HIPPI interface sends this class of ARP response to the ARP Agent for broadcast.

The second class of ARP response is a datagram in response to an ARP request. The HIPPI interface sends this class of ARP response directly to the destination host (it is not sent to the ARP Agent).

All ARP AR_Request and AR_Response packets are sent with CampON set in the HIPPI-SC CCI field.

The ARP Agent strives to keep its logical address table accurate by:
- Updating the table with information in each received ARP packet,
- Deleting (temporary) entries when connections are rejected, and
- Performing address discovery on a periodic basis (every 5 minutes).

The NIC must be configured, by esshippi, to be in HIPPI-FP mode for the Agent to operate. There does not have to be a logical address assigned to the NIC and the NIC does not have to be configured by "ifconfig".

Section 4
ARP Tables

The host system manages ARP tables that map a network IP address to a hardware (ULA) address. The host system also manages the published addresses. The driver manages a ULA to logical address mapping table that permits the logical address of a target to be found given the ULA.

Unix ARP manages the cache of IP to ULA mappings with minimal administrator intervention. The system originates ARP request messages when it needs information about a remote host. The Network Driver formats the ARP request into a HIPPI-LE ARP request and sends it to the ARP agent at logical address 0xFE0. When any of the HIPPI-LE ARP messages arrive at the NIC, the ULA and Logical address mapping information is placed in the driver cache as temporary entries. ARP request and response datagrams are passed to the host ARP stack.

When the administrator has to create a mapping, the "arp" utility is used. When a table entry for the HIPPI interface needs to be managed, the administrator uses the "essarp" utility instead of "arp" ("essarp" calls "arp"). In addition to the usual host name and ULA required by "arp -s", "essarp -s" also specifies the target NIC logical address. The table entries are permanent unless the command includes the "temp" option (e.g., essarp -s <host> <ULA> <Logical_address> temp).

Permanent and temporary entries are deleted using "essarp -d <host>".

The driver mapping table may be cleared of all temporary entries using "essarp -c".

The hardware (ULA) address is optional on HIPPI and some HIPPI end points do not support it. All Essential NICs are assigned a unique ULA which is used by the Network Driver. Whenever the ULA of a remote system is undefined or unknown, the driver sets up a local ULA for the NIC.

Section 5
Published ARP

Some host systems do not support ARP over certain interfaces. These systems cannot generate or receive ARP datagrams. The systems can be supported in a network as follows.

When the host does not support ARP over an interface, the administrator uses the "arp" utility to statically configure the ARP table to publish needed targets (this is also called Proxy ARP). The host system will have no need to generate ARP messages.

Other host systems that need to communicate with a system that does not support ARP can do so in one of two ways: use "arp" to statically configure the needed entry (since an ARP request will be ignored), and configure a host on the network to Publish ARP for the system that does not support ARP. The first case requires the administrator to use "arp" to configure every host that needs to communicate with the target system. The second approach permits the administrator to configure a single host on the network to publish ARP for the target system.

Since ARP is broadcast all ARP request messages are received by all hosts. The host that is publishing for another host responds to the ARP request that are directed to the other host as if the other host is responding. This processing is done in the host system ARP stack.

The "arp -s <host> <ULA> pub" command is used to publish ARP for a host. When the command is issued, the host sends an ARP response to all hosts that is formatted as if it were sent by the target host. The publishing system listens for ARP requests for the published system and generates an ARP response whenever a request arrives.

When publishing for a HIPPI interface, use "essarp" instead of "arp" and provide the logical address of the target.

Section 6
Address Discovery

The Network Driver must know the logical address of the resident NIC in order to format packets for transmission. Until the Logical address of the NIC is known, the Network Driver will not accept datagrams from the host communication stack and will not accept network packets from the network . (It will, however, support ARP Agent and IP Broadcast Agent activity).

The NIC logical address is invalid when any of the following is true:
• System is booted,
• When "essarp -i" (invalidate logical address) is run, or
• When discovery fails to find a logical address.

The NIC logical address is verified by sending a HIPPI-LE ARP_S_RESPONSE packet to itself. If the packet is received by the NIC the logical address is valid and discovery is complete.

When the logical address is not known, when the logical address is known and a "LINK_ON" occurs, or when the verification fails, the driver attempts to discover the NIC logical address by sending a HIPPI-LE ARP_S_RESPONSE message to all of the unreserved logical addresses (0x000-0xF8F inclusive). The last Response message to be received becomes the logical address of the NIC.

Address verification and discovery is done in the following circumstances:
• "ifconfig up",
• When "essarp -t" is run,
• When Network Driver is "UP" and "Link ON" occurs, and
• Every 3 minutes during normal operation.

When discovery succeeds in discovering the NIC logical address and the IP address has been configured for the interface, an ARP response message for the interface is sent to the ARP Agent at 0xFE0 for broadcast. In a similar fashion HIPPI-LE ARP_S_REQUEST messages are sent to the ARP and IP broadcast agents. The agents respond with a ARP_S_RESPONSE message that includes the host name of the agent.

There are HIPPI devices that don't handle HIPPI-LE ARP messages. The administrator can configure the driver to not generate ARP messages to specific logical addresses by using the "Do not disturb" flag (dnd) in the "essarp -s <host> <ULA> <LA> dnd". The <host> and <ULA> field can contain anything since they are not used. The dnd logical addresses must be configured before address discovery is attempted.

Section 7
ARP Agent

The ARP Agent accepts ARP Request and Response packets that are directed to Logical address 0xFE0. It sends a copy of the packet to all know HIPPI end points that are not marked "Do not disturb" (dnd). After the ARP packet is sent to the last end point in the table the packet is discarded and processing continues with the next ARP packet.

The ARP agent is part of the Essential Network driver and is always active. The HIPPI fabric administrator configures one HIPPI endpoint to serve as the ARP Agent by assigning logical address 0xFE0 to the desired physical port and configuring all switches in the fabric to map logical address 0xFE0 to that port. The administrator can reconfigure the ARP Agent to a different host (that is running the Essential Network Driver) at any time.

When the first packet arrives for the ARP Agent, the Agent performs a discovery to update the driver cache table. The driver cache table is kept up to date by all received ARP datagrams. Additionally, the ARP Agent verifies that it is still the ARP Agent and performs discovery every 5 minutes. When a connection is rejected, the (temporary) table entry is deleted.

Section 8
IP Broadcast Operations

When an IP datagram (Ether_Type 0x800) with a destination broadcast address (255.255.255.255) and destination broadcast ULA (FF:FF:FF:FF:FF:FF) is passed to the Network Driver, the packet is sent to the IP Broadcast Agent at logical address 0xFE1.

When the packet arrives it is placed at the end of the queue of packets to be broadcast. The IP broadcast operates the same as the ARP Agent (see Section 7).

When a NIC is configured "up" or a new logical address is discovered, the host sends a HIPPI-LE ARP_S_Response message to the IP Broadcast Agent so the Agent can update its tables. The ARP_S_Response message is discarded (it is not broadcast).

When the first packet arrives for the IP Broadcast Agent, the Agent performs a discovery to update the driver cache table. The driver cache table is kept up to date by all received ARP datagrams and all received broadcast requests. Additionally, the IP Broadcast Agent verifies that it is still the IP Broadcast Agent and performs discovery every 5 minutes. When a connection is rejected, the (temporary) table entry is deleted.

The NIC must be configured, by esshippi, to be in HIPPI-FP mode for the Agent to operate. There does not have to be a logical address assigned to the NIC and the NIC does not have to be configured by "ifconfig".

Section 9
IP Broadcast Agent

The IP Broadcast Agent accepts UDP/IP packets that are directed to Logical address 0xFE1. It sends a copy of the packet to all known HIPPI end points that are not marked "Do not disturb" (dnd). After the UDP/IP packet is sent to the last end point in the table the packet is discarded.

The IP Broadcast Agent is part of the Essential Network driver and is always active. The HIPPI fabric administrator configures one HIPPI endpoint to serve as the IP Broadcast Agent by assigning logical address 0xFE1 to the desired physical port and configuring all switches in the fabric to map logical address 0xFE1 to that port. The administrator can reconfigure the IP Broadcast Agent to a different host (that is running the Essential Network Driver) at any time.

When the first packet arrives for the IP Broadcast Agent, the Agent performs a discovery to update the driver cache table. The driver cache table is kept up to date by all received UDP/IP datagrams and all HIPPI-LE ARP packets. Additionally, the IP Broadcast Agent performs discovery every 5 minutes. When a connection is rejected, the (temporary) table entry is deleted.

Section 10
Direct Connect

When two NICs are directly connected instead of being connected through a HIPPI-SC switch, there is no need to perform discovery and no need for the ARP and IP Broadcast Agents to operate. All packets are passed from the Source of one NIC to the Destination of the other NIC.

The driver uses reserved Logical Address 0xFC0 to assist in discovering whether or not two NICs are directly connected. Whenever discovery is performed an ARP_S_REQUEST is sent to logical address 0xFC0. When the NIC is connected to a switch, the packet is rejected. When directly connected, the packet is delivered to the remote NIC. When a NIC receives a ARP_S_REQUEST from 0xFC0, it sets IS_DIRECT (see esshippi status man page) and sends a ARP_S_RESPONSE to 0xFC0. When this is received by the originator, the originator sets IS_DIRECT.

When directly connected, the Network driver may be configured to prevent unneeded processing by either running "essarp -l <logical_address>" or by running "esshippi on direct".

The driver may be returned to "switched" operation by "essarp -i" or "esshippi on switched".

"esstune" may be used to permanently set "direct" or "switched" mode as the default mode for the driver. See the man page for further details.

Section 11
Deviations from Standards

The following differences exist between the Essential Network Driver and the governing standards.

- No support for transmission using Source addressing. All addresses are Logical Addresses
- The FP D2_Offset is always 0. This applies to both Source and Destination
- The FP W bit is always 0. This is a "Single Wide" implementation
- The FP D1 area is always 3. The HIPPI-LE header is the only data in the D1 area
- The FP B field is always 0 at Source. There is no short first burst

- The ARP Agent does not support the ARP Server model described in RFC-1374

Section 12
Troubleshooting

This section covers configuration details that are needed to properly use the network driver.

Before doing the following make sure that the NIC is operating properly and has been correctly configured into the switch as described in the troubleshooting section of the "CDI Users Guide and Reference".

12.1 HIPPI Network

Verify that each NIC in the HIPPI Network has a single "network" logical address in the range 0x000-0xF8F.

Verify that every NIC can send packets to every other NIC in the network using the assigned logical address. (The "blast" and "sink" utilities are useful to verify this. See the CDI Users Guide and Reference for details.)

Verify that there is one 0xFE0 and one 0xFE1 mapping in the network.

Verify that all NICs can send packets to 0xFE0 and 0xFE1.

Verify that the 0xFE0 and 0xFE1 agents can send packets to all NICs in the network.

Verify that Logical address 0xFC0 is not mapped to any port in the Fabric.

12.2 `essd` Daemon

The "essd" daemon must be running. Use "ps" to verify that it is running. It should be uninterruptable in the kernel.

12.3 NIC Configuration

Each NIC in the configuration must have the following characteristics:

- RunCode must be running
- Link must be ON
- NIC must be in "fp" mode

```
# esshippi status
```

will report the current operating status.

12.4 NIC Identification

Each NIC in the configuration is given a unit identification during system bootup. The interfaces are listed by:

```
# netstat -i
```

Each interface is separate and unique. Each interface must be individually configured.

12.5 Default Network Interface

When multiple interfaces are configured in the system the network daemons will usually use a default interface for configuration information (DNS and NIS queries among other tasks). The default interface is usually the first interface that is discovered during system bootup.

```
# netstat -i
```

lists the interfaces and usually the first in the list is the default device.

Unless the HIPPI NIC is the desired default device (usually when the HIPPI device is the only device), the physical configuration should be changed so that the desired interface is the default. This usually involves changing the slots that are occupied by the various network interface cards (NICs).

12.6 Host Name

In configurations that have a single interface card it is convenient to use the domain name as the host name as it reduces confusion in people identifying the machine and the network identifying the machine.

When multiple interfaces are in the system, each interface has a different name. When communicating over the network the desired domain name must be used, not the host name.

It is very common for the default interface and host name to be the same.

12.7 IP and netmasks

An IP field is 32 bits long. Each of the 4 bytes is represented as a decimal value and the fields are separated by "."s (e.g. 199.89.135.16).

An IP address is divided into two fields: network and unit. The network field identifies the network that contains the unit. The network field can indicate three classes of network. Each class has a different number of bits in the network field. Class A (8-bits), class B (16-bits) and class C (24 bits). For each class the remaining bits are the unit field. Class A has a 24 bit unit field, class B has a 16 bit field and class C has an 8 bit field.

Internet administrators assign IP networks to companies and organizations based on the expected number of units that are expected to be assigned to the network. It is most common to acquire a class C IP address which permits 254 units. (unit 0 is reserved and unit 255 is the broadcast unit for the network).

The local network administrator can divide the assigned IP address into 1 or more sub-nets by use of the netmask field.

The netmask is a mask to "and" with the IP to get the network field. The netmask is represented the same as an IP address. In a netmask, the "left" side identifies the network (which is indicated by setting all of the bits in the field). The unit is the "right" side of the netmask with all bits reset.

The "default" netmask for each class is: class A 255.0.0.0, class B 255.255.0.0, and class C 255.255.255.0. The netmask field permits the use of some of the unit bits as network bits. This permits an administrator to create sub-nets within the assigned IP. For example, an administrator of a class B network can treat the class B network as 256 class C networks by using a netmask of 255.255.255.0 instead of the default 255.255.0.0.

There are two reserved unit numbers for each network: 0 (which is not used) and all bits set which is the broadcast unit for the network.

12.8 Domain Names

The Internet administrators assign a domain of names for companies and other organizations. For example, esscom.com is the domain name for Essential Communications Corp. Actually "esscom" is a sub-domain of "com". The network administrators at esscom can provide additional sub-domains as needed. It may be convenient to assign a sub-domain for the HIPPI network.

Each domain is assigned to an IP address (as described above).

12.9 Domain and IP Mappings

The domain name and corresponding IP addresses must be known for each NIC. For an organization it is common to use DNS (Domain Name Server) on one of the host systems. Alternatively, NIS and the /etc/hosts files may be used.

When DNS is used it is common for each host to have an /etc/resolv.conf file that points to the server.

In any event, each NIC must be accessible by both domain name and IP address. "nslookup" should be able to find all NICs by name or IP.

```
sandia> nslookup shawmut-h
Server:  micmac.esscom.com
Address:  199.0.65.2

Name:    shawmut-h.esscom.com
Address:  198.17.163.71

sandia> nslookup 198.17.163.71
Server:  micmac.esscom.com
Address:  199.0.65.2

Name:    shawmut-h.esscom.com
Address:  198.17.163.71

sandia>
```

12.10 Networks and Routing

All units in a network are directly accessible to each other. A message broadcast on the network will be received by all units in the network. In the context of IP addresses, for every IP address, the IP address "and" the netmask is the same (the netmask must be the same for all units in the network).

Units on other networks are accessed by passing through a gateway or bridge. Datagrams to other networks are sent to an appropriate gateway or router machine. The gateway machine receives the datagram and forwards it to another network. This process is continued until the datagram reaches the final destination (or it can't be delivered).

A default route for the host can be established using:

route add default <gateway_IP>

Alternatively, gated or routed can be used to dynamically determine routing. gated and routed use the RIPP routing protocol which is a broadcast UDP/IP protocol.

It is frequently convenient to assign all HIPPI devices in a HIPPI fabric to a single network.

12.11 Assigning IP, Domain Names and Netmasks

Each interface is configured using ifconfig (see below).

Before using "ifconfig" the following should have been done:

- Assign a name to each NIC in the HIPPI network
- Assign an IP address to each NIC in the HIPPI network
- Assign the netmask to each NIC in the HIPPI network
- Verify that all of the names are in the same domain and all IP addresses are in the same network
- Configure the information into whatever is serving the domain (/etc/hosts, DNS or NIS)
- From each host in the network use "nslookup" to verify the above name and IP mappings

In a multiple NIC configuration (e.g., Ethernet NIC and a HIPPI NIC), each NIC that is configured must have a different network address. IP "and" netmask is different for each NIC.

12.12 Interface Configuration, IP, and netmasks

Each interface is configured using ifconfig. ifconfig assigns an IP address and netmask to the NIC. Until the IP address and netmask are assigned, the NIC cannot be used as a network interface.

The actual "ifconfig" calls are usually in a system boot script. A typical "ifconfig" follows.

ifconfig hip0 inet 199.23.116.4 netmask 255.255.255.0 up

is a typical configuration command for a NIC.

hip0 is the name of the interface as shown by "netstat -i".
inet sets the interface IP (in our case 199.23.116.4, some hosts permit the name to be used here)
netmask sets the network mask (in our case 255.255.255.0)
up configures the interface up

"up" will perform an address discovery.

12.13 ARP Agent

Each HIPPI Fabric should have exactly 1 ARP Agent. See section 12.1.

12.14 IP Broadcast Agent

Each HIPPI Fabric should have exactly 1 IP Broadcast Agent. See section 12.1.

12.15 Testing the Configuration

After all NICs have been "ifconfig"ed, "ifconfig <interface>" should report the interface to be "up" and "running" and the proper IP address and netmask should be displayed.

"essarp -h" should report the ULA and Logical Address that are assigned to the NIC.

Next try to "ping" each HIPPI NIC from each host.

Annex - A
HIPPI-SC Excerpts

This is excerpted from the HIPPI-SC specification; for more information refer to the specification. When the HIPPI-SC option is set the Host Software must construct valid CCI fields for proper send operation through a HIPPI switch (see figure A-1). The detailed settings must be determined after analysis of the switch configuration.

The HIPPI-SC CCI field describes the CCI field as it appears on the HIPPI media. In host memory and in the NIC the CCI field is 8 bytes long. This permits the packet to be 8 byte aligned in an 8 byte or larger buffer (which is important for 64 bit CPUs). The first 4 bytes of the extended CCI contain 0. The next 4 bytes are as described below.

The CCI is in Big Endian on the media. Little Endian Hosts must byte swap.

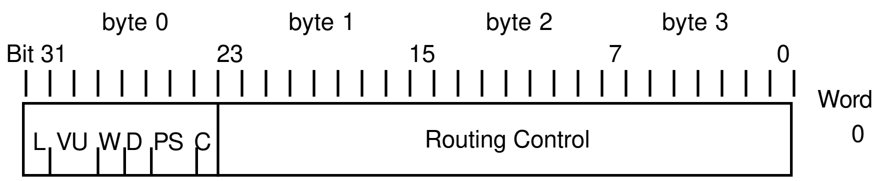

Figure A-1 CCI and I-Field Format

L - Locally administered (bit 31) L=0 designates that the I-Field is defined by the standard. L=1 designates that rest of the bits (30-0) are locally defined and are not defined by the standard.

VU - Vendor Unique (bits 30-29) - The contents of these bits are not defined by the standard. Switches pass these bits through to the Destination (use 00).

W - Double Wide (bit 28) - W=0 designates that the source is using the 800 Mbit/sec data rate option. The Essential NIC requires that this bit be set to 0.

D - Direction (bit 27) - D = 0 designates that the right-hand end (least significant bits) of the routing control field shall be the current sub-field. D=1 designates that the left-hand end (most significant bits) of the routing control field shall current sub-field.

PS - Path Selection (bits 26-25) - Used to select either (1) a source route, i.e., a specific route through the switches, with output port numbers specified for each switch, or (2) to specify the Logical Address.

PS=00 - source routing Source selects the route through the switches.
PS=01 - Logical address: Switches select the first route from a list of possible routes.
PS=10 - reserved
PS=11 - logical address: Switches select a route.

C - Camp-on (bit 24) - C=0 specifies that the switch shall reply with a connection reject sequence if unable to complete the connection. C=1 specifies that the switch shall attempt to establish a connection until either the connection is completed or the Source aborts the connection request.

Note that the Essential NIC requires that the W bit be 0.

ANNEX - B
HIPPI-FP Excerpts

This is excerpted from the HIPPI-FP specification for more information refer to the specification. When the HIPPI-FP option is set the Host Software must construct valid FP headers for proper send operation through the NIC. Fields in this header are used by the NIC to direct processing. The detailed settings must be determined after analysis of processing requirements.

The header is always in Big Endian on the media. Little Endian Host systems must perform the needed byte swaps.

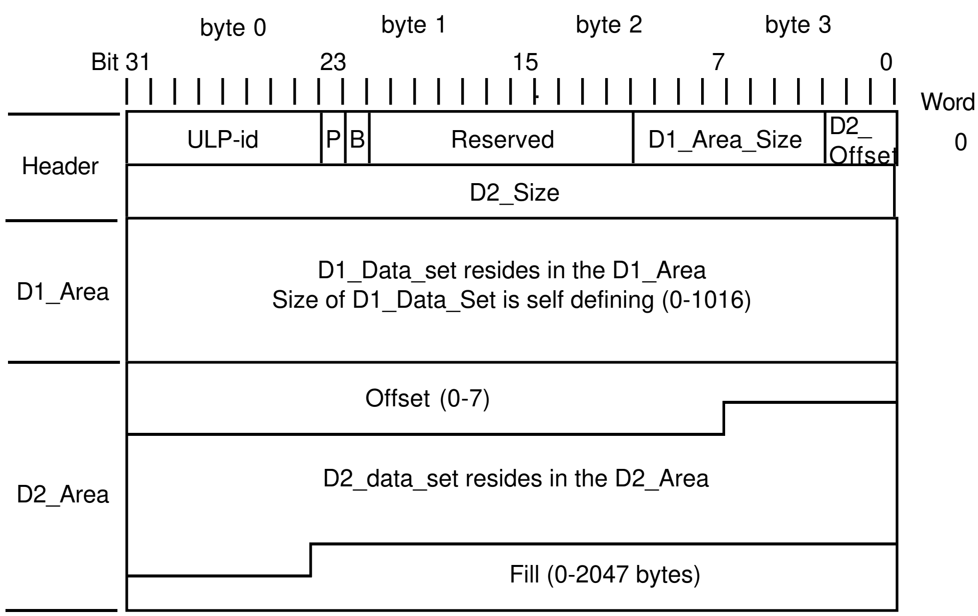

Figure B-1 FP Packet Format

The packet data presented by the Source ULP with a FP_TRANSFER Request primitive shall be transferred to the Destination with a HIPPI-FP header as shown in figure B-1. The HIPPI-FP packets shall be composed of three areas, (1) Header, (2) D1_Area, and (3) D2_Area, each starting and ending on a 64 bit boundary. If the D1_Data_Set is used as control information, the D2_Data_Set is intended as the data associated with that control information.

Header

The Header shall be the first 64 bits of the packet, and shall be completely contained in the first burst.

ULP-id - (8-bits) designates the Destination ULP to which the packet is to be delivered. Undefined ULP values are reserved. The following are defined:
00000010b - Memory Interface
00000011b - Memory Interface initialization
00000100b - ISO 8802.2 Link Encapsulation

00000110b - IPI-3 Slave
00000111b - IPI-3 Master
1xxxxxxxb - Locally assigned

P - D1_Data_Set_Present = 1 designates that a D1_Data_Set is present in this packet.

B - Start_D2_on_Burst_Boundary = 0 designates that the D2_Area starts at or before the beginning of the second burst. B=1 designates that the D2_Area starts at the beginning of the second burst.

D1_Area_Size (8 bits) designates the size of the D1_Area, the number of 64 bit Double Words that are between the Header and the start of the D2_Area.

D2_Offset (3 bits) designates the number of offset bytes from the start of the D2_Area to the first byte of the D2_Data_Set

Reserved (11 bits) must be 0.

D2_Size (32 bits) is the length in bytes of the D2_Data_Set portion of the packet. The D2_Size does not include the bytes contained in the D2_Offset or the fill following the D2_Data_Set. A D2_Size of 0xFFFFFFFF specifies that the D2_Data_set size is unknown. This means that packets with an unknown D2_DataSet size cannot be terminated at arbitrary byte boundaries, only at 64 bit HIPPI-PH burst boundaries. A D2_Size of zero specifies that the D2_Area and the D2_Data_Set do not exist.

D1_Area

The D1_Area shall immediately follow the Header_Area, shall be completely contained in the first burst, shall contain an integral number of 64 bit Double Words, and shall contain the D1_Data_Set (if present).

If Present, the D1_Data_Set shall be the first information in the D1_Area. If the P bit of the Header is 0, then the D1_Data_Set is not present. The D1_DataSet is intended for control information that may be delivered to the destination ULP on receipt, without waiting for the arrival of other bursts of the packet.

The size of the D1_Data_Set is self defining. The maximum size is 1016 bytes.

D2_Area

The D2_Area, if D2_Size is not zero, shall immediately follow the D1_Area, shall start and end on 64 bit Double Word boundaries, and shall contain the D2_Data_Set. If the B bit in the Header is 1, the D2_Area shall start at the beginning of the second HIPPI_PH burst.

The Offset is the unused bytes from the start of the D2_Area to the first byte of the D2_Data_Set.

Fill is the unused bytes between the end D2_DataSet and the end of the D2_Area. When D2_Size is all binary ones, then there is no fill.

Annex - C
HIPPI-LE ARP Messages

HIPPI-LE specifies four ARP packet types with the following Message_Type (M_Type - byte 0 of the LE header) values:

- 0001b Address Resolution Request PDU (AR_Request)
- 0010b Address Resolution Response PDU (AR_Response)
- 0011b Self Address Resolution Request PDU (AR_S_Request)
- 0100b Self Address Resolution Response PDU (AR_S_Response)

ARP packets as processed by the host system ARP stack are either AR_Request or AR_Response packets. The AR_Request packet is sent to Destination Logical Address 0xFE0.

An AR_Response packet that is a reply to a AR_Request is sent directly to the Source Logical Address found in the AR_Request packet.

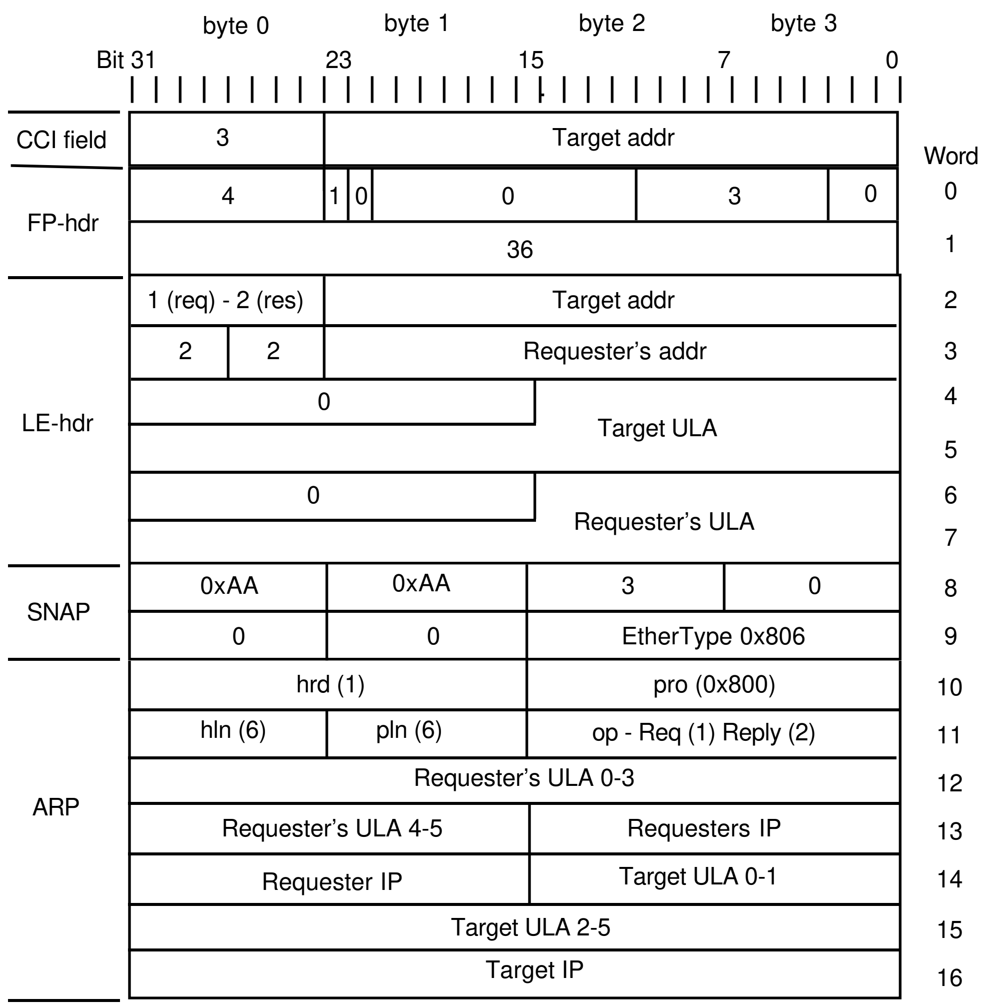

Figure C-1 ARP Request/Response Packet Format

AR-S_Response messages are used by address discovery and to notify the Agents that a host is up.

The typical ARP request and ARP response packet is found in Figure C-1.

The CCI field only appears on transmit packets. It is removed by the RunCode on receive packets. The CCI field target address is the same as the LE header Target Address.

The LE Target address when a host sends a Request is 0xFE0. When the ARP Agent sends a packet it contains the destination address of the packet. When a host sends a response packet the address is the address of the destination host.

When sending a packet to the ARP Agent, the Target ULA is FF:FF:FF:FF:FF:FF (the broadcast ULA). The Target and Requester's ULA are copied from fields in the ARP section.

The ARP section is a standard ARP datagram as processed by the host ARP stack.

The typical ARP_S_Request and ARP_S_Response packet is found in Figure C-2.

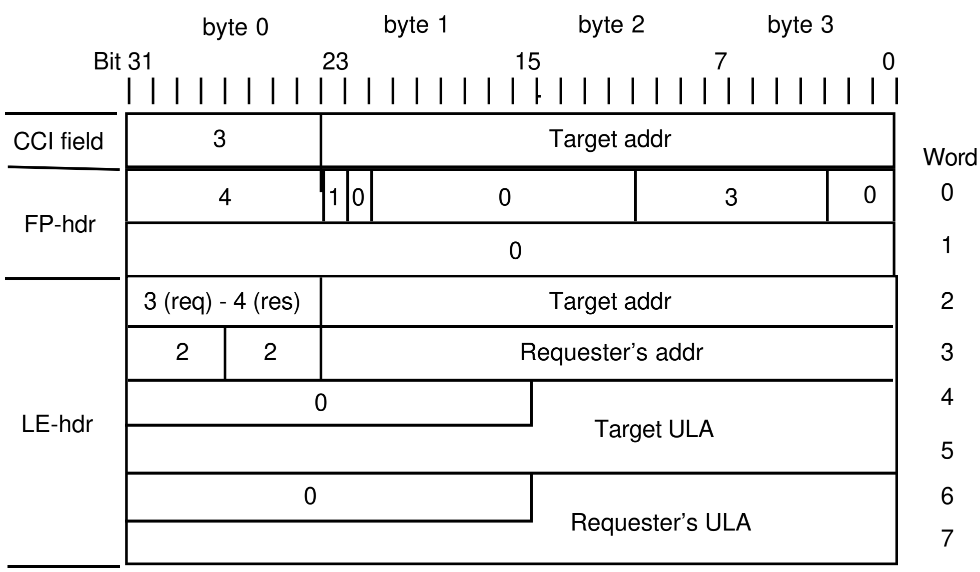

Figure C-2 ARP_S_ Request/ARP_S_Response Packet Format

The ARP_S_Request and ARP_S_Response packets are use in address discovery, address verification and to notify the IP Broadcast Agent that the host is available.

In all cases the CCI field Target address and the LE Target address are the same.

Discovery uses Response messages (M_Type 4). The Target address and Requester's address are the same. The Target ULA and Requester's ULA are the same. Verification uses a discovery packet where the addresses are the suspected logical address of the NIC.

The notification packet to the IP Broadcast Agent has a Target address of 0xFE1 and the Target ULA of FF:FF:FF:FF:FF:FF (the broadcast ULA). The Requester's address is the NIC address and the Requester's ULA is the NIC ULA.

Annex - D
IP over HIPPI

IP over HIPPI is done as described in RFC 1374. IP is in Big Endian form. Communication stacks in Host software traditionally byte swap as needed. An IP packet as it is passed through a HIPPI-SC compliant switch has the following format:

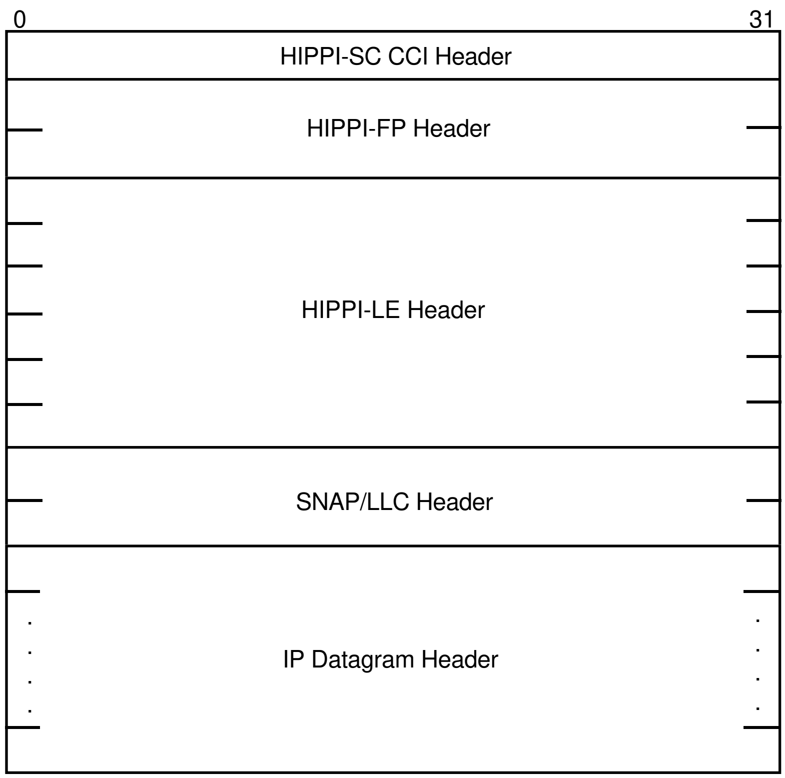

Figure D-1 IP over HIPPI Packet Format

The Ether Type (last 2 bytes) field in the SNAP header can be 0x0800 indicating an IP packet or 0x0806 indicating an ARP packet.

Annex - E
IP Datagram Checksum

The IP checksum is formed by treating the header as a sequence of 16-bit integers (in network byte order), adding them together using one's complement arithmetic, and then taking the one's complement of the result. For purposes of computing the checksum, field HEADER CHECKSUM is assumed to contain 0.

In the figures, the bits are in Big Endian order as defined in the RFCs.

It is important to note that the checksum applies only to values in the IP header and not to the data.

0	4	8	16	19	24	31
VERS	HLEN	SERVICE TYPE	TOTAL LENGTH			
IDENTIFICATION			FLAGS	FRAGMENT OFFSET		
TIME TO LIVE		PROTOCOL	HEADER CHECKSUM			
SOURCE IP ADDRESS						
DESTINATION IP ADDRESS						
IP OPTIONS (IF ANY)					PADDING	
DATA						
…						

Figure E-1 IP Datagram Format

HLEN is the length of the header in 32 bit words. HEADER CHECKSUM is calculated over HLEN times 4 bytes.

The UDP and TCP protocol stacks layer above IP (See figure E-2)

UDP	TCP
IP	

Figure E-2 Protocol Layering

The UDP datagram is appended to the IP datagram.

0	16 31
UDP SOURCE PORT	UDP DESTINATION PORT
UDP MESSAGE LENGTH	UDP CHECKSUM

Figure E-3 UDP Datagram Format

The UDP checksum covers more information than is present in the UDP header alone. UDP prepends a -header to the UDP datagram and appends an octet of zeros to pad the datagram into an exact multiple of 2 byte objects and computes the checksum on the result. The Octet used to pad the datagram is not transmitted and is not included in the length. The checksum field is assumed to be 0 during checksum computation.

0	8	16	31
SOURCE IP ADDRESS			
DESTINATION IP ADDRESS			
0	17	UDP LENGTH	

Figure E-4 UDP Pseudo-Header

The SOURCE IP ADDRESS and DESTINATION IP ADDRESS are from the IP datagram. The UDP LENGTH is the length of the UDP datagram excluding the pseudo-header and the pad.

A TCP segment is as follows:

0	4	10	16	24	31
SOURCE PORT			DESTINATION PORT		
SEQUENCE NUMBER					
ACKNOWLEDGEMENT NUMBER					
HLEN	RESERVED	CODE BITS	WINDOW		
CHECKSUM			URGENT POINTER		
OPTIONS (IF ANY)				PADDING	
DATA					
…					

Figure E-5 TCP Segment

HLEN contains the length of the segment header in 32 bit words.

The TCP checksum covers more information than is present in the TCP segment alone. TCP prepends a pseudo-header to the TCP segment and appends an octet of zeros to pad the datagram into an exact multiple of 2 byte objects and computes the checksum on the result. The Octet used to pad the datagram is not transmitted and is not included in the length. The checksum field is assumed to be 0 during checksum computation.

0	8	16	31
SOURCE IP ADDRESS			
DESTINATION IP ADDRESS			
0	6	TCP LENGTH	

Figure E-6 TCP Pseudo-Header

The SOURCE IP ADDRESS and DESTINATION IP ADDRESS are from the IP datagram. The TCP LENGTH is the length of the TCP segment excluding the pseudo-haeder and the pad.

Annex - F
HIPPI-PH Excerpts

The following Physical framing hierarchy describes the data flow over HIPPI media.

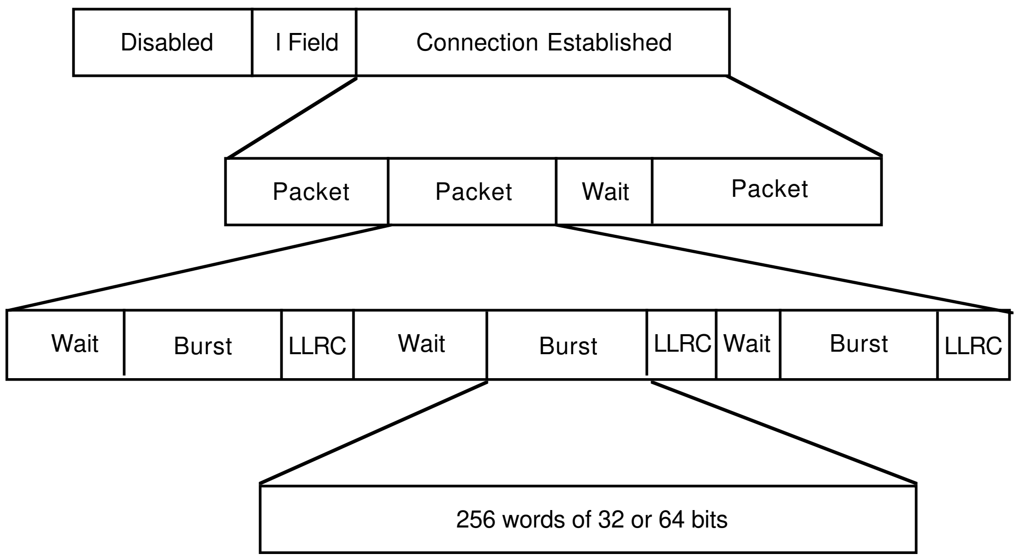

Figure F-1 Physical Framing Hierarchy

The Essential NIC implements 32 bit words.

References

The following documents are useful in understanding the context in which the NIC operates. It is always encouraged to use the latest revision of each document.

Road Runner PCI/HIPPI Controller Specification 0.9

PCI Local Bus Specification Revision 2.0

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

Serial-HIPPI Specification, Revision 1.0, Serial HIPPI Implementors Group, May 17,1991

ANSI X3.218-199x, High-Performance Parallel Interface - Framing Protocol (HIPPI-FP)

ANSI X3.218-199x, High-Performance Parallel Interface - Encapsulation of ISO 8802-2 (IEEE Std 802.2) Logical Link Control Protocol Data Units (HIPPI-LE)

ANSI X3.222-199x, High-Performance Parallel Interface - Physical Switch Control (HIPPI-SC)

IEEE Standard 802.1A

IP - RFC 791

UDP/IP - RFC 768

TCP/IP - RFC 793

Assigned numbers - RFC 1060

IP over HIPPI - RFC 1374

ARP over HIPPI - RFC xxx

SNMPv2 - RFC 1157

MIB-II - RFC 1213

Glossary

Big Endian: most significant bit of data of a multiple byte object is at the lowest byte address

Card: Essential Communications Corp. HIPPI/PCI Network Interface Card (NIC)

Checksum: The 8802.2 defined checksums for the IP header, TCP/IP and UDP/IP datagrams

Destination: HIPPI receiver that can accept Packets

DMA: Direct Memory Access. Data are moved directly between the NIC and Host memory without passing through the CPU

Double Word: 8 byte object

duplex: Device can both send and receive data

Endian: The end of a multiple byte object that has the low order byte address. See Big Endian and Little Endian.

End Point: A HIPPI Source or Destination

Firmware: Software that is running in the NIC

Full Duplex: Device can simultaneously send and receive data

HIPPI: High Performance Parallel Interface 800 or 1600 M bit/sec communication media

Half Word: 2 byte object

Host: System in which the NIC is operating

Host Software: Software that is executing on the Host system

K: 1024

KB: 1024 bytes

Little Endian: least significant bit of data of a multiple byte object is at the lowest byte address

Master mode DMA: DMA operations are initiated and controlled by the NIC

MB: 1024*1024 bytes

msec: millisecond, 1.0E-3 seconds

NIC: Network Interface Card

nsec: nanosecond, 1.0E-9 seconds

Packet: A Packet as defined in HIPPI-PH

PCI: Peripheral bus standard

Power PC: CPU based on IBM/Motorola/Apple design

Road Runner: ASIC that is the heart of NIC. It controls processing and the host interface

RunCode: Firmware used during normal operation of the NIC

sec: second

simplex: One way communication path. Device can either send or receive

Single Wide: HIPPI operating at 800 M bit/second

Source: HIPPI transmitter that can generate Packets

SNMPv2: Simple Network Management Protocol version 2.

usec: microsecond, 1.0E-6 seconds

Word: 4 byte object