Chapter 1

Some HIPPI Basics

This chapter is an introduction to the High-Speed Parallel Processing Interface (HIPPI) protocol and the IRIS HIPPI hardware. The chapter provides a brief introduction to HIPPI, a description of the HIPPI protocol, some common configurations of HIPPI equipment, how to obtain official HIPPI documentation, and a theory of operations section for the IRIS HIPPI hardware.

1.1 Introduction to the HIPPI Protocol

This section provides a brief introduction to HIPPI.

1.1.1 HIPPI Terminology

HIPPI uses *source* to refer to the transmitting host, endpoint, network interface, or program.

It uses *destination* to refer to the receiving host, endpoint, network interface, or program.

1.1.2 How HIPPI Works

HIPPI is an extremely fast, simplex point-to-point protocol. HIPPI provides for transmission at 800 or 1600 megabits per second*. Before data can be sent from one HIPPI network interface (endpoint) to another there must be both a physical link and an open connection between them. The physical link consists of up to 25 meters of copper cable that connects a sending endpoint with either an intermediate HIPPI switch or a receiving endpoint. The open connection is an agreement for data transfer between two endpoints and is arranged with an exchange of signals.

---

* IRIS HIPPI supports only 800 Mbits per second.

Figure 1-1 illustrates a configuration of HIPPI equipment with six physical links. This configuration supports nine different endpoint-to-endpoint communication paths (listed below) of which three can be simultaneously active:

- A-source transmitting to C-destination, B-destination, or A-destination
- B-source transmitting to C-destination, B-destination, or A-destination
- C-source transmitting to C-destination, B-destination, or A-destination

An open connection consists of an exchange of signals between a source and a destination. During this exchange, the destination agrees to accept data exclusively from the source. Each endpoint-to-endpoint link supports one connection (that is, HIPPI is point to point). It is common for an interface's source and destination channels to have open connections with different hosts (for example, A-source connected to B-destination while A-destination is connected to C-source). To move data in both directions between two hosts, two endpoint-to-endpoint links and two open connections are needed between the two hosts.

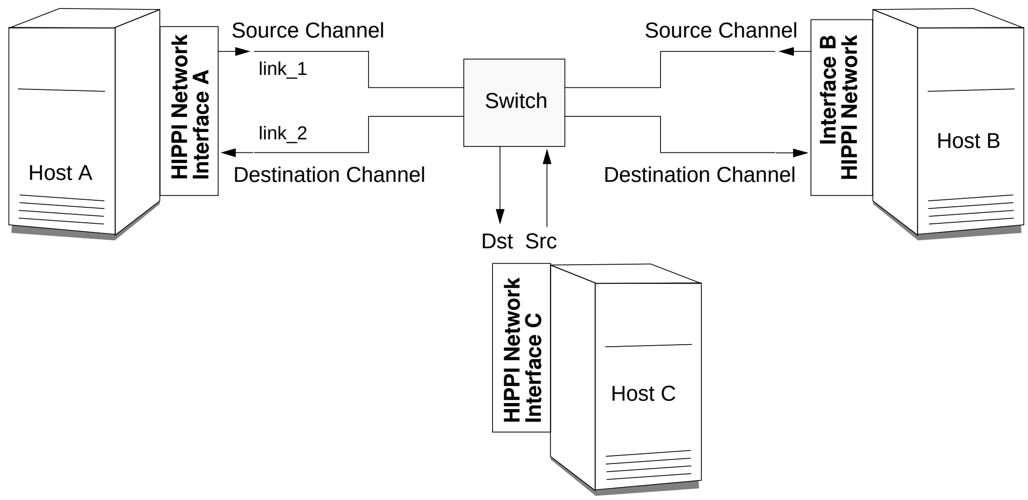

Figure 1-1 HIPPI Links and Connections

Unlike Ethernet, 802.5 Token Ring, or FDDI, HIPPI does not use a shared medium. Once a connection is established, the cable between the two HIPPI interfaces contains only packets transmitted by the source (that is, HIPPI connections are simplex). HIPPI packets may be seen by intermediate switches but not by other host interfaces. When one packet has been sent, the connection may be closed or kept open so that additional packets can be sent; however, each endpoint may not participate in another connection until the current one has been closed.

HIPPI communication is controlled by three basic functions: connection control, packet control, and routing control (pertinent only when one or more switches are involved). Each of these is discussed separately in the subsections that follow.

1.1.3 Connection Control

Before a source (transmitting) HIPPI network interface can send a packet, it must open a connection to one destination HIPPI interface. The source interface is always the initiator for opening the connection. To open a connection, the sender issues a connection request by asserting the `REQUEST` signal on the link. Each connection request includes an I-field (described in Section 1.2). The I-field contains the following two kinds of information:

- Connection setup information, used by the destination endpoint.
- Routing and control information, used by any switches encountered along the path to the destination.

The destination accepts a connection by asserting its `CONNECT` signal in response to the request. If the destination endpoint is unwilling to accept the connection or if there is a problem with the connection request (for example, bad parity on the I-field or incompatible word size), the connection request is denied (that is, acknowledged, then rejected). The transmitter must wait and try again later or forgo the communication. If the destination is unreachable (for example, a broken physical link, a powered-down or dysfunctional network interface), there is no response and the source program times out.

When a switch exists between the source and destination, the source receives its connection rejections from the switch, not directly from the destination. The rejection can be caused by any of the following conditions, and it is not possible to distinguish among them (except as explained below):

1. the destination is malfunctioning
2. the destination refuses to accept the requested connection
3. the connection request has an error
4. a section of the physical link to the destination is busy (currently engaged in another connection)

A feature is available that allows the source to be informed of rejections due to error conditions (items 1-3 above) but not to be informed (bothered) when the rejection is due to a busy link (item 4). This feature is called *camp-on*. By setting the camp-on bit in the I-field, the source can program the switches to hold onto the connection request until the busy link to the destination becomes available.

When the camp-on bit is set, any switch along the path to the destination that finds its link toward the destination busy, queues the connection request, and periodically checks to see if the link has become available. When the link becomes available, it sends the connection request. A switch will continue to wait until it sends the `REQUEST` to the ultimate destination endpoint or until the source aborts the connection request. If a number of sources are all trying to send data through the same link, the camp-on feature ensures fair (first come, first served) access to the link.

Once opened, a HIPPI connection may be kept open for as long as the two endpoints maintain it. Either endpoint may terminate the connection at any point in time; however, the source network interface is usually the initiator.

1.1.4 Packet Control (Data Flow)

Once a connection is open, one or multiple packets may be sent. The destination indicates it is ready to receive data by sending a READY signal to the source endpoint. Each READY allows the source to transmit one HIPPI *burst* (as explained below). All HIPPI source endpoints are required to enqueue a minimum of 63 READYs. There is no minimum requirement for a destination's ability to generate READYs. [*]. By sending ahead and queuing READYs, the two endpoints can optimize the throughput on their connection.

A HIPPI packet consists of one or more bursts, as illustrated in Figure 1-2. Each burst contains 256 words, except in the case where the burst is a *short* burst (as described below). The size of each word depends on the source's data bus (32 or 64 bits, as indicated by a bit in the I-field)[†]. At the end of each burst the source generates a checksum (LLRC) so that the destination can detect any errors in the received data; in addition, each word has four bits of parity for error checking.

The HIPPI protocol requires very small waiting periods between packets and between bursts. These required periods are counted in nanoseconds and are essentially imperceptible to the user; however, in normal operation there may be noticeable pauses between bursts (for example, when the source is waiting to receive a READY).

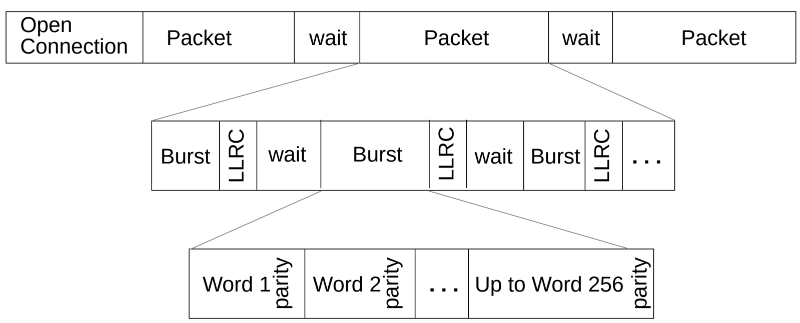

Figure 1-2 HIPPI Packets and Bursts

As long as the source has READYs, it can transmit data as fast as it is capable of transmitting (but no faster than the protocol allows: 25 million words per second). When the sender has sent all the data for one packet, it indicates the end of the packet. Indicating the end of the packet is necessary because HIPPI allows packet size to be undefined (indeterminate) at the start of the packet. A sender could essentially send an infinite-sized packet by keeping the packet signal asserted at all times.

A packet's first burst often contains some kind of header (for example, a HIPPI-FP header as described in Section 1.2). The first words of user data (for example, HIPPI-FP D2 data) can be part of the first burst or the second. If the source program is generating HIPPI-FP

---

* The source channel on the IRIS HIPPI board can enqueue up to 65,535 READYs; the destination channel can generate up to 255 outstanding READYs .

† The IRIS HIPPI board supports 32-bit words only.

PostScript File

packets, it indicates the location of the packet's first byte of user data by setting the B bit in the HIPPI-FP header.

Either the first or the last burst of a packet (but not both) can be less than 256 words. This burst is referred to as a *short* burst. Usually, the last burst is the short one. When the first burst is the short one, it contains only the header; the first byte of user data is located in the second burst, and the final burst may be padded to meet the 256-word length requirement. When the last burst is the short one, the packet's final burst never needs to be padded and the first word of user data is included within the first burst (immediately after the last word of the header).

Once the end of the packet has been indicated, the source has the option of keeping the connection open to transmit additional packets or of closing the connection.

1.1.5 Routing

The I-field contains HIPPI routing information in its 24-bit Routing Control field. This information is interpreted only by intermediate systems (switches); the Routing Control information is not interpreted when a connection is directly between two end systems.

The addresses in a Routing Control field can be in "logical addressing" or "source addressing" format. The format is indicated by the Path Selection bits of the I-field. The two formats cannot be used simultaneously in one I-field; however, both formats can be used simultaneously in one HIPPI local area network (LAN).

Note: The word *source* in "source addressing format" does not mean that the address is the source's address; it refers to the fact that the address supplied by the sending endpoint defines the complete path (route).

1.1.5.1 Logical Addressing

With logical addressing, the Routing Control field contains two 12-bit addresses: a destination (receiver's) address and a source (sender's) address, as illustrated in Figure 1-3. The order in which the addresses are placed within the field is defined by the Direction bit, as illustrated in Figure 1-3.

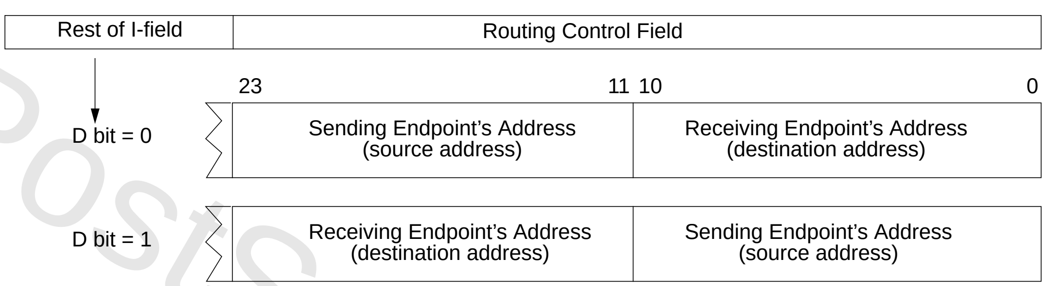

Figure 1-3 Routing Control Field with Logical Addressing