X3T11/92-
REV 8.2

HIGH-PERFORMANCE PARALLEL INTERFACE -

Mechanical, Electrical, and Signalling Protocol Specification

(HIPPI-PH)

maintenance copy of
American National Standard
for Information Systems
X3.183-1991

March 3, 1993

Secretariat:

Computer & Business Equipment Manufacturers Association

ABSTRACT: The described High-Performance Parallel Interface (HIPPI-PH) is intended as the physical layer of an efficient simplex high-performance point-to-point interface for transmitting digital data at peak data rates of 800 or 1600 Mbit/s between data-processing equipment using multiple twisted-pair copper cabling at distances up to
25 m.

NOTE:

This document is an X3T11 maintenance copy of American National Standard X3.183-1991. A list of the proposed changes since approval of the standard is included. Some or all of these changes may be included in a future erratum, amendment, interpretation, or revision to the standard. The committee lists included in the approved standard, and some other boilerplate, have been omitted in this draft. For current information on the status of this document contact the individual shown below.

POINTS OF CONTACT:

Roger Cummings (X3T11 Chairman)
Storage Technology Corporation
2270 South 88th Street
Louisville, CO 80028-0268
(303) 661-6357, FAX (303) 684-8196
E-mail: Roger_Cummings@Stortek.com

Carl Zeitler (X3T11 Vice-Chairman)
IBM Corporation, MS 9440
11400 Burnet Road
Austin, TX 78758
(512) 838-1797, FAX (512) 838-3822
E-mail: zeitler@ausvm6.vnet.ibm.com

Don Tolmie (HIPPI-PH Technical Editor)
Los Alamos National Laboratory
C-5, MS-B255
Los Alamos, NM 87545

(505) 667-5502, FAX (505) 665-7793
E-mail: det@lanl.gov

Proposed changes to ANSI X3.183-1991 – (and which have been incorporated in this draft)

Proposed technical changes are preceded with (T), proposed editorial changes are preceded with (E).

Rev 8.2, March 3, 1993 version - To date all of the proposed changes are editorial (E).

1. (E) Change 2.1, line 1, from "For the purposes of..." to "For the purpose of...".
2. (E) Change 2.1.7, last line, from "...or by a upper..." to "...or by an upper...".
3. (E) Change 2.2, last line, change "...English meanings." to "...English meaning.".
4. (E) Deleted an extra space after the period in some of the primitive names, e.g., "PH_HANGUP. Request". This affected line 5 of 4.5.1, line 12 of 4.8.2, line 1 of 4.8.4, line 1 of 4.9.2, line 1 of 4.10.2, line 3 of 4.10.3, line 5 of 4.10.4, and last line of 4.12.1.
5. (E) In 4.6.2, line 5/6, moved the period after "ANSWER" to before "Request".
6. (E) In 4.12.3, lines 6/7, moved "A change in the INTERCONNECT signal" to the end of the previous line instead of indenting it.
7. (E) In 5.1.5, added a period after the sentence "All addition is modulo 2", in two places.
8. (E) Capitalized the words "source" and "destination in figures 17 and 18, S1170, S1230, S1260, D1670, D1770, D1630, and D1750.
9. (E) In 8.1.3, 8.2.2, and 8.5 added a space between the number and the Ω.
10. (E) In 8.2.1, line 2, change "...voltage shall be meet..." to "...voltage shall meet...".
11. (E) In B.6, paragraph 2, change "Figure B.4" to "Figure B.3".
12. (E) In B.6, paragraph 3, change "figure B.5" to "figure B.4".
13. (E) In E.2, add a space between the numeric value and the units, (m, cm, ft, in).
14. (E) In the Index, change "*int*" to "Introduction" Introduction" in four places. Delete "*(int = Introduction)*".

Contents

Tables

Figures

Annexes

Foreword (This Foreword is not part of American National Standard X3.183-1991.)

This standard defines the mechanical, electrical, and signalling protocol specifications of an efficient simplex high-performance point-to-point interface, called the High-Performance Parallel Interface (HIPPI-PH). The -PH abbreviation stands for "physical" layer. This interface was previously named the "High-Speed Channel (HSC)". The name was changed October, 1989, to avoid infringing on an existing trademark, and the abbreviation was changed April, 1990, from HPPI to HIPPI to avoid another trademark.

The HIPPI-PH is designed for transmitting digital data at peak data rates of 800 or 1600 Mbit/s between data-processing equipment using multiple twisted-pair copper cabling at distances up to 25 meters. This standard responds to an industry market need (expressed both by users and manufacturers) to standardize the interconnection of data processing equipments at these data rates.

The HIPPI-PH signalling protocol is designed to be distance independent, allowing the average data rate to approach the peak data rate, even over distances longer than specified for the HIPPI-PH.

This standard was developed by Task Group X3T9.3 of Accredited Standards Committee X3 during 1987 and 1988. The standards approval process started in early 1989.

The HIPPI Framing Protocol (HIPPI-FP) is an upper-layer protocol for the HIPPI-PH.

The American convention of numbering is used i.e., the period is used as the decimal point, and four digit numbers are contiguous. This is equivalent to the ISO convention of a comma for the decimal point, and a space separating thousands and higher multiples.

ISO	American
0,6	0.6
1 600	1600
1 323 462,9	1,323,462.9

This document includes annexes which are informative and are not considered part of the standard.

Requests for interpretation, suggestions for improvement or addenda, or defect reports are welcome. They should be sent to the X3 Secretariat, Computer and Business Equipment Manufacturers Association, 1250 Eye Street, NW, Suite 200, Washington, DC 20005.

This standard was processed and approved for submittal to ANSI by Accredited Standards Committee on Information Processing Systems, X3. Committee approval of the standard does not necessarily imply that all committee members voted for approval. At the time it approved this standard, the X3 Committee had the following members:

(List of X3 Committee members included in published standard)

Subcommittee X3T9 on Computer Input/Output Interfaces, which reviewed this standard, had the following members:

(List of X3T9 committee members included in published standard)

Task Group X3T9.3 on Device Level Interfaces, which developed this standard, had the following participants:

(Lis tof X3T9.3 committee participants included in published standard)

Introduction

This High-Performance Parallel Interface, Mechanical, Electrical, and Signalling Protocol Specification (HIPPI-PH) American National Standard defines the physical layer of an efficient simplex high-performance point-to-point interface operating at speeds of 800 or 1600 Mbit/s. The -PH abbreviation stands for "physical layer".

Characteristics of this HIPPI physical layer interface include:

– Point-to-point connections use one or two copper twisted-pair cables for distances of up to 25 m.

– The HIPPI-PH is a simplex interface, capable of transferring data in one direction only. Two HIPPI-PHs may be used to implement a full-duplex interface.

– Data transfers are performed and flow controlled in increments of bursts, each burst normally containing 256 words.

– Signalling and control sequences are kept simple, and a look-ahead flow control is used, to allow average transfer rates for large file transfers to approach the peak transfer rate, even over distances longer than specified for the HIPPI-PH cables.

– The HIPPI-PH provides support for low-latency, real-time, and variable size packet transfers.

– The HIPPI-PH is designed to facilitate use in a circuit-switched environment. In support of this feature, a limited information field is available for subdevice addressing or other nonspecified control functions during the connection phase of operation. One round-trip cable delay is required to establish or terminate a connection.

– The HIPPI-PH is also designed to transmit multiple packets after a connection has been established. No round-trip cable delays are required between packets.

The information in the document is organized as follows:

– Clause 1 provides the introductory material.

– Clause 2 provides a glossary and editorial conventions.

– Clause 3 gives an overview of the HIPPI structure and logical organization.

– Clause 4 specifies the services provided by the HIPPI-PH to station management and upper-layer protocols.

– Clause 5 specifies the interface format and the definitions of the signals used in the interface.

– Clause 6 specifies control sequences, using pseudo-code and flow diagrams, required to implement the information transfers.

– Clause 7 provides detailed timing information.

– Clause 8 specifies the electrical and mechanical connections defined by this standard. These specifications cover drivers, receivers, line termination, logic levels, connectors, and cable parameters.

Figure 1 shows the interrelationship of the different clauses of the document. The upper-layer protocols and station management protocols are not covered in this standard.

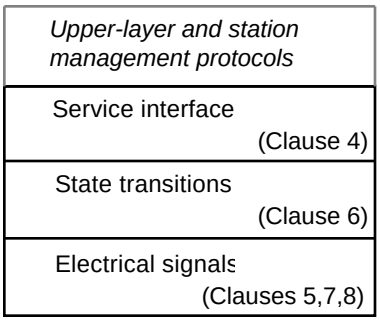


Figure 1 – Control hierarchy

American National Standard
for Information Systems –

High-Performance Parallel Interface – Mechanical, Electrical, and Signalling Protocol Specification (HIPPI-PH)

1 Scope

This American National Standard provides the mechanical, electrical, and signalling protocol specifications for an efficient simplex high-performance point-to-point interface between pieces of data-processing equipment.

The interface described in this document can be operated at peak data rates of 800 or 1600 Mbit/s, over distances of up to 25 m by means of copper cabling. A distance-independent signalling protocol allows the average data rates to approach the peak data rates, even over distances longer than specified for the HIPPI-PH.

The purpose of this American National Standard is to facilitate the development and use of computer systems by providing a common interface at the physical and data framing layers. HIPPI-PH provides an efficient interconnection between computers, high-performance display systems, and high-performance, intelligent block-transfer peripherals. It is optimized for large block transfers.

2 Definitions and conventions

2.1 Definitions

For the purpose of this standard, the following definitions apply.

2.1.1 burst: A group of words, sent during contiguous CLOCK periods. A burst may be sent by the Source for each READY indication received from the Destination. Bursts contain 1 to 256 words. Bursts that contain less than 256 words are called short bursts. A packet contains no more than one short burst. A short burst will be either the first or last burst of a packet.

2.1.2 connection: Condition of the HIPPI-PH when data transfers from Source to Destination are possible.

2.1.3 Destination: The equipment that receives the data.

2.1.4 I-Field: A 32-bit information field sent as part of the sequence of operations establishing a connection from a Source to a Destination. The contents of the I-Field are defined by an upper-layer protocol and are not defined in this standard.

2.1.5 length/longitudinal redundancy check-word (LLRC): A single word that is sent on the DATA BUS from Source to Destination after each burst.

2.1.6 optional: Features that are not required by the standard. However, if any optional feature defined by the standard is implemented, it shall be implemented according to the standard.

2.1.7 packet: A data set sent from Source to Destination. A packet is composed of one or more bursts. The HIPPI-PH specification does not limit the maximum packet size, but a maximum size may be imposed by a given HIPPI-PH implementation, or by an upper-layer protocol.

2.1.8 service interface (SI): The means by which the HIPPI-PH provides services to upper-layer protocols.

2.1.9 Source: The equipment that transmits the data.

2.1.10 state: The current condition of the HIPPI-PH, excluding transitions, as indicated by the control signals.

2.1.11 station management (SMT): The supervisory entity that monitors and controls the HIPPI-PH.

2.1.12 upper-layer protocol (ULP): The protocols above the service interface. These could be done in hardware or software, or they could be distributed between the two.

2.1.13 wait: Wait is every CLOCK period when there is no valid information on the DATA BUS. Some wait times are required by the HIPPI-PH signalling protocol; others may be the result of flow control or upper-layer protocol operations.

2.1.14 word: A unit of information, consisting of 32 or 64 bits, matching the DATA BUS width, and transferred from the Source to the Destination during a CLOCK period.

2.2 Editorial conventions

In this standard, certain terms that are proper names of signals or similar terms are printed in upper case to avoid possible confusion with other uses of the same words (e.g., REQUEST, CONNECT, BURST). Any lower case 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 upper case and the rest lower case (e.g., Source, Destination). Any lower case uses of these words have the normal technical English meaning.

3 HIPPI structure

3.1 Configuration characteristics

The HIPPI-PH has been designed in a modular fashion to support different peak bandwidth requirements.

3.1.1 800 Mbit/s

An HIPPI-PH with a DATA BUS width of 32-bit words provides 800 Mbit/s data transfer rates.

3.1.2 1600 Mbit/s

An HIPPI-PH with a DATA BUS width of 64-bit words provides 1600 Mbit/s data transfer rates.

3.2 Logical framing hierarchy

Figure 2 shows the basic organization of the information on the HIPPI-PH.

Once a connection is established a packet (or multiple packets) can be sent from the Source to the Destination. Each packet shall contain one or more bursts. Bursts shall contain 1 to 256 words. Bursts that contain less than 256 words are called short bursts. A packet shall contain no more than one short burst. A short burst shall be either the first or last burst of a multi-burst packet.

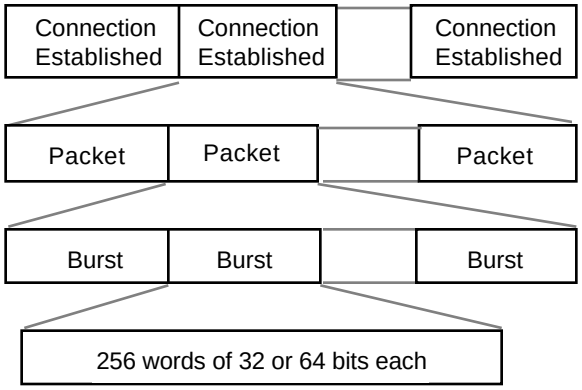


Figure 2 – Logical framing hierarchy

4 Service interface

This clause specifies the services provided by HIPPI-PH. The intent is to allow ULPs to operate correctly with this HIPPI-PH. How many of the services described herein are chosen for a given implementation is up to that implementor; however, a set of HIPPI-PH services must be supplied sufficient to satisfy the ULP(s) being used. The services as defined herein do not imply any particular implementation, or any interface.

Figure 3 shows the relationship of the HIPPI-PH interfaces.

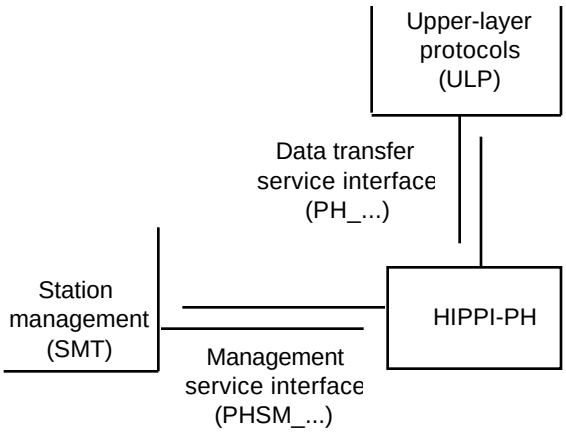


Figure 3 – HIPPI-PH service interface

4.1 Service primitives

The primitives, in the context of the state transitions in clause 5, are declared required or optional. Additionally, parameters are either required, conditional, or optional. All of the primitives and parameters are considered as required except where explicitly stated otherwise.

HIPPI-PH service primitives are of four types.

– *Request primitives* are issued by a service user to initiate a service provided by the HIPPI-PH. In this standard, a second Request primitive of the same name shall not be issued until the Confirm for the first request is received.

–*Confirm primitives* are issued by the HIPPI-PH to acknowledge a Request.

– *Indicate primitives* are issued by the HIPPI-PH to notify the service user of a local event. This primitive is similar in nature to an unsolicited interrupt. Note that the local event may have been caused by a service Request. In this standard, a second Indicate primitive of the same name shall not be issued until the Response for the first Indicate is received.

– *Response primitives* are issued by a service user to acknowledge an Indicate.

4.2 Sequences of primitives

The order of execution of service primitives is not arbitrary. Logical and time sequence relationships exist for all described service primitives. Time sequence diagrams are used to illustrate a valid sequence. Other valid sequences may exist. The sequence of events between peer users across the user/provider interface is illustrated. In the time sequence diagrams the HIPPI-PH users are depicted on either side of the vertical bars while the HIPPI-PH acts as the service provider.

4.3 Service primitives summary

Initiate a Connection
- PH_RING.Request (CCI)
- PH_RING.Confirm
- PH_RING.Indicate (CCI)
- PH_RING.Response

Complete the Connection
- PH_ANSWER.Request (Accept/Reject)
- PH_ANSWER.Confirm
- PH_ANSWER.Indicate (Accept/Reject)
- PH_ANSWER.Response

Flow Control
- PH_FLOW.Request
- PH_FLOW.Confirm
- PH_FLOW.Indicate (Enabled)
- PH_FLOW.Response

Packet Control
- PH_PACKET.Request (Begin/End)
- PH_PACKET.Confirm (Accept/Reject)
- PH_PACKET.Indicate (Begin/End,Status)
- PH_PACKET.Response

Burst Transfer
- PH_TRANSFER.Request (Length,Burst)
- PH_TRANSFER.Confirm (Accept/Reject)
- PH_TRANSFER.Indicate (Status,Length,Burst)
- PH_TRANSFER.Response

Terminate the Connection
- PH_HANGUP.Request
- PH_HANGUP.Confirm
- PH_HANGUP.Indicate
- PH_HANGUP.Response

Control Interface
- PHSM_CONTROL.Request (Parameter_List)
- PHSM_CONTROL.Confirm (Status,Status_List)

Interface Status
- PHSM_STATUS.Request
- PHSM_STATUS.Confirm (Status)
- PHSM_STATUS.Indicate
- PHSM_STATUS.Response

4.4 Operational sequences

Primitives issued by the ULP shall be serviced by the HIPPI-PH in the sequences defined in the state transitions of clause 6. An implementation may present the HIPPI-PH with multiple requests for services, but the HIPPI-PH shall service the requests one at a time.

The following sequence of service primitives is an example of normal operation of the HIPPI-PH.

4.4.1 Enable the interface

The interface may be enabled using whatever implementation dependent enabling method, if any, is specified.

4.4.2 Initiate a connection

PH_RING primitives shall be used to initiate a connection from the Source to the Destination. The Connection Control Information (CCI) may be used for non-specified control functions.

4.4.3 Complete the connection

PH_ANSWER primitives shall be used to establish or reject the connection.

4.4.4 Enable Destination reception

When it is ready, the Destination ULP may use the PH_FLOW.Request primitive to indicate that it is willing to accept bursts from the HIPPI-PH.

4.4.5 Start a packet

PH_PACKET (Begin) primitives shall be used to indicate the start of a packet.

4.4.6 Send burst

PH_TRANSFER primitives shall be used to transfer one burst of a packet.

4.4.7 Send more bursts

More bursts may be sent for this packet by returning to 4.4.6.

4.4.8 Terminate the packet

PH_PACKET (End) primitives shall be used to indicate the end of a packet.

4.4.9 Send more packets

More packets may be sent by returning to 4.4.5.

4.4.10 Terminate the connection

The Source or Destination may terminate the connection by using the PH_HANGUP primitives. Note that dedicated point-to-point HIPPI-PHs may never need to terminate the connection once it is established, or only terminate the connection as an error recovery process. See the cautionary note in 5.3.6.

4.4.11 Initiate another connection

Return to 4.4.2 to initiate another connection.

4.5 Initiate connection service primitives

Figure 4 is a diagram of these primitives. They shall be used to request that a connection be established between the Source and Destination.

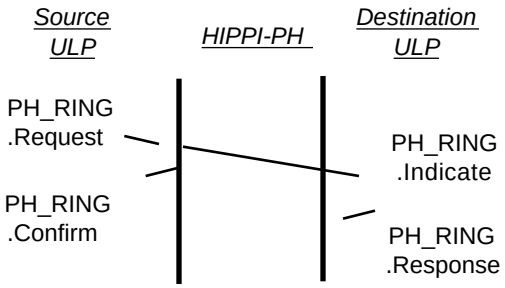


Figure 4 – Initiate the connection service primitives

4.5.1 PH_RING.Request

Issued by the Source ULP to request establishment of a connection from the Source to the Destination. In the case where a previous connection attempt did not complete with a PH_ANSWER.Indicate, and was aborted with a Source ULP issued PH_HANGUP.Request, the Source must wait a time T1 before initiating another connection. T1 shall be a round-trip propagation delay plus appropriate action time. No assumptions are made as to where or how the timer is implemented. It may be done in the ULP or the HIPPI-PH. The timer may be implemented with hardware, software, or a mixture of the two. The default value of T1 shall be approximately 2 ms.

Semantics – PH_RING.Request (CCI)

The CCI parameter is a 32-bit field that may be used for non-specified control operations for the connection establishment.

Issued – The Source ULP issues this primitive to the HIPPI-PH when a connection to a Destination ULP is required.

Effect – The HIPPI-PH shall initiate a connection.

4.5.2 PH_RING.Confirm

This primitive acknowledges the PH_RING.Request from the Source ULP.

Semantics – PH_RING.Confirm

Issued – The HIPPI-PH shall issue this primitive to the Source ULP to acknowledge a PH_RING.Request.

Effect – Unspecified

4.5.3 PH_RING.Indicate

This primitive indicates to the Destination ULP an attempt by a Source ULP to establish a connection.

Semantics – PH_RING.Indicate (CCI)

The CCI parameter is a 32-bit field that may be used for non-specified control operations for connection establishment. This CCI may be different from the CCI supplied by the PH_RING.Request primitive due to the action of intermediate devices, such as switches, between the Source and Destination.

Issued – The HIPPI-PH shall issue this primitive to the Destination ULP when it has received a connection request.

Effect – The Destination ULP should accept or reject the connection request.

4.5.4 PH_RING.Response

This primitive acknowledges the PH_RING.Indicate from the HIPPI-PH.

Semantics – PH_RING.Response

Issued – The Destination ULP issues this primitive to acknowledge receipt of the PH_RING.Indicate.

Effect – The HIPPI-PH is enabled to issue another PH_RING.Indicate.

4.6 Complete the connection service primitives

Figure 5 is a diagram of these primitives. To respond to a connection request, the PH_ANSWER primitives shall be used.

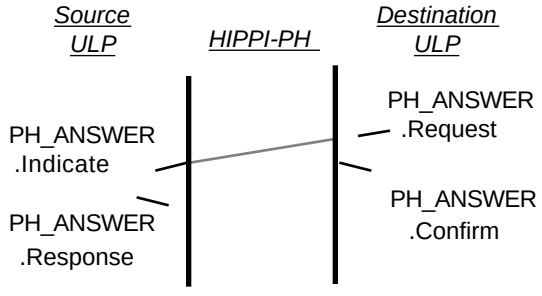


Figure 5 – Complete the connection service primitives

4.6.1 PH_ANSWER.Request

This primitive is issued by the Destination ULP in response to the PH_RING primitives to instruct the HIPPI-PH to reject or establish a connection.

Semantics – PH_ANSWER.Request (Accept/Reject)

The Accept/Reject parameter instructs the HIPPI-PH to complete the connection or to reject it.

Issued – The Destination ULP should issue this primitive to the HIPPI-PH in response to a PH_RING.Indicate.

Effect – The HIPPI-PH shall complete the connection if the Accept parameter is used. If the Reject parameter is used, the HIPPI-PH may either do nothing (in which case the Source ULP shall not receive a PH_ANSWER .Indicate), or perform a short connection sequence (in which case the Source ULP shall receive a PH_ANSWER .Indicate with the Reject parameter).

4.6.2 PH_ANSWER.Confirm

This primitive acknowledges the PH_ANSWER.Request from the Destination ULP.

Semantics – PH_ANSWER.Confirm

Issued – The HIPPI-PH shall issue this primitive to the Destination ULP to acknowledge a PH_ANSWER .Request.

Effect – Unspecified

4.6.3 PH_ANSWER.Indicate

This primitive indicates to the Source ULP that the connection has been accepted or rejected.

Semantics – PH_ANSWER.Indicate (Accept/Reject)

If the Accept parameter is used, the connection has been accepted and is available for data transfer. If the Reject parameter is used, the connection has been rejected and the interface is available for another PH_RING.Request.

Issued – The HIPPI-PH shall issue this primitive to the Source ULP when it has determined the status of the connection request.

Effect – The connection sequence is complete.

4.6.4 PH_ANSWER.Response

This primitive acknowledges the PH_ANSWER.Indicate from the HIPPI-PH.

Semantics – PH_ANSWER.Response

Issued – The Source ULP issues this primitive to acknowledge receipt of the PH_ANSWER.Indicate.

Effect – The HIPPI-PH is enabled to issue another PH_ANSWER.Indicate.

4.7 Flow control service primitives (optional)

Figure 6 is a diagram of these primitives. These primitives are optional, and may be used to pass flow control information between the HIPPI-PH and the Source and Destination ULPs.

The Flow Control service primitives are used by the Destination ULP to inform the HIPPI-PH that it is willing to accept a burst in the form of the PH_TRANSFER.Indicate.

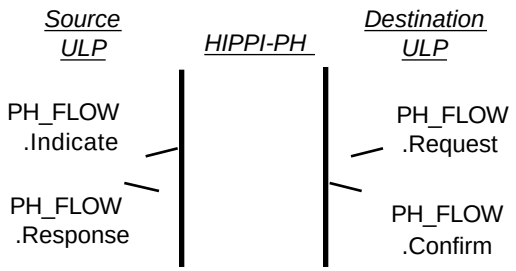


Figure 6 – Flow control service primitives

4.7.1 PH_FLOW.Request

This primitive is issued by the Destination ULP to tell the HIPPI-PH that it is ready to accept a burst from the HIPPI-PH.

Semantics – PH_FLOW.Request

Issued – The Destination ULP issues this primitive to the HIPPI-PH when it is ready to receive bursts.

Effect – The HIPPI-PH shall be enabled to send another burst to the Destination ULP. Each PH_FLOW.Request enables an additional PH_TRANSFER.Indicate.

4.7.2 PH_FLOW.Confirm

This primitive acknowledges the PH_FLOW.Request from the Destination ULP.

Semantics – PH_FLOW.Confirm

Issued – The HIPPI-PH shall issue this primitive to the Destination ULP to acknowledge a PH_FLOW.Request.

Effect – Unspecified

4.7.3 PH_FLOW.Indicate

This primitive notifies the Source ULP that it can send a burst. It shall indicate to the Source ULP the current number of available bursts that the HIPPI-PH is willing to accept.

Semantics – PH_FLOW.Indicate (Enabled)

The Enabled parameter shall be the current number of bursts that the HIPPI-PH is willing to accept. The range is implementation dependent.

Issued – The HIPPI-PH shall attempt to issue this primitive to the Source ULP every time it receives an indication that another burst can be transmitted.

Effect – Unspecified

4.7.4 PH_FLOW.Response

This primitive acknowledges the PH_FLOW.Indicate from the HIPPI-PH.

Semantics – PH_FLOW.Response

Issued – The Source ULP issues this primitive to acknowledge receipt of the PH_FLOW.Indicate.

Effect – The HIPPI-PH is enabled to issue another PH_FLOW.Indicate.

4.8 Packet service primitives

Figure 7 is a diagram of these primitives. They shall be used to delimit one or more bursts into entities called packets.

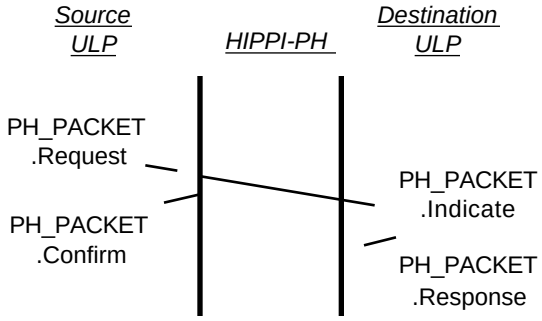


Figure 7 – Packet service primitives

4.8.1 PH_PACKET.Request

This primitive is issued by the Source ULP to delimit one or more bursts into a packet entity.

Semantics – PH_PACKET.Request (Begin/End)

The parameter marks either the beginning or the end of the packet.

Issued – The Source ULP issues this primitive to the HIPPI-PH to delimit one or more bursts into a packet.

Effect – Upon receipt of a Begin parameter, the HIPPI-PH shall mark the start of a packet. Upon receipt of an End parameter, the HIPPI-PH shall mark the end of a packet.

4.8.2 PH_PACKET.Confirm

This primitive acknowledges the PH_PACKET.Request from the Source ULP.

Semantics – PH_PACKET.Confirm (Accept/Reject)

The parameter denotes whether or not the primitive was accepted by the HIPPI-PH. The PH_PACKET.Request primitive may have been issued by the ULP along with other primitives, e.g., RING.Request. If the connection sequence fails, then the packet delimiter would not be inserted and this PH_PACKET.Confirm primitive would indicate Reject.

Issued – The HIPPI-PH shall issue this primitive to the Source ULP to acknowledge a PH_PACKET.Request.

Effect – Unspecified

4.8.3 PH_PACKET.Indicate

This primitive indicates to the Destination ULP that a packet delimiter has been received from the Source ULP.

Semantics – PH_PACKET.Indicate (Begin/End, Status)

The Begin/End parameter indicates either the beginning or the end of the packet.

The Status parameter may include, but is not limited to, an error when a burst with an illegal size and correct LLRC was received as part of this packet.

Issued – The HIPPI-PH shall issue this primitive to the Destination ULP when it receives a packet boundary indication.

Effect – Unspecified

4.8.4 PH_PACKET.Response

This primitive acknowledges the PH_PACKET.Indicate from the HIPPI-PH.

Semantics – PH_PACKET.Response

Issued – The Destination ULP issues this primitive to acknowledge receipt of the PH_PACKET.Indicate.

Effect – The HIPPI-PH is enabled to issue another PH_PACKET.Indicate.

4.9 Data transfer service primitives

Figure 8 is a diagram of these primitives. They shall be used to transfer a burst from the Source ULP to the Destination ULP.

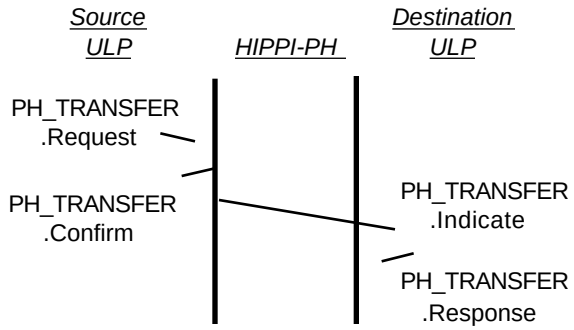


Figure 8 – Data transfer service primitives

4.9.1 PH_TRANSFER.Request

This primitive is issued by the Source ULP to request the transfer of a burst.

Semantics – PH_TRANSFER.Request (Length,Burst)

The length parameter may be specified in any units, but an integral number of words shall be transferred.

Issued – The Source ULP issues this primitive to the HIPPI-PH to request the transfer of a burst to the Destination ULP.

Effect – The HIPPI-PH shall accept the burst for transmission.

4.9.2 PH_TRANSFER.Confirm

This primitive acknowledges the PH_TRANSFER.Request from the Source ULP.

Semantics – PH_TRANSFER.Confirm (Accept/Reject)

The parameter denotes whether or not the primitive was accepted by the HIPPI-PH.

Issued – The HIPPI-PH shall issue this primitive to the Source ULP to acknowledge a PH_TRANSFER.Request.

Effect – Unspecified

4.9.3 PH_TRANSFER.Indicate

This primitive indicates to the Destination ULP that a burst has been received from the Source ULP.

Semantics – PH_TRANSFER.Indicate (Status,Length, Burst)

The Status parameter reports any parity or LLRC errors detected during the data transfer if error checking is supported. Illegal length bursts, with correct LLRC, shall be indicated in the PH_PACKET.Indicate primitive if error checking is supported.

Issued – The HIPPI-PH shall issue this primitive to the Destination ULP when a burst has been received.

Effect – Unspecified

4.9.4 PH_TRANSFER.Response

This primitive acknowledges the PH_TRANSFER.Indicate from the HIPPI-PH.

Semantics – PH_TRANSFER.Response

Issued – The Destination ULP issues this primitive to acknowledge receipt of the PH_TRANSFER.Indicate.

Effect – The HIPPI-PH is enabled to issue another PH_TRANSFER.Indicate.

4.10 Hangup service primitives

Figure 9 is a diagram of these primitives. They shall be used to terminate a connection between the Source and Destination. Note that a Hangup can be initiated from either the Source or Destination and will usually affect both the local and opposite ends of the interface.

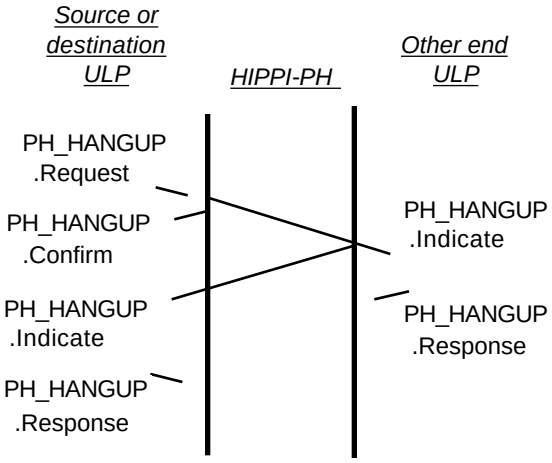


Figure 9 – Hangup service primitives

4.10.1 PH_HANGUP.Request

This primitive is issued by either the Source ULP or Destination ULP to request that the connection be terminated.

Semantics – PH_HANGUP.Request

Issued – The Source ULP or Destination ULP issues this primitive to the HIPPI-PH to terminate a connection.

Effect – The HIPPI-PH shall terminate the connection.

4.10.2 PH_HANGUP.Confirm

This primitive acknowledges the PH_HANGUP.Request to the issuing ULP.

Semantics – PH_HANGUP.Confirm

Issued – The HIPPI-PH shall issue this primitive to the ULP to acknowledge a PH_HANGUP.Request.

Effect – Unspecified

4.10.3 PH_HANGUP.Indicate

This primitive indicates to the ULP that the connection has been terminated. If a connection exists, the ULP that issued the PH_HANGUP.Request shall receive a PH_HANGUP.Confirm when the hangup sequence starts and a PH_HANGUP.Indicate when the sequence completes. The other end ULP shall receive only a PH_HANGUP.Indicate.

Semantics – PH_HANGUP.Indicate

Issued – The HIPPI-PH shall issue this primitive to the ULP when the connection has been terminated.

Effect – The ULP may no longer use the connection for data transfers, and all pending .Requests are cancelled.

4.10.4 PH_HANGUP.Response

This primitive acknowledges the PH_HANGUP.Indicate from the HIPPI-PH.

Semantics – PH_HANGUP.Response

Issued – The ULP issues this primitive to acknowledge receipt of the PH_HANGUP.Indicate.

Effect – The HIPPI-PH is enabled to issue another PH_HANGUP.Indicate.

4.11 Control service primitives

Figure 10 shows the SMT Control service primitives. They shall be used to set parameters and control the interface. Note that a Control primitive can be initiated from either the Source or Destination.

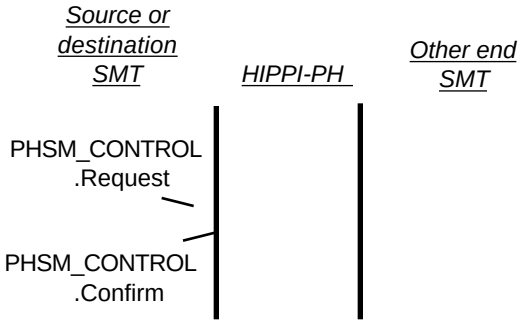


Figure 10 – Control service primitives

4.11.1 PHSM_CONTROL.Request

This primitive is issued by either the Source SMT or Destination SMT to set parameters within the interface, start diagnostics, or otherwise control the interface. One function is specified and others are left to specific implementations.

Semantics – PHSM_CONTROL.Request (Parameter_List)

The Parameter_List specifies what is to be set, started, etc. The Parameter list includes but is not limited to:

Reset

Issued – The Source or Destination SMT issues this primitive to perform some control function over the interface as a whole.

Effect – The HIPPI-PH shall perform the function specified. The Reset parameter shall take the HIPPI-PH to the Disabled state and cancel all pending .Requests.

4.11.2 PHSM_CONTROL.Confirm

This primitive acknowledges the PHSM_CONTROL.Request to the issuing SMT.

Semantics – PHSM_CONTROL.Confirm (Status, Status_ List)

Status reports the success or failure of the PHSM_CONTROL.Request operation. There should be one status value in Status_List for each event initiated with the PHSM_CONTROL.Request.

Issued – The HIPPI-PH shall issue this primitive to the SMT when all of the operations specified in the PHSM_CONTROL.Request have been accepted or completed.

Effect – Unspecified

4.12 Status service primitives

Figure 11 shows the SMT Status service primitives. They shall be used to obtain local status information from the HIPPI-PH. Note that a Status primitive can be initiated from either the Source or Destination and shall only affect the local end of the interface.

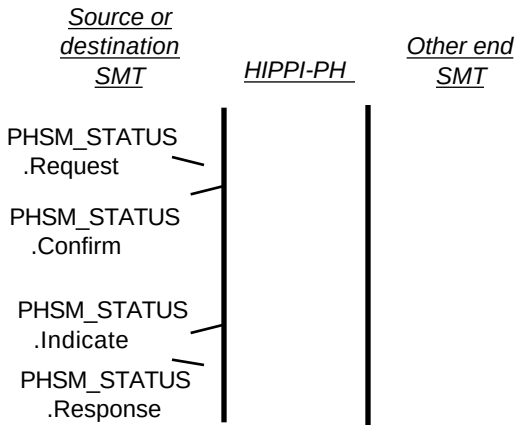


Figure 11 – Status service primitives

4.12.1 PHSM_STATUS.Request

This primitive is issued by either the Source SMT or Destination SMT to request a status report from the HIPPI-PH.

Semantics – PHSM_STATUS.Request

Issued – The SMT issues this primitive to the HIPPI-PH when it wishes to obtain the status of the interface.

Effect – The HIPPI-PH shall respond with a PHSM_STATUS.Confirm.

4.12.2 PHSM_STATUS.Confirm

This primitive replies to the previous PHSM_STATUS.Request with status information.

Semantics – PHSM_STATUS.Confirm (Status)

Status shall contain, but is not limited to:
INTERCONNECT state
Errors

Issued – The HIPPI-PH shall issue this primitive to the SMT to acknowledge a PHSM_STATUS.Request.

Effect – Unspecified

4.12.3 PHSM_STATUS.Indicate

This primitive informs the SMT entity that a major event has occurred that affects the operation of the HIPPI-PH.

Semantics – PHSM_STATUS.Indicate

Issued – The HIPPI-PH shall issue this primitive to the SMT whenever a major event is detected. Major events include but are not limited to a change in the INTERCONNECT signal.

NOTE – If a PHSM_CONTROL.Request was accepted successfully but not completed, then an PHSM_STATUS.Indicate could be used to indicate completion.

Effect – Upon receipt of this primitive, the local SMT entity should issue a PHSM_STATUS.Request to read the status of the HIPPI-PH and determine which event occurred.

4.12.4 PHSM_STATUS.Response

This primitive acknowledges the PHSM_STATUS.Indicate from the local SMT entity.

Semantics – PHSM_STATUS.Response

Issued – The SMT issues this primitive to acknowledge receipt of the PHSM_STATUS.Indicate.

Effect – The HIPPI-PH is enabled to issue another PHSM_STATUS.Indicate.

5 Interface format and signals

5.1 Physical framing hierarchy

Information shall be transferred across the HIPPI-PH as shown in figure 12. The interface signals are illustrated in figure 13.

5.1.1 Connections

Connections shall be delimited by both the REQUEST and CONNECT signals being true.

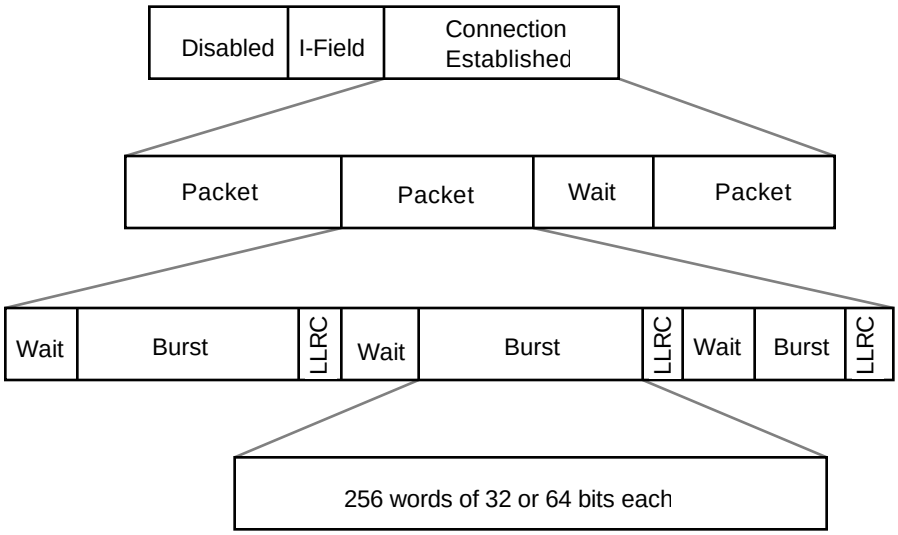
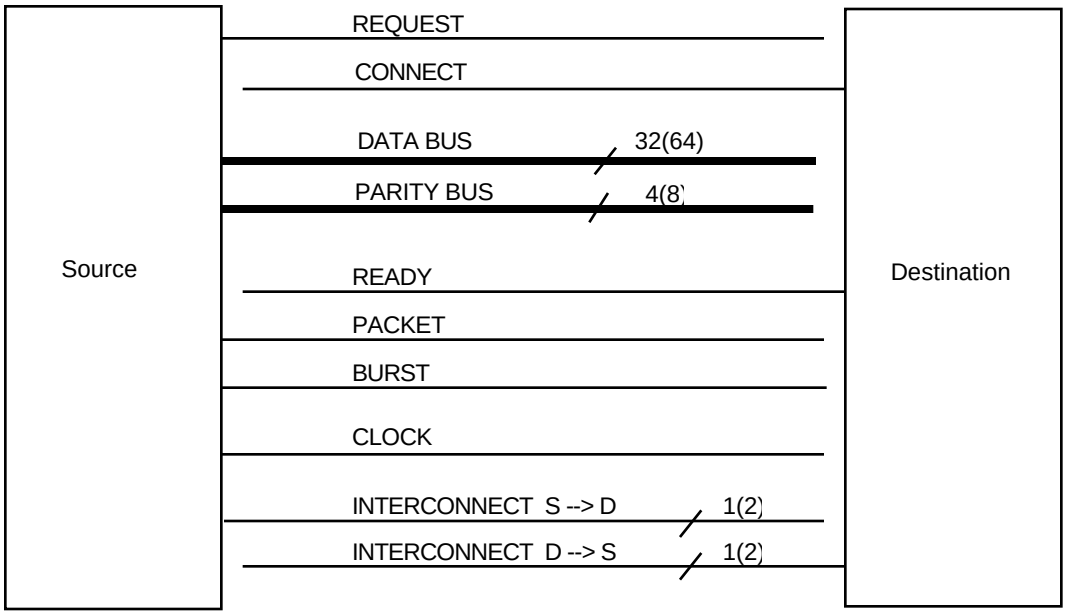


Figure 12 – Physical framing hierarchy



(Numbers in parentheses are for 1600 Mbit/s option)

Figure 13 – Interface signal summary

5.1.2 I-Field

The I-Field shall be delimited by the REQUEST signal being asserted and the CONNECT signal going true. The 32-bit I-Field may be used to supply non-specified control information when a connection is made. The I-Field timing is described in 7.5.

DATA BUS signals D00 through D31, on Cable-A, shall be used for the I-Field. The other signals of the word, if any, shall be zeros. Parity shall be correct for all bytes.

5.1.3 Packets

Packets shall be delimited by the PACKET signal being true. Once a connection has been established, a packet, or multiple packets, may be transferred from the Source to the Destination. Packets shall be composed of one or more bursts. For descriptive purposes only, packets are organized into classes based on the order of short bursts and full bursts.

– Class 1 shall include all packets which are composed of a single short burst.

– Class 2 shall include all packets which are composed of a short burst followed by one or more full bursts.

– Class 3 shall include all packets which are composed of one or more full bursts followed by a short burst.

– Class 4 shall include all packets which are composed of one or more full bursts and no short bursts.

5.1.4 Bursts

Bursts shall be delimited by the BURST signal being true. Bursts shall consist of a group of words sent on the DATA BUS, one word per CLOCK period, during contiguous clock periods. Bursts shall contain 1 to 256 words. Bursts that contain less than 256 words are called short bursts. A packet shall contain no more than one short burst. A short burst shall be either the first or last burst of a multi-burst packet.

5.1.5 LLRC

A length/longitudinal redundancy checkword (LLRC) shall be sent from the Source to the Destination on the DATA BUS during the first clock period following the burst. See annex C for an example LLRC calculation.

For data signals D00-D31, the LLRC value in data signal position *n* is defined as:

$$L_n = \left(\sum_{i=1}^{B} D_{ni} \right) + S_n$$

where

B is the number of words in the burst;

D_{ni} is the bit value in position *n* of word *i* of the burst;

S_n is the value of seed in position *n* where the seed is defined as B modulo 256.

All addition is modulo 2.

When the 1600-Mbit/s option is used the LLRC value in data signal position *n,* for data signals D32-D63, is defined as:

$$L_n = \left(\sum_{i=1}^{B} D_{ni} \right) + S_{(n-32)}$$

where $S_{(n-32)}$ is the value of the seed in position (*n* - 32) where the seed is defined as B modulo 256

All addition is modulo 2.

5.1.6 Wait time

The amount of wait time between packets and bursts may vary. The minimum wait times are specified in 7.8. Maximum wait times depend on the data flow to or from the upper-layer protocols and on the data flow to or from the opposite end of the interface. The contents of the DATA BUS and PARITY BUS are undefined during wait times.

5.2 Data rate options

The HIPPI-PH supports peak data rates of 800 and 1600 Mbit/s. These different data rates shall be achieved by using different widths for the DATA BUS. Table 1 summarizes the data rate options.

Table 1 – Data rate options

	Peak Data Rate (Mbit/s)	
	800	1600
Width of DATA BUS (bits)	32	64
Word size (bits)	32	64
Width of PARITY BUS (bits)	4	8
Size of I-Field (bits)	32	32
Maximum size of burst (bytes)	1024	2048
Number of cables per simplex interface	1	2

5.3 Usage of signals

Signal levels are "asserted" or "deasserted" at the driving end and "true" or "false" at the receiving end. The signal levels are more fully defined in 8.1.4.

5.3.1 INTERCONNECT

The INTERCONNECT signals shall indicate to both the Source and the Destination that the associated cable(s) are connected and that the other end is powered up. INTERCONNECT false means that all other signals are invalid. One wire-pair in each cable shall be used for the INTERCONNECT signals. The INTERCONNECT signals shall meet the electrical requirements described in 8.2. A recommended circuit for the INTERCONNECT signals is detailed in 8.2.

NOTE – It is recommended that the INTERCONNECT signal be sampled for multiple CLOCK times to insure that spurious noise does not cause false indications.

5.3.1.1 With 800-Mbit/s option

When the 800-Mbit/s data rate option is used, there shall be two INTERCONNECT signals: INTERCONNECT-A Source to Destination and INTERCONNECT-A Destination to Source. In this standard, INTERCONNECT at the Source shall mean the INTERCONNECT-A Destination to Source signal. INTERCONNECT at the Destination shall mean the INTERCONNECT-A Source to Destination signal.

5.3.1.2 With 1600-Mbit/s option

When the 1600-Mbit/s data rate option is used, there shall be four INTERCONNECT signals: INTERCONNECT-A Source to Destination, INTERCONNECT-B Source to Destination, INTERCONNECT-A Destination to Source, and INTERCONNECT-B Destination to Source. The -A and -B specify the cable number.

In this standard, INTERCONNECT at the Source shall mean the "logical and" of INTERCONNECT-A Destination to Source and INTERCONNECT-B Destination to Source. INTERCONNECT at the Destination shall mean the "logical and" of INTERCONNECT-A Source to Destination and INTERCONNECT-B Source to Destination.

5.3.2 DATA BUS

The DATA BUS shall consist of 32 or 64 signals labelled D00 through D31 or D00 through D63. As shown in table 2, D00 through D31 shall be in Cable-A, D32 through D63 shall be in Cable-B. The width of the DATA BUS controls the data rate of the HIPPI-PH as specified in table 1. The DATA BUS shall be used to transmit the I-Field, bursts, and LLRCs. The DATA BUS signals shall meet the timing requirements described in 7.3.

The data transferred between the ULP and the HIPPI-PH shall be 32-bit or 64-bit words, to match the HIPPI-PH DATA BUS width as defined in table 1. When mixing 32-bit and 64-bit systems and HIPPI-PHs, the data shall be packed as shown in figure 14. Clause B.6 describes the byte ordering within the words, which is specified by the HIPPI-FP framing protocol standard.

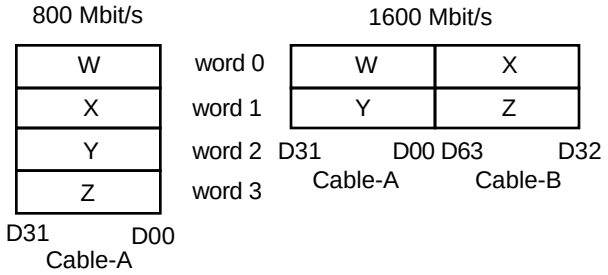


Figure 14 – Data packing

5.3.3 PARITY BUS

The PARITY BUS shall implement odd parity for each byte transmitted on the DATA BUS. The PARITY BUS signals shall meet the timing requirements described in 7.3. See annex C for an example calculation.

The transverse (byte) parity signal in position *n* for each word is defined as:

$$P_n = \left(\sum_{i=0}^{7} D_{8n+i} \right) + 1$$

where

n is the signal position on the PARITY BUS;

D_{8n+i} is the signal position on the DATA BUS.

All addition is modulo 2.

5.3.4 REQUEST

The REQUEST signal shall be asserted by the Source to notify the Destination that a connection is desired. The Source shall begin asserting the REQUEST signal only when the CONNECT signal is false. REQUEST shall remain asserted for the duration of the connection.

REQUEST shall be accompanied by an I-Field on the DATA BUS. The I-Field shall meet the specifications described in 5.1.2 and the timing requirements described in 7.5.

An aborted connection sequence occurs when the Source deasserts the REQUEST signal before sensing that the CONNECT signal has changed from false to true. After an aborted connection sequence, the Source shall wait for a time equivalent to the round-trip propagation delay between the Source and Destination, including the time required by any transparent line extension devices in the path, before reasserting REQUEST to start another connection sequence.

REQUEST shall be deasserted when INTERCONNECT is false or when CONNECT makes a transition from true to false. REQUEST may be deasserted by the Source at any time to indicate to the Destination that the connection is no longer available.

Once REQUEST is deasserted, the Source shall not reassert REQUEST until CONNECT is in the false state.

When REQUEST is false, the Destination shall assume that no other signals asserted by the Source, except CLOCK and INTERCONNECT, are valid.

The REQUEST signal shall meet the timing requirements described in 7.4.

5.3.5 CONNECT

The CONNECT signal shall be asserted by the Destination in response to a REQUEST. CONNECT notifies the Source that the Destination is available for data transfers. CONNECT shall remain asserted for the duration of the connection. When both REQUEST and CONNECT are true, a Connection is said to be established or available.

CONNECT shall not be asserted unless both INTERCONNECT and REQUEST are true. The Destination shall not assert CONNECT until it is acceptable to have the I-Field removed from the DATA BUS.

A rejected connection sequence occurs when the Destination asserts the CONNECT signal for at least four but no more than 16 CLOCK periods. The Destination shall not generate any READY indications during the rejected connection sequence.

CONNECT shall be deasserted if either INTERCONNECT or REQUEST is false. Once a connection has been established CONNECT may be deasserted by the Destination at any time to indicate to the Source that the connection is no longer available. Once CONNECT is deasserted, the Destination shall not reassert CONNECT until REQUEST has been received in the false state and returns to the true state.

When CONNECT is false, the Source shall assume that no other signals asserted by the Destination, except INTERCONNECT, are valid. The CONNECT signal shall meet the timing requirements described in 7.7.

5.3.6 READY

READY indications may be sent by the Destination after a connection is established, i.e., after CONNECT is asserted. A Destination shall send one READY indication for each burst that it is prepared to accept from the Source. This burst look-ahead function may be implemented in the Destination to minimize the effect of loop propagation delays that would otherwise slow throughput on long cables.

For each READY indication received, the Source has permission to send one burst. All Sources shall accept a minimum of 63 look-ahead READY indications.

The Destination shall deassert READY when CONNECT is deasserted.

Before a connection is established, the READY counts shall be cleared in both the Source and Destination.

The READY signal shall meet the timing requirements described in 7.7.

NOTES
1 – With the burst look-ahead function, the Destination controls the flow by enabling as many bursts as it can accept in its buffer memory. As long as the READY indications arrive at the Source before the Source is ready to send the next burst, there will be no time lost between bursts. Bursts of 256 words require about one burst buffer for each km of cable distance.

2 – Errors may cause READY indications to be lost. This can cause performance degradation or deadlock of the interface until a higher layer requests a hangup and a reconnect. For this reason, implementors should restrict the length of time that a connection is held depending on the error characteristics of the link.

5.3.7 PACKET

The PACKET signal is a delimiter asserted by the Source to mark a group of bursts as a packet. The PACKET signal shall be asserted before the first burst and shall remain asserted until after the final burst of the packet.

The PACKET signal shall be deasserted when REQUEST is deasserted.

The PACKET signal shall meet the timing requirements described in 7.4, 7.8, and 7.9.

5.3.8 BURST

The BURST signal is a delimiter marking a group of words on the DATA BUS as a burst. The BURST signal shall be asserted by the Source with the first word of the burst and shall remain asserted for all subsequent words of the burst.

The BURST signal shall be deasserted when PACKET or REQUEST is deasserted.

The BURST signal shall meet the timing requirements described in 7.4, 7.8, and 7.9.

5.3.9 CLOCK

The CLOCK signal shall be a symmetrical signal with a period of 40 ns (i.e., a frequency of 25 MHz). CLOCK shall be provided by the Source and shall run continuously. The CLOCK signal shall meet the timing requirements described in 7.1 and 7.2.

5.4 Error detection

5.4.1 Parity

Odd byte parity shall be transmitted on the PARITY BUS with each byte transmitted on the DATA BUS. This includes the I-Field, each word of every burst, and the LLRC.

5.4.2 LLRC

An LLRC, as specified in 5.1.5, shall be transferred across the DATA BUS during the first CLOCK period following each burst.

5.4.3 Destination error checking

The Destination is not required to check for any errors.

5.4.4 Recovery

The HIPPI-PH is a simplex point-to-point interface, therefore, retransmissions and error status reports from the Destination to the Source are beyond the scope of this standard.

6 State transitions

The HIPPI-PH service primitives and physical signals are tied together by the state transition pseudo-code and flow diagrams of this clause. The flow diagrams are included as a convenience and are not a rigorous definition.

The state transitions and flow diagrams do not describe the means or specific implementation by which the functions are provided. However, other implementations with fewer or additional states shall behave in a manner which is compatible with peer protocols implemented identical to the model.

Source states start with the letter S, Destination states with the letter D. Source states within this standard are numbered 1Jx0 where J = 0 - 4. Destination states within this document are numbered 1Kx0 where K = 5 - 9. This numbering scheme is used to avoid confusion between the Source and Destination states, and between states in this document and states in the data framing layer.

6.1 State exit

Within a state that is testing for some condition, the pseudo-code is assumed to loop indefinitely within the state until some exit condition is met. There is no prescribed order for testing the conditions except that the change in INTERCONNECTis an asynchronous event. INTERCONNECT going false takes the Source to S1160, and the Destination to D1660.

In the event that control sequence errors are detected, the Destination shall go to D1600. For purposes of reporting errors, control sequence errors take precedence over parity or LLRC errors. Control sequence errors include violations of 5.3, 7.8 and 7.9.

6.2 Interlocks

The implementor is cautioned that interlock flags or queues may be necessary to enforce the requirement of 4.1 that a second .Indicate primitive may not be issued by the HIPPI-PH until a .Response primitive of the same name has been received from the ULP. These interlocks are not illustrated in the pseudo-code.

6.3 Source READY pseudo-code

The Source READY flow diagram in figure 15 gives an overview of the Source READY pseudo-code. The S_READY_Ctr counts READY indications received from the Destination. Each READY indication gives the Source permission to send a burst. The Source decrements the S_READY_Ctr for each burst it sends to the Destination.

6.3.1 SR1000

Initialize the S_READY_Ctr. Enter on Power-up, Master Reset, or when the connection is broken.

```
Reset S_READY_Ctr
Goto SR1010
```

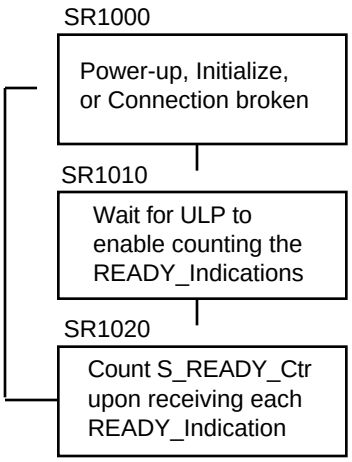


Figure 15 – Source READY flow diagram

6.3.2 SR1010

Wait for the S_READY_Ctr to be enabled by the Source pseudo-code S1190.

```
IF S_READY_Enable = set
   THEN Goto SR1020
```

6.3.3 SR1020

Count READY indications received from the Destination.

```
IF READY_Indication
   THEN Increment S_READY_Ctr
IF S_READY_Enable = reset
   THEN Goto SR1000
```

6.4 Destination READY pseudo-code

The Destination READY flow diagram in figure 16 gives an overview of the Destination READY pseudo-code. Each READY indication issued gives the Source permission to send a burst.

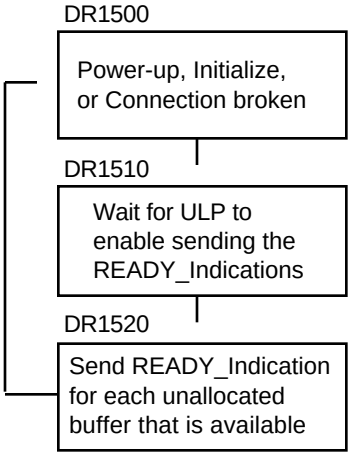


Figure 16 – Destination READY flow diagram

6.4.1 DR1500

Initialize the circuitry that keeps track of the available burst buffers in the Destination. Enter on Power-up, Master Reset, or when the connection is broken.

```
Deassert READY
Reset Available_Buffers_Logic
Goto DR1510
```

6.4.2 DR1510

Wait for the Destination pseudo-code D1690 to enable sending READY indications. The connection must be accepted with CONNECT true before sending READY indications.

```
IF D_READY_Enable = set
   THEN Goto DR1520
```

6.4.3 DR1520

Send READY indications as buffers are freed up.

```
IF Unallocated Burst Buffer is Available
   THEN Issue READY_Indication
IF D_READY_Enable = reset
   THEN Goto DR1500
```

6.5 Source pseudo-code

The Source flow diagram in figure 17 gives an overview of the Source pseudo-code.

6.5.1 S1100

Disabled state; initialize the HIPPI-PH and relevant local state variables. Enter on Power-up or system master reset.

```
Deassert BURST
Deassert PACKET
Deassert REQUEST
Reset S_READY_Enable
Goto S1110
```

6.5.2 S1110

Check the INTERCONNECT signal from the Destination. Wait for old REQUEST and CONNECT signals to die out on the cable during time T1 (see 4.5.1).

```
IF INTERCONNECT = false
   THEN Goto S1140
Wait time T1
Goto S1130
```

6.5.3 S1120

Inform the Source SMT that the INTERCONNECT signal has gone to the true state.

```
Issue PHSM_STATUS.Indicate
Goto S1110
```

6.5.4 S1130

Idle state; the interface is available, but not connected. Wait for the Source ULP to request a connection.

```
IF INTERCONNECT = false
   THEN Goto S1160
IF PH_RING.Request (CCI)
   AND IF CONNECT = false
      THEN Goto S1150   ; Start connection
```

6.5.5 S1140

Wait for the INTERCONNECT signal to go true.

```
IF INTERCONNECT = true
   THEN Goto S1120
```

6.5.6 S1150

The Source ULP has requested that a connection sequence be started. Use the information in the Source ULP CCI parameter as the I-Field.

```
Issue PH_RING.Confirm
Assert CCI on DATA BUS as I-Field
Assert REQUEST
Goto S1170
```

6.5.7 S1160

Inform the Source SMT that the INTERCONNECT signal has gone false.

```
Deassert BURST
Deassert PACKET
Deassert REQUEST
Reset S_READY_Enable
Issue PHSM_STATUS.Indicate
Goto S1140
```

6.5.8 S1170

A connection sequence is in process. Wait for the Destination to complete the connection, either accepted or rejected, or for the Source ULP to abort the connection request.

```
IF INTERCONNECT = false
   THEN Goto S1160
IF PH_HANGUP.Request
   THEN Goto S1180     ; Abort connection
IF CONNECT = true
   THEN Goto S1190     ; Connection completed
```

6.5.9 S1180

The connection is being aborted by the Source ULP before the Destination has completed the connection, i.e., the CONNECT signal has not been received. Time T1, as defined in 4.5.1, allows the REQUEST and CONNECT signals to propagate and settle before allowing another connection request sequence.

```
Deassert REQUEST
Issue PH_HANGUP.Confirm
Reset S_READY_Enable
Wait time T1
Goto S1130
```

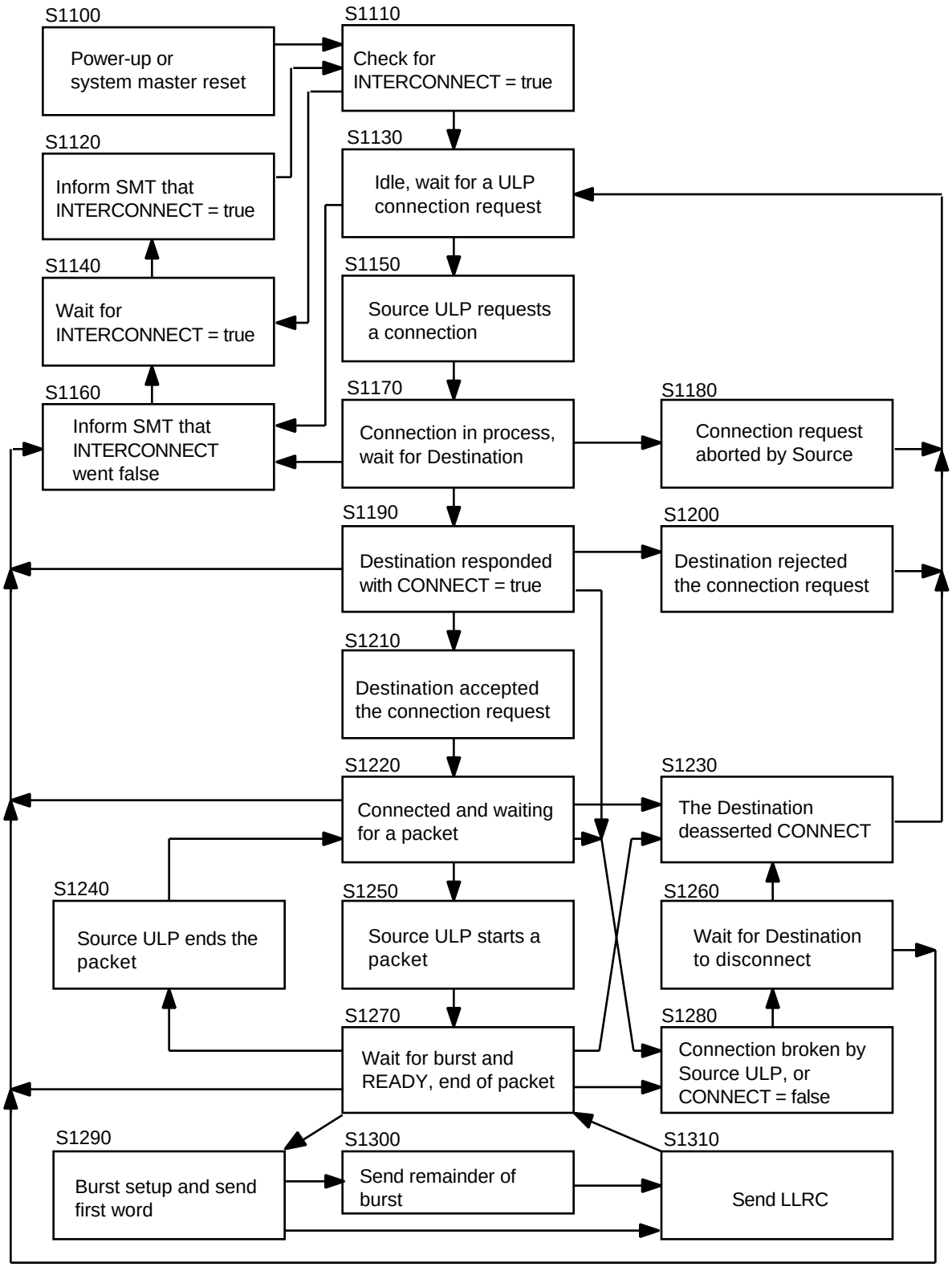


Figure 17 – Source flow diagram

6.5.10 S1190

The Destination has responded with the CONNECT signal true. Wait for the CONNECT signal to be true for at least 17 CLOCK periods, or receiving a READY indication, before signalling that the connection has been accepted. The Destination may reject this connection request by asserting CONNECT for a time between 4 and 16 CLOCK periods and then deasserting CONNECT.

```
Set S_READY_Enable
IF INTERCONNECT = false
   THEN Goto S1160
IF PH_HANGUP.Request
   THEN Goto S1280     ; Abort connection
IF CONNECT = false
   THEN Goto S1200     ; Connection rejected
IF S_READY_Ctr > 0
   THEN Goto S1210     ; Connection accepted
IF CONNECT = true
   AND 17 CLOCK periods have expired
      THEN Goto S1210  ; Connection accepted
```

6.5.11 S1200

Inform the Source ULP that the Destination has rejected the connection by deasserting CONNECT before sending a READY indication.

```
Issue PH_ANSWER.Indicate (Reject)
Deassert REQUEST
Reset S_READY_Enable
Goto S1130
```

6.5.12 S1210

Inform the Source ULP that the connection has been established.

```
Issue PH_ANSWER.Indicate (Accept)
Goto S1220
```

6.5.13 S1220

The connection has been established and the interface is now available for packet transfers.

```
IF INTERCONNECT = false
   THEN Goto S1160
IF CONNECT = false
   THEN Goto S1230     ; Connection broken
IF PH_PACKET.Request (Begin)
   THEN Goto S1250     ; Start of packet
IF PH_HANGUP.Request
   THEN Goto S1280     ; Break connection
```

6.5.14 S1230

Inform the Source ULP that the Destination has broken the connection by deasserting CONNECT.

```
Issue PH_HANGUP.Indicate
Deassert PACKET
Deassert REQUEST
Reset S_READY_Enable
Goto S1130
```

6.5.15 S1240

The Source ULP has terminated a packet. Deassert the PACKET signal and acknowledge the ULP primitive.

```
Deassert PACKET
Issue PH_PACKET.Confirm (Accept)
Goto S1220
```

6.5.16 S1250

The Source ULP has started a packet. Assert the PACKET signal and acknowledge the ULP primitive.

```
Assert PACKET
Issue PH_PACKET.Confirm (Accept)
Goto S1270
```

6.5.17 S1260

Wait for Source ULP breaking of the connection to be acknowledged by the Destination deasserting CONNECT.

```
IF INTERCONNECT = false
   THEN Goto S1160
IF CONNECT = false
   THEN Goto S1230     ; Connection broken
```

6.5.18 S1270

A packet has been started; wait for a burst from the Source ULP or an end of packet primitive. The Word_Counter shall be used to count the number of words in the burst.

```
IF INTERCONNECT = false
   THEN Goto S1160
IF CONNECT = false
   THEN Goto S1230     ; Connection broken
IF PH_PACKET.Request (End)
   THEN Goto S1240     ; End of packet
IF PH_HANGUP.Request
   THEN Goto S1280     ; Break connection
Reset Word_Counter
Reset LLRC calculation
IF PH_TRANSFER.Request (Length, Burst)
   AND IF S_READY_Ctr > 0
      THEN Goto S1290
```

6.5.19 S1280

Break the connection from the Source end. The connection was completed, i.e., a PH_ANSWER.Indicate (Accept) has been issued.

```
Deassert REQUEST
Deassert PACKET
Reset S_READY_Enable
Issue PH_HANGUP.Confirm
Goto S1260
```

6.5.20 S1290

There is a burst to transfer and a READY indication available. Acknowledge the burst primitive, decrement the READY counter, and send the first word.

```
Convert Length to words
Initialize LLRC with Length
Issue PH_TRANSFER.Confirm (Accept)
Decrement S_READY_Ctr
Assert BURST
Assert  first word of burst on DATA BUS
Accumulate LLRC
Count CLOCK period in Word_Counter
IF Word_Counter = Length
   THEN Deassert BURST
      Goto S1310
Goto S1300
```

6.5.21 S1300

Send the remainder of the burst to the Destination.

```
Assert words of burst on DATA BUS
Accumulate LLRC
Count CLOCK periods in Word_Counter
IF Word_Counter = Length
   THEN Deassert BURST
      Goto S1310
```

6.5.22 S1310

Send the LLRC associated with the previous burst to the Destination.

```
Assert LLRC on DATA BUS
Goto S1270
```

6.6 Destination pseudo-code

The Destination flow diagram in figure 18 gives an overview of the Destination pseudo-code.

6.6.1 D1600

Disabled state; initialize the HIPPI-PH and relevant local state variables. Enter on Power-up or system master reset.

```
Reset D_READY_Enable
Deassert CONNECT
Goto D1610
```

6.6.2 D1610

Check the INTERCONNECT signal from the Source. Wait for any previous connection attempts to be terminated, i.e., REQUEST = false.

```
IF INTERCONNECT = false
   THEN Goto D1640
IF REQUEST = false
   THEN Goto D1630
```

6.6.3 D1620

Inform the Destination SMT that the INTERCONNECT signal has gone to the true state.

```
Issue PHSM_STATUS.Indicate
Goto D1610
```

6.6.4 D1630

Idle state; the interface is available but not connected. Wait for the Source to request a connection.

```
IF INTERCONNECT = false
   THEN Goto D1660
IF REQUEST = true
   THEN Goto D1650      ; Connection request
```

6.6.5 D1640

Wait for the INTERCONNECT signal to be true.

```
IF INTERCONNECT = true
   THEN Goto D1620
```

6.6.6 D1650

The Source has requested a connection. Use the information in the I-Field as the CCI parameter passed to the Destination ULP. Note that this CCI may be different from the CCI supplied by the PH_RING.Request primitive (in 6.5.6 S1150) due to the action of intermediate devices, such as switches, between the Source and Destination.

```
Issue PH_RING.Indicate (CCI)
Goto D1670
```

6.6.7 D1660

Inform the Destination SMT that the INTERCONNECT signal has gone false.

```
Reset D_READY_Enable
Deassert CONNECT
Issue PHSM_STATUS.Indicate
Goto D1640
```

6.6.8 D1670

A connection sequence is in process. Wait for the Destination ULP to complete the connection, either accepted or rejected, or for the Source to abort the connection request.

```
IF INTERCONNECT = false
   THEN Goto D1660
IF REQUEST = false
   THEN Goto D1680      ; Connection request aborted
IF PH_ANSWER.Request (Accept)
   THEN Goto D1690      ; Connection accepted
IF PH_ANSWER.Request (Reject)
   THEN Goto D1700      ; Connection rejected
```

6.6.9 D1680

The connection is being aborted from the Source end before the Destination has completed the connection.

```
Issue PH_HANGUP.Indicate
Goto D1630
```

6.6.10 D1690

Inform the Destination ULP that the connection has been completed.

```
Issue PH_ANSWER.Confirm
Assert CONNECT
Set D_READY_Enable
Goto D1710
```

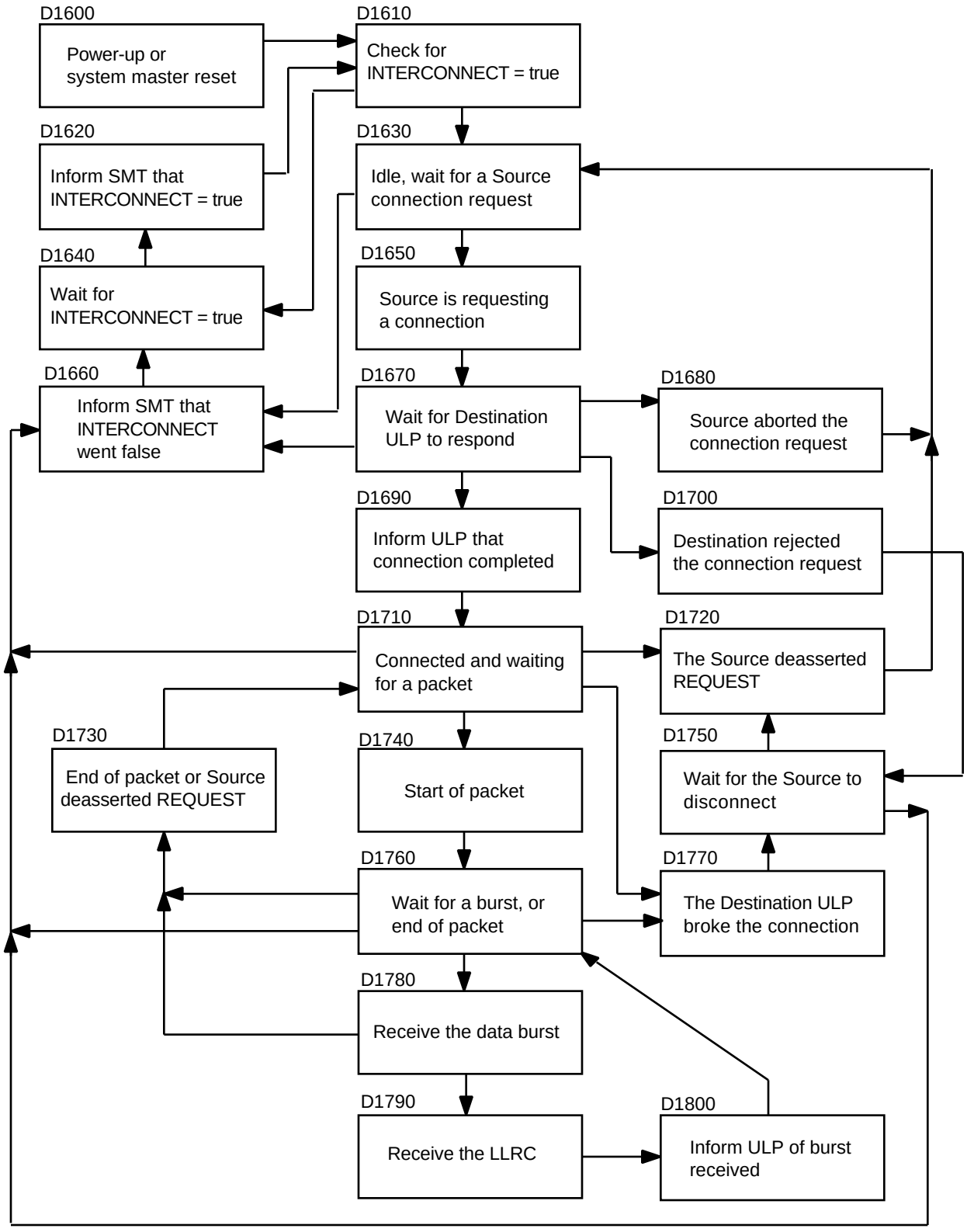


Figure 18 – Destination flow diagram

6.6.11 D1700

The Destination rejects the connection by asserting the CONNECT signal for a period of between 4 and 16 CLOCK periods and then deasserting CONNECT.

```
Issue PH_ANSWER.Confirm
Assert CONNECT
Wait 4 to 16 CLOCK periods
Deassert CONNECT
Goto D1750
```

6.6.12 D1710

The connection has been established and the interface is now available for packet transfers.

```
IF INTERCONNECT = false
   THEN Goto D1660
IF REQUEST = false
   THEN Goto D1720     ; Source broke connection
IF PACKET = true
   THEN Goto D1740     ; Start of packet
IF PH_HANGUP.Request
   THEN Goto D1770     ; Break connection
```

6.6.13 D1720

Inform the Destination ULP that the Source has deasserted REQUEST.

```
Issue PH_HANGUP.Indicate
Reset D_READY_Enable
Deassert CONNECT     ; If asserted
Goto D1630
```

6.6.14 D1730

Inform the Destination ULP that the Source has ended the packet.

```
Issue PH_PACKET.Indicate (End, Status)
Goto D1710
```

6.6.15 D1740

Inform the Destination ULP that the Source has started a packet.

```
Issue PH_PACKET.Indicate (Begin)
Goto D1760
```

6.6.16 D1750

Wait for the Destination ULP breaking of the connection to be acknowledged by the Source deasserting REQUEST.

```
IF INTERCONNECT = false
   THEN Goto D1660
IF REQUEST = false
   THEN Goto D1720     ; Source broke connection
```

6.6.17 D1760

A packet has been started; wait for another burst or an end of packet indication. The Word_Counter shall be used to count the number of words in the burst.

```
IF INTERCONNECT = false
   THEN Goto D1660
IF REQUEST = false
   THEN Goto D1730     ; Source broke connection
IF PACKET = false
   THEN Goto D1730     ; End of packet
IF PH_HANGUP.Request
   THEN Goto D1770     ; Break connection
Reset Word_Counter
Reset LLRC calculation
IF BURST = true
   THEN Goto D1780     ; Receive burst
```

6.6.18 D1770

Break the connection from the Destination end.

```
Reset D_READY_Enable
Deassert CONNECT
Issue PH_HANGUP.Confirm
Goto D1750
```

6.6.19 D1780

Receive the burst being sent from the Source.

```
Count CLOCK periods in Word_Counter
Receive a burst on the DATA BUS
Check parity on each word
Accumulate LLRC
IF REQUEST = false
   THEN Set Status = Error for PH_PACKET.Indicate
        Goto D1730     ; Source disconnect
IF BURST = false
   THEN Goto D1790
```

6.6.20 D1790

Receive the Source-generated LLRC for the burst.

```
Receive LLRC on DATA BUS
Goto D1800
```

6.6.21 D1800

Inform the Destination ULP that the burst has been received. The Word_Counter shall be used to derive the Length parameter.

```
Issue PH_TRANSFER.Indicate (Status,Length, Burst)
Goto D1760
```

7 Timing

Unless otherwise specified, all parameters shall be measured at the bulkhead connector of the Source or Destination equipment. Specifications shall be met when operating with a maximum length cable, as specified in 8.5, and terminated as specified in 8.1.3.

Example timing waveforms are shown in figure 19 and figure 20. The falling edge of a signal is defined as the Pin+ side of the differential pair, as shown in figure 20, going from the more-positive level to the less-positive level. Time intervals for differential signals are measured from the cross-over point where the differential voltage is zero.

Annex A contains waveform examples.

7.1 Source CLOCK signal

The transmitted CLOCK signal at the Source bulkhead connector, as shown in figure 19, shall have a nominal period TPav of 40 ns ± 0.004 ns (± 0.01 %).

CLOCK symmetry, measured as the percentage of time in the high state compared to the total CLOCK period, shall be 50(± 5) %. Peak jitter shall be less than 0.5 ns.

The minimum time TPmin between adjacent falling edges of the CLOCK shall be 38.996 ns [40 - 0.004 - (2 x 0.5)] ns. The maximum time TPmax between adjacent falling edges of the CLOCK shall be 41.004 ns [40 + 0.004 + (2 x 0.5)] ns.

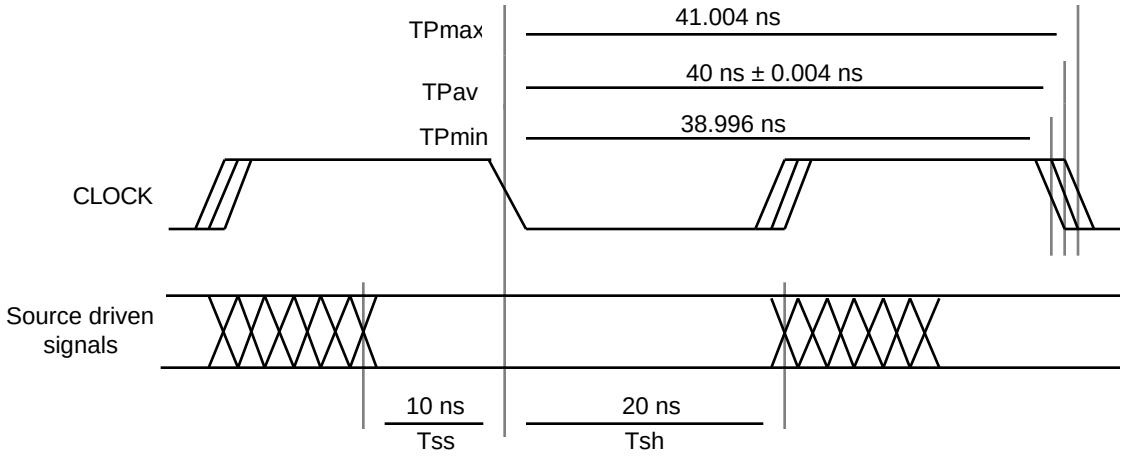


Figure 19 – Source driven signals at the Source

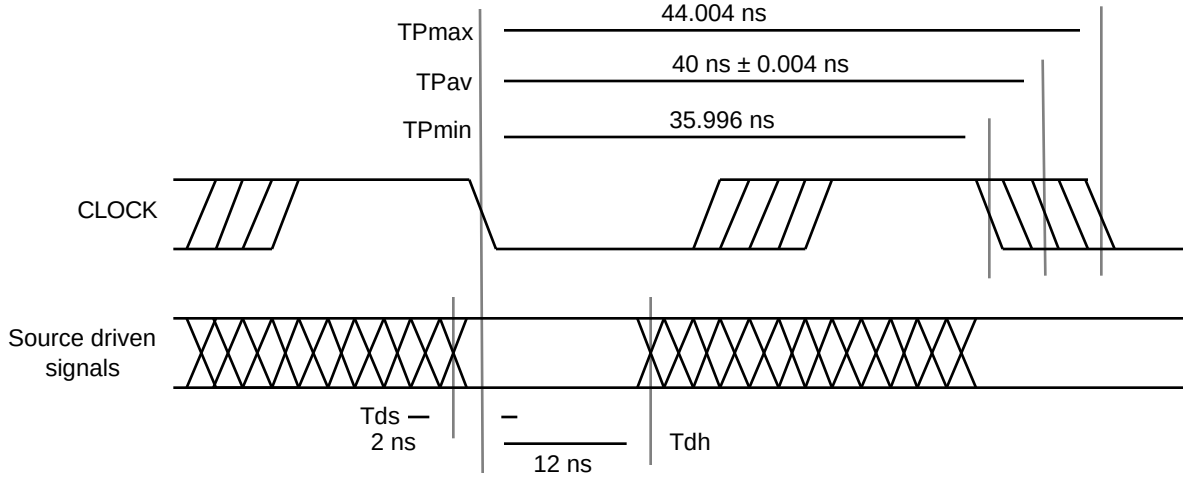


Figure 20 – Source driven signals at the Destination

In addition, to ensure that the short-term average frequency within a burst is equal to the long-term average frequency, the elapsed time between any two falling edges of the CLOCK within a single burst, separated by N CLOCK periods, shall be between N x (40 - 0.004) ns - 1 ns and N x (40 + 0.004) ns + 1 ns.

7.2 Destination CLOCK signal

The received CLOCK signal at the Destination bulkhead connector, as shown in figure 20, shall have a nominal period TPav of 40 ns with a tolerance of ± 0.004 ns (± 0.01 %). CLOCK symmetry, measured as the percentage of time in the high state compared to the total CLOCK period, shall be 50 % ± 10 %. Peak jitter shall be less than 2.0 ns.

The minimum time TPmin between adjacent falling edges of the CLOCK shall be 35.996 ns [40 - 0.004 - (2 x 2.0)] ns. The maximum time TPmax between between adjacent falling edges of the CLOCK shall be 44.004 ns [40 + 0.004 + (2 x 2.0)] ns.

In addition, to ensure that the short-term average frequency within a burst is equal to the long-term average frequency, the elapsed time between any two falling edges of the CLOCK within a single burst, separated by N CLOCK periods, shall be between N x (40 - 0.004) ns - 4 ns and N x (40 + 0.004) ns + 4 ns.

7.3 DATA BUS and PARITY BUS timing

The Source shall assert information on the DATA BUS, with odd byte parity on the PARITY BUS, so that all data and parity lines are stable for a period not less than Tss = 10 ns before the falling edge of the CLOCK signal, and not less than Tsh = 20 ns after the falling edge (see figure 19).

The Destination shall operate within specifications with all DATA BUS and PARITY BUS signals stable for a setup time of Tds = 2 ns and a hold time of Tdh = 12 ns. Both setup and hold times are measured with respect to the falling edge of the CLOCK signal (see figure 20).

7.4 Source control signals

The Source shall assert the REQUEST, PACKET, and BURST signals to meet the setup time as specified in 7.3 for the DATA BUS. For the final clock period, during which they are asserted, they shall meet the hold time as specified in 7.3 for the DATA BUS.

The end of a packet shall be signalled by the Source deasserting the PACKET signal one or more CLOCK periods after the deassertion of BURST. Both PACKET and BURST shall be deasserted simultaneously when the Source enters the Idle or Disabled states.

7.5 I-Field information

The Source shall place the I-Field information on the DATA BUS, with odd byte parity on the PARITY BUS, before or with the same CLOCK edge used to assert the REQUEST signal. The I-Field shall meet the setup times as specified in 7.3 for the DATA BUS.

The I-Field shall remain asserted until CONNECT is received from the Destination or until the Source deasserts REQUEST.

7.6 LLRC

The LLRC shall be placed on the DATA BUS, with odd byte parity on the PARITY BUS, during the first CLOCK period after BURST is deasserted. The LLRC shall meet the same setup and hold times as specified in 7.3 for the DATA BUS.

7.7 Destination control signals

When asserted, the CONNECT and READY signals shall be asserted for at least four CLOCK periods. When deasserted, the CONNECT and READY signals shall be deasserted for at least four CLOCK periods.

After asserting CONNECT, the Destination shall wait at least four CLOCK periods before asserting the READY signal.

No clocking accompanies the CONNECT and READY signals from the Destination. A CONNECT or READY signal that is stable at the Source for two consecutive CLOCK periods shall be considered a valid signal.

7.8 Source wait gaps

The Source shall provide at least the following number of CLOCK periods between the control signals during normal operation. All of the control signals may be deasserted simultaneously when going to the Disabled state.

– PACKET asserted to BURST asserted = 1
– PACKET asserted to PACKET deasserted = 3
– PACKET deasserted to PACKET asserted =2
– BURST deasserted to PACKET deasserted = 1
– BURST deasserted to BURST asserted = 3

(This applies only within a packet.)

NOTE – For any of these gaps, greater numbers of CLOCK periods may be inserted by the Source to accommodate upper-layer protocols, buffer management hardware, and the like, but with the realization that larger gaps will decrease throughput.

7.9 Destination wait gaps

The Destination shall operate with the following minimum number of CLOCK periods between signals during normal operation. All of the control signals may be deasserted simultaneously when going to the Disabled state.

– PACKET asserted to BURST asserted = 1
– PACKET asserted to PACKET deasserted = 3
– PACKET deasserted to PACKET asserted =1
– BURST deasserted to PACKET deasserted = 1
– BURST deasserted to BURST asserted = 2

(This applies only within a packet.)

NOTE – The number of intermediate CLOCK periods at the Destination may be smaller than specified in 7.8 for the Source. The extra CLOCK period is supplied by the Source so that an optional intermediate repeater will have a CLOCK period (other than the one marking the LLRC) that may be deleted during signal resynchronization.

8 Physical characteristics

Unless otherwise specified, all parameters shall be measured at the bulkhead connector of the Source or Destination equipment. Specifications shall be met when operating with a maximum length cable terminated as specified in 8.1.3.

V_{EE} is the negative supply voltage for the Emitter Coupled Logic compatible (ECL) drivers and receivers.

8.1 Differential circuit characteristics

Uni-directional ECL differential line drivers and receivers shall be used for all interface signals (see figure 21 for a representative differential circuit) except for the INTERCONNECT signal.

8.1.1 Line drivers

All differential line drivers shall meet the following specifications. Rise and fall times shall be measured between the 20 % and 80 % points of the signal transition.

– Maximum high-level output voltage; VDHmax = -0.75 V

– Minimum high-level output voltage; VDHmin = -1.00 V

– Maximum low-level output voltage; VDLmax = -1.60 V

– Minimum low-level output voltage; VDLmin = -2.00 V

– Minimum rise time = 0.5 ns

– Maximum rise time = 2.3 ns

– Minimum fall time = 0.5 ns

– Maximum fall time = 2.3 ns

8.1.2 Line receivers

All differential line receivers shall operate correctly when receiving signals meeting the following voltage specifications at the Destination connector.

– Common mode input voltage range;
VRcmr = -0.9 to -2.5 V

– Minimum peak-to-peak differential input voltage;
VRdif min = 300 mV

– Maximum peak-to-peak differential input voltage;
VRdif max = 1.1 V

NOTE – The voltage at the input to a receiver is the sum of the instantaneous common mode voltage and one half of the simultaneous differential mode voltage. For example, with the nominal common mode voltage of -1.30 V and a differential mode voltage of 1.0 V, the input will swing from -0.8 V max to -1.8 V min.

8.1.3 Differential line terminators

Each differential signal shall use a twisted-pair conductor line and terminator resistors as shown in figure 21. The driving end termination shall be 330 Ω (± 2 %) to voltage V_{EE} from each line of the twisted-pair. The receiving end termination shall be 110 Ω (± 2 %) across the lines of the twisted-pair.

8.1.4 Electrical/logic levels

Figure 21 shows the logic conventions used for the differential signals. Active control signals shall be "Asserted" at the driver end, have the Pin+ more positive than the Pin-, and shall be "True" or called by the signal name at the receiver end. Nonactive signals shall be "Deasserted" at the driver end, have the Pin- more positive than the Pin+, and shall be "False" or called by putting "not" in front of the signal name (e.g., "not CONNECT") at the receiver end. Similarly, a logical 1 at the driver shall result in the Pin+ being more positive than Pin-, and a logical 1 at the receiver.

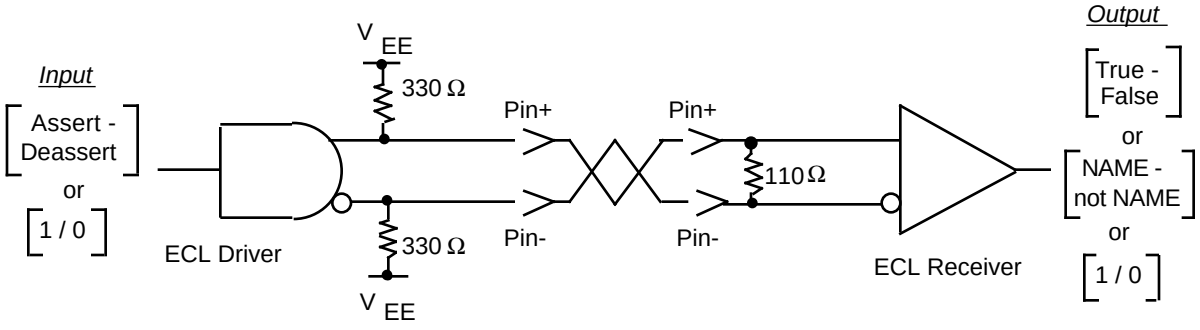


Figure 21 – Differential circuit

8.2 INTERCONNECT signal characteristics

The two signals (INTERCONNECT Source to Destination and INTERCONNECT Destination to Source) share a cable wire-pair. The recommended circuit for each of these signals is shown in figure 22. Normal conditions (cable connected and power on at both ends) result in a low-level input to the ECL receiver. Error conditions (the cable disconnected or power lost at the drive end of the circuit) result in a high-level input to the ECL receiver.

8.2.1 INTERCONNECT voltage levels

In the normal (or true) state, the INTERCONNECT line voltage shall meet the following specifications (see figure 22).

– Maximum low-level voltage; VILmax = -1.6 V

– Minimum low-level voltage; VILmin = -3.7 V

In the error (or false) state, the INTERCONNECT line voltage shall meet the following specifications (see figure 22).

– Maximum high-level voltage; VIHmax = 0 V

– Minimum high-level voltage; VIHmin = -1.0 V

8.2.2 INTERCONNECT line terminator

The INTERCONNECT drive terminator shall be 220 Ω (± 5 %) to voltage V_{EE}. The INTERCONNECT load terminator shall be 220 Ω (± 5 %) to ground (0 V).

8.3 Ground signals

Connector pins labelled Ground in table 2 shall have each wire of the pair returned to ground (0 V).

8.4 Reserved signals

Signals labelled "Reserved" in table 2 are reserved for future HIPPI-PH options and shall not be used. If implemented, drivers and receivers for the Reserved signals shall meet the signal directions specified in table 2 where SD refers to Source to Destination, and DS refers to Destination to Source. Reserved signals shall be terminated as in 8.1.3.

8.5 Cable specifications

The cable shall consist of 50 twisted pairs of 0.09 mm^2 (#28 AWG 7/36) tinned copper. The cable shall have two overall shields, one foil and one braid. The shields shall be connected to the connector shell at each end of the cable. The following electrical properties shall apply to the twisted pairs within the bulk cable assembly:

– Characteristic impedance = 108 Ω (+6 Ω, - 5 Ω) (differential TDR)

– Signal attenuation max = 0.28 dB/m (0.085 dB/ft) at 50 MHz

– Pair-to-pair propagation delay delta max = 0.13 ns/m (0.04 ns/ft)

– Conductor DC resistance max = 0.23 Ω/m (0.070 Ω/ft) at 20 °C

– Pair-to-shield max delta capacitance = 2.6 pF/m (0.8 pF/ft)

The cable foil and braid shields shall be electrically connected. The following properties shall apply to the over-all bulk cable assembly:

– Foil shield = 0.025 mm (0.001 in), min 25 % lap

– Drain wire = 0.09 mm^2 (#28 AWG 7/36)

– Braid shield = 0.013 mm^2 (#36 AWG), min 80 %coverage

– Jacket wall avg = 0.889 mm (0.035 in)

– Jacket material = 80 °C Flexible PVC

– Outside diameter = 12.32 mm ± 0.38 mm (0.485 in ± 0.015 in)

The maximum cable length shall be 25 m (82 ft). See annex E for recommended cable lengths and other cable options.

When Cable-B is used to implement the 1600-Mbit/s data transfer rate option, then the maximum pair-to-pair propaga-tion delay delta across the full set of twisted pairs in Cable-A and Cable-B shall be less than or equal to 4.5 ns.

NOTE – The cable should meet the appropriate requirements for fire and safety, for example, the 1990 USA National Electrical Code (NEC) sections 725-51(c) and 725-53(a).

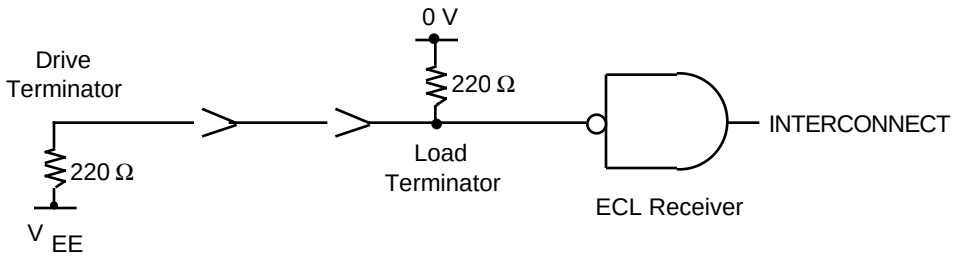


Figure 22 – INTERCONNECT circuit

8.6 Cable grounding

The cable shield shall be connected to chassis ground, through the connector shells, at each end of the cable. For safety reasons the equipments connected shall be on the same facility ground.

Because differential receivers are used and they have a limited common mode voltage tolerance, and because a single-ended circuit is used for INTERCONNECT, care must be taken by the user to insure that the equipments are installed such that the stated common mode input voltage range of the receiver is not exceeded. It is outside the scope of this standard to specify the exact method needed in any particular installation, however methods such as a local common ground grid may be required.

8.7 Connector specifications

The cable connectors shall be two-row, 100-pin shielded tab connectors. See figure 23.

The bulkhead connectors shall be two-row, 100-pin panel mount receptacles. See figure 24.

The shells of the connectors shall provide radio-frequency interference (RFI) shielding and ensure the integrity of the cable shield to chassis current path. The resistance of the cable shield to equipment chassis shall not exceed 75 mΩ, after a minimum of 500 cycles of mating and unmating.

NOTE – The use of an optional alignment guide on the panel, as shown in annex E, is recommended for mounting the connectors in the horizontal direction.

8.8 Connector pin assignments

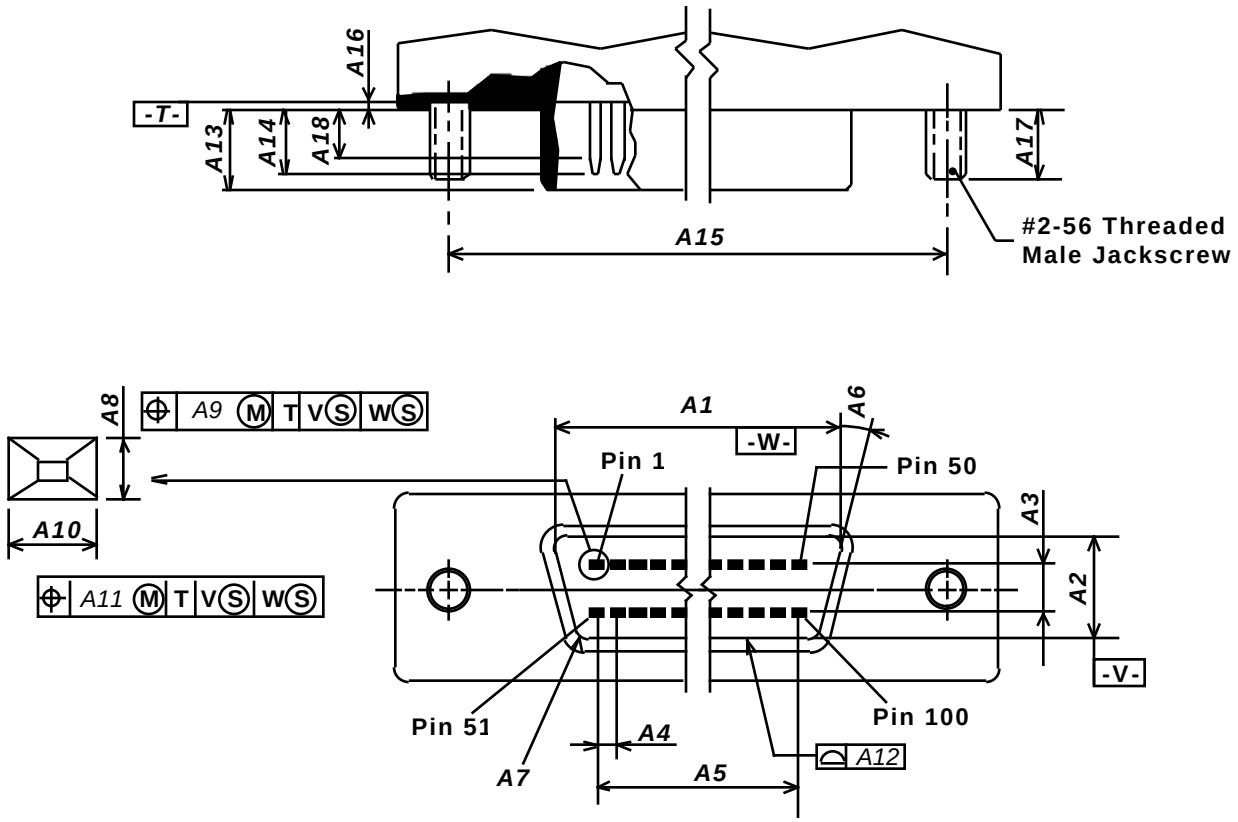
The signal assignments follow the modular definition of the interface given in table 1. An 800-Mbit/s HIPPI-PH uses Cable-A only. A 1600-Mbit/s HIPPI-PH uses both Cable-A and Cable-B. Connector pin assignments are detailed in table 2.

Within this standard Source bulkhead connectors are designated TA and TB and the Destination bulkhead connectors are designated RA and RB.

NOTE – Cable-A (TA to RA) is physically the same as Cable-B (TB to RB). They differ only in the signal assignments.

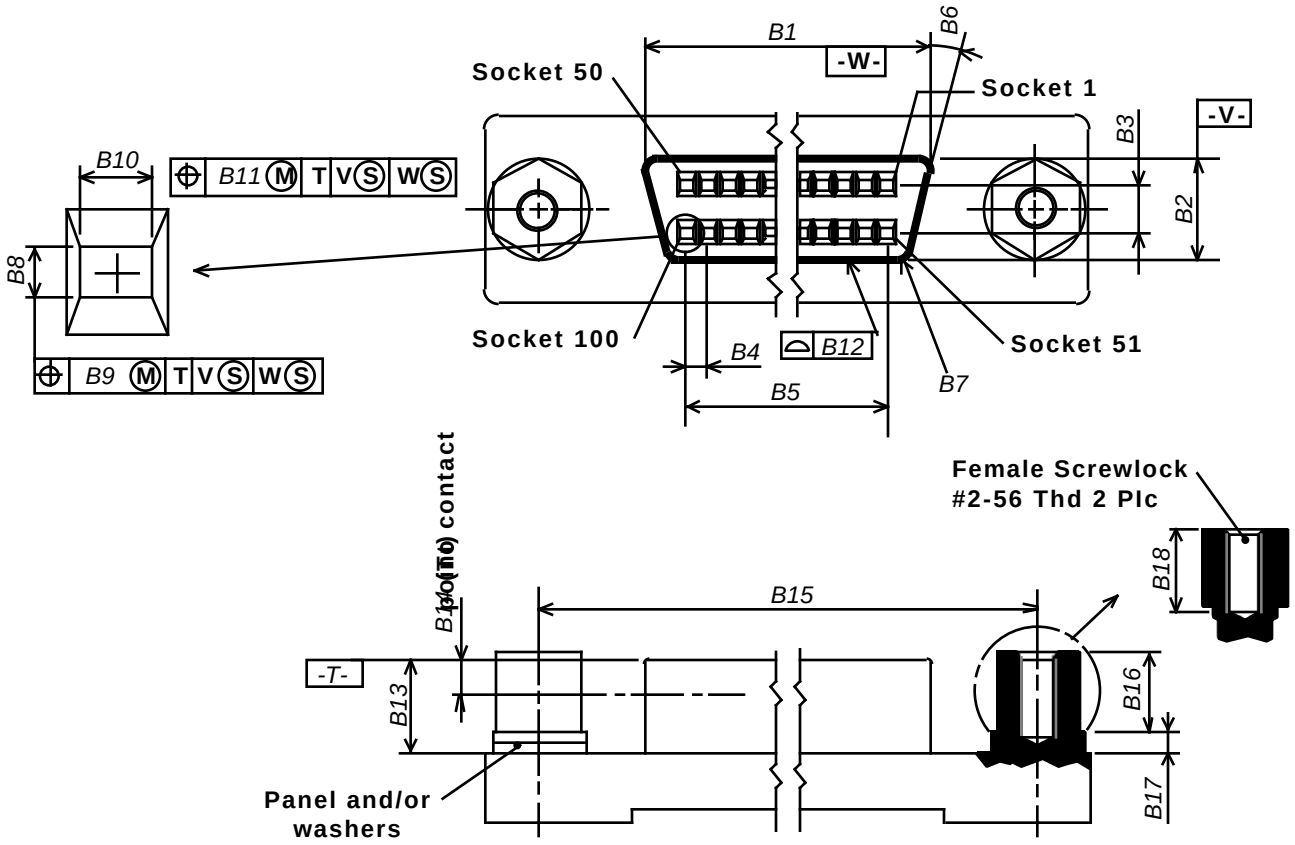
Table 2 – Connector pin assignments

Pin+	Pin-	Cable-A	Cable-B	Notes
1	2	D00	D32	
51	52	D01	D33	
3	4	D02	D34	
53	54	D03	D35	
5	6	D04	D36	
55	56	D05	D37	
7	8	D06	D38	
57	58	D07	D39	
9	10	P0	P4	
59	60	D08	D40	
11	12	D09	D41	
61	62	D10	D42	
13	14	D11	D43	
63	64	D12	D44	
15	16	D13	D45	
65	66	D14	D46	
17	18	D15	D47	
67	68	P1	P5	
19	20	D16	D48	
69	70	D17	D49	
21	22	D18	D50	
71	72	D19	D51	
23	24	D20	D52	
73	74	D21	D53	
25	26	D22	D54	
75	76	D23	D55	
27	28	P2	P6	
77	78	D24	D56	
29	30	D25	D57	
79	80	D26	D58	
31	32	D27	D59	
81	82	D28	D60	
33	34	D29	D61	
83	84	D30	D62	
35	36	D31	D63	
85	86	P3	P7	
37	38	INTERCONNECT-A	Ground	(Cable-A INTERCONNECT Source to Destination
87	88	CLOCK	Ground	is on pin 37; Destination to Source is on pin 38.)
39	40	BURST	Ground	
89	90	PACKET	Ground	
41	42	READY	Ground	
91	92	Ground	INTERCONNECT-B	(Cable-B INTERCONNECT Source to Destination
43	44	Reserved SD1	Reserved SD3	is on pin 91; Destination to Source is on pin 92.)
93	94	Reserved SD2	Reserved SD4	
45	46	Reserved DS1	Reserved DS3	
95	96	Reserved DS2	Reserved DS4	
47	48	Ground	Ground	
97	98	Ground	Ground	
49	50	CONNECT	Ground	
99	100	REQUEST	Ground	



Dimension	mm	in
A1	66.60	2.622
A2	5.69	0.224
A3	2.54	0.100
A4	1.27	0.050
A5	62.23	2.450
A6	15°	15°
A7	1.04 R	0.041 R
A8	0.396 ± 0.010	0.0156 ± 0.0004
A9	0.23	0.009
A10	0.61 ± 0.03	0.024 ± 0.001
A11	0.23	0.009
A12	0.05	0.002
A13	4.90 ± 0.10	0.193 ± 0.004
A14	4.27 max.	0.168 max.
A15	78.23 ± 0.13	3.080 ± 0.005
A16	0.25 ± 0.13	0.010 ± 0.005
A17	3.73 ± 0.15	0.147 ± 0.006
A18	2.64 min.	0.104 min.

Figure 23 – Cable connector – tabs



Dimension	mm	in
B1	66.45	2.616
B2	5.54	0.218
B3	2.54	0.100
B4	1.27	0.050
B5	62.23	2.450
B6	15°	15°
B7	1.00 R	0.039 R
B8	0.61 ± 0.05	0.024 ± 0.002
B9	0.15	0.006
B10	0.86 ± 0.10	0.034 ± 0.004
B11	0.15	0.006
B12	0.05	0.002
B13	5.00 ± 0.13	0.197 ± 0.005
B14	1.65 max.	0.065 max.
B15	78.23 ± 0.13	3.080 ± 0.005
B16	3.99 ± 0.05	0.157 ± 0.002
B17	1.52 ± 0.13	0.060 ± 0.005
B18	3.86 min.	0.152 min.

Figure 24 – Bulkhead connector – receptacle

Annex A
(informative)

Waveform examples

A.1 Introduction

Figure A.1 shows typical waveforms for an HIPPI-PH transfer consisting of two packets; the first packet contains two bursts and the second packet contains one burst. Detailed diagrams of the major switching points are contained in figures A.2 through A.5. An index to the detail drawings is at the top of figure A.1.

There are some indeterminate timing intervals, which are noted as dotted lines in the CLOCK waveforms of the detail figures. These times may be one or multiple CLOCK periods long. Descriptions of each of these intervals are included.

The INTERCONNECT signal, not shown, is assumed to be asserted and true for all of the waveforms. When INTERCONNECT is false, either the cable is disconnected or the power is off at the other end of the HIPPI-PH, and none of the other signals are to be used.

The symbol (S) after a signal name means that the signal is driven from the Source end of the HIPPI-PH; (D) means that the signal is driven from the Destination end of the HIPPI-PH.

The notation "w254" on the DATA BUS means the 254th word of a burst; "w1" means the first word of a burst; "n-1" means the next to last word of a burst; and "n" means the last word of a burst.

A.2 Connection and start packet

Figure A.2 details the connection portion of the HIPPI-PH operation. The Source puts the I-Field on the DATA BUS and asserts REQUEST. (The I-Field may be used for nonspecified control functions.) The Destination responds with CONNECT if it is willing to make the connection.

A.2.1 Time T1

Time T1 depends on two cable propagation delays and on the time the Destination takes to respond. REQUEST must propagate from Source to Destination, and CONNECT must propagate from Destination to Source. The Source keeps the I-Field word stable on the DATA BUS until it sees CONNECT.

A.2.2 Time T2

Time T2 depends on when the Source signals that it wants to start a packet. The PACKET signal is a delimiter used to mark a group of bursts as a logical packet. PACKET may be asserted by the Source after it sees CONNECT in the true state. The PACKET signal is independent of the READY signal; it may come before or after receiving READY indications.

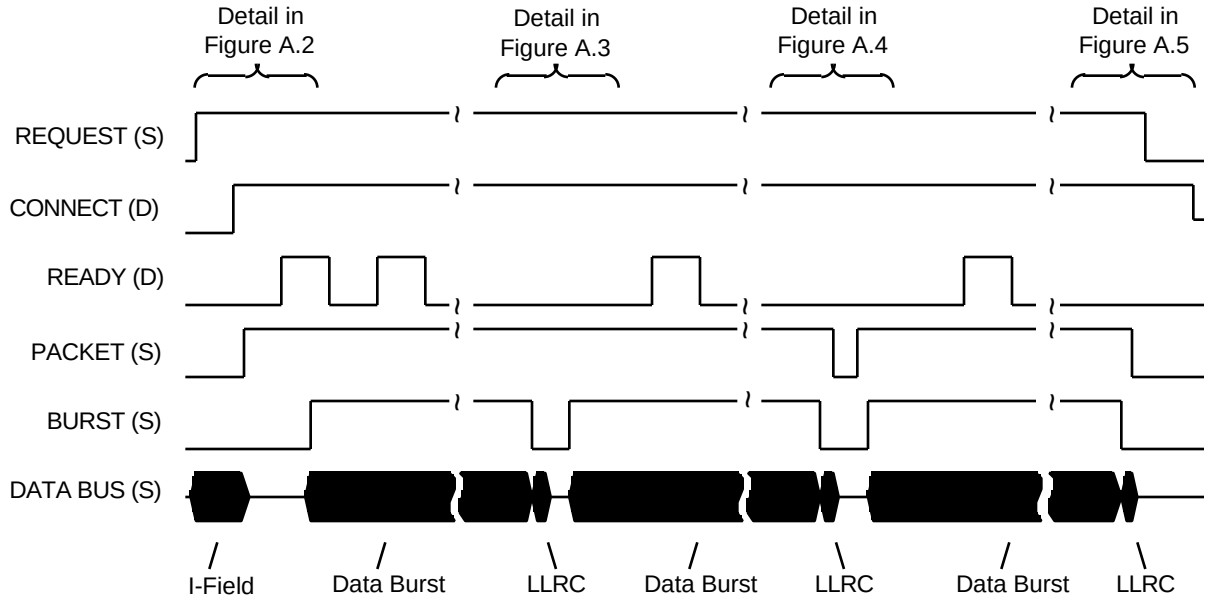


Figure A.1 – Typical HIPPI-PH waveforms

A.2.3 Time T3

Time T3 depends on when the Destination has a buffer available to receive a burst from the Source. After asserting CONNECT for four CLOCK periods, the Destination may begin sending READY indications to the Source. Each READY indication gives the Source permission to send a burst to the Destination. The Destination may send as many READY indications as it has buffers available to accept bursts. The Source should have a mechanism to keep track of the READY indications, at least up to a count of 63, counting up for each READY received and counting down each time a burst is sent.

Note that the CONNECT and READY signals must be asserted and negated for four-CLOCK-period minimums. The four-CLOCK minimum allows the Source to improve the noise immunity by sampling the signals for multiple CLOCK periods before treating them as valid. The CONNECT and READY signals are treated in this special way because no clocking accompanies them from the Destination.

A.2.4 Time T4

Time T4 depends on when the Source has a burst available to send and the fact that a READY indication has been received from the Destination for this burst.

The first word of the 256-word burst is asserted on the DATA BUS at the same time BURST is asserted. A new data word follows on each of the following 255 CLOCK periods, with no gaps or pauses in the burst. The CLOCK signal is a continuous 25 MHz signal (40 ns period) with no gaps or pauses.

To a first approximation the Source changes the data and control signals on the rising edge of CLOCK, and the Destination samples them on the falling edge of CLOCK.

A.3 End burst, start burst

Figure A.3 details the timing between bursts of the same packet. The BURST signal is deasserted after the last data word. Note that a Destination that treats the BURST signal as an extra data signal will see BURST true for each data word and false to indicate LLRC and gaps between bursts.

The LLRC (an even-parity longitudinal checkword) is a single word sent during the CLOCK period immediately after the burst.

A.3.1 Time T5

Time T5 depends on when the Source has another burst ready to transfer and whether the Destination has supplied another READY indication to accept the next burst.

Note that the Source must always insert at least two CLOCK periods after the LLRC before asserting BURST but that the Destination must operate correctly with only one CLOCK period after the LLRC. The difference is to allow optional intermediate repeaters to strip one of the CLOCK periods while synchronizing the signals to another clock.

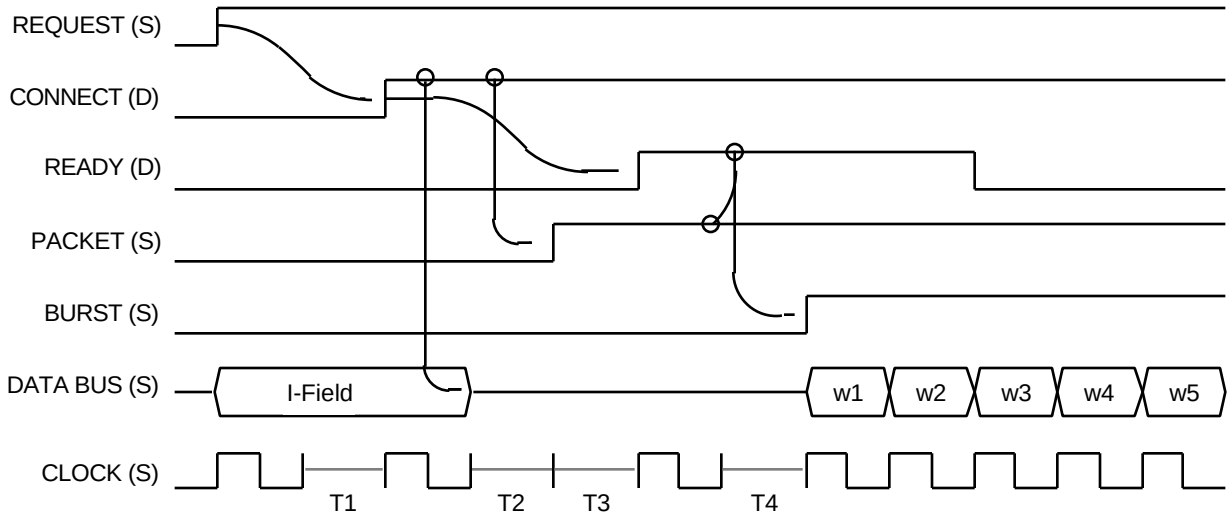


Figure A.2 – Connect, start packet, start burst

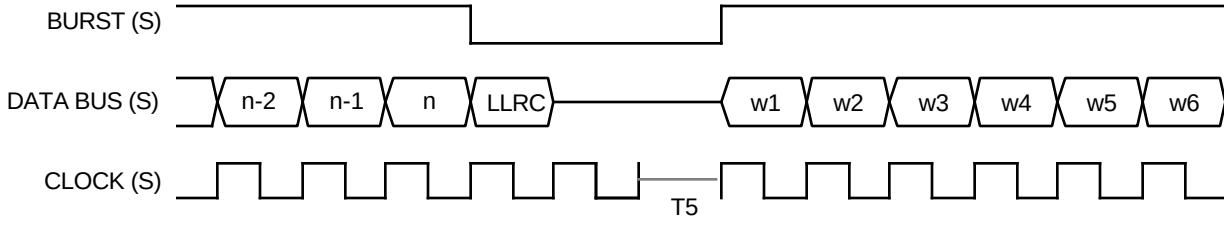


Figure A.3 – End burst, start burst

A.4 End burst, end packet, start packet, start burst

Figure A.4 details the timing between packets. The PACKET signal is deasserted one or more CLOCK periods after the deassertion of BURST.

A.4.1 Time T6

Time T6 depends on how long the Source takes to complete the packet after sending the last burst. Time T6 may be zero CLOCK periods.

A.4.2 Time T7

Time T7 is similar to Time T5. That is, the Source must deassert the PACKET signal for at least two CLOCK periods, but the Destination must operate correctly with the PACKET signal false for one CLOCK period. This is also to allow optional intermediate repeaters between the Source and Destination. Time T7 is also dependent on when the Source has another packet to send.

A.4.3 Time T8

Time T8 depends on when the Source has another burst ready to transfer and whether the Destination has supplied a READY indication to accept the next burst.

A.5 End burst, end packet, disconnect

Figure A.5 details the waveforms as seen at the Source when the last packet has been transferred and the connection between the Source and Destination is being torn down. The Source breaks the connection by deasserting the REQUEST signal. Upon seeing REQUEST false, the Destination responds by deasserting CONNECT.

A.5.1 Time T6

Time T6 is the same as in A.4, namely the time the Source takes to complete the last packet after sending the last burst.

A.5.2 Time T9

Time T9 depends on how long the Source takes to terminate the connection after completing the last packet. Time T9 may be zero CLOCK periods.

A.5.3 Time T10

Time T10 is similar to Time T1. That is, it depends on two cable propagation delays and on the time the Destination takes to respond.

Note that before another connection may be attempted, the Source must wait at least one cable propagation delay time and see CONNECT false. The READY counters in the Source and Destination are set to zero at the beginning of each connection.

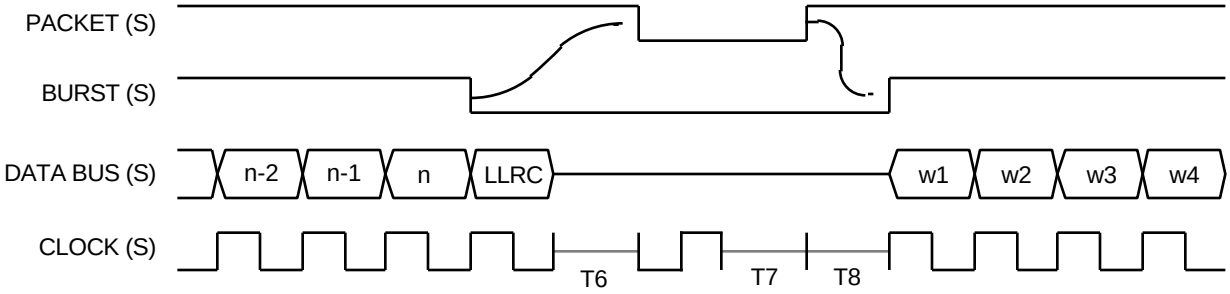


Figure A.4 – End burst, end packet, start packet, start burst

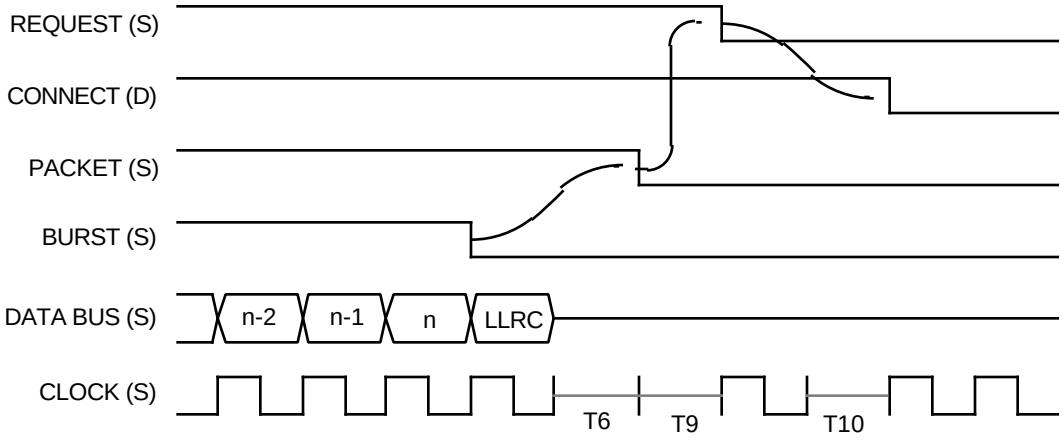


Figure A.5 – End burst, end packet, disconnect

A.6 Illegal end termination

Figure A.6 details the waveforms when the Destination decides to break the connection, possibly as the result of detecting an error.

The Destination breaks the connection by deasserting CONNECT any time the connection is valid. If the Destination was sending a READY indication, READY would be deasserted along with CONNECT.

Time T11

Time T11 is dependent on a cable propagation delay and on the time the Source takes to respond. The Source would deassert all of its asserted signals, whatever they may be at that time.

Note that it is not guaranteed that the Source will see this Illegal End Termination. If the Source was breaking the connection and deasserting REQUEST at the same time the Destination was deasserting CONNECT, then the Source would think that the connection was being broken as a result of its deasserting REQUEST.

A.7 Rejected connection sequence

Figure A.7 details the waveforms when the Destination decides that it will not honor the connect request of the Source but that it will abort the connection attempt early rather than force the Source to time out.

A.7.1 Time T1

Time T1 depends upon two cable propagation delays and how long the Destination takes to respond as in A.2. Note that in the Destination's response it does not send any READY indications to the Source.

A.7.2 Time T12

Time T12 is the time required for the Source to signal that it has a packet to send. In this sequence the Source may or may not assert the PACKET signal before the Destination deasserts CONNECT, depending on the relative speed of the two operations.

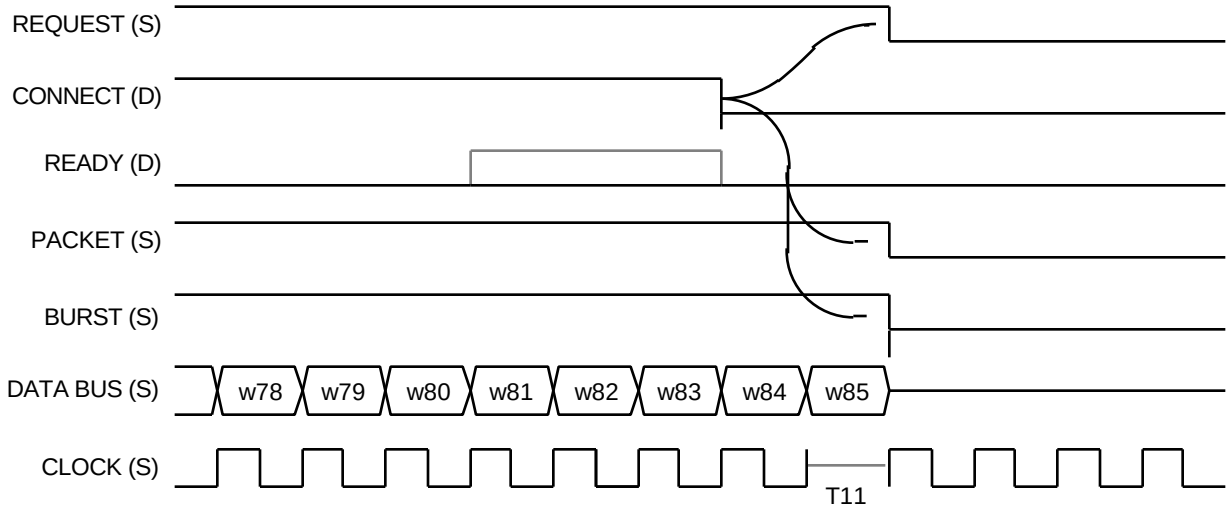


Figure A.6 – Illegal end termination

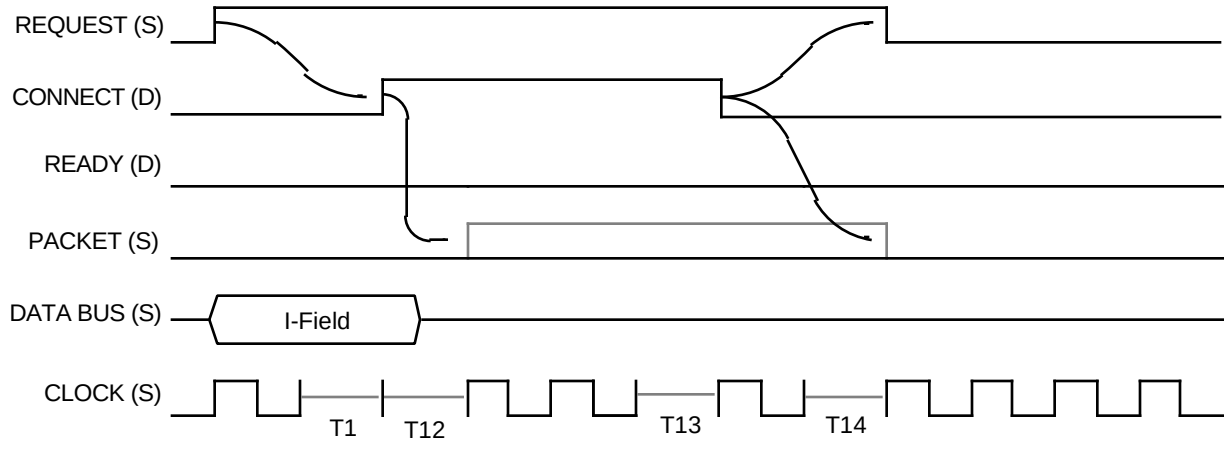


Figure A.7 – Rejected connection sequence

A.7.3 Time T13

Time T13 is the time required for the Destination to break the connection by dropping CONNECT. Time T13 is such that CONNECT is asserted for between 4 and 16 CLOCK periods.

A.7.4 Time T14

Time T14 depends on a cable propagation delay and on the time the Source takes to respond.

A.8 Aborted connection sequence

Figure A.8 details the waveforms when the Source decides for whatever reason to abort the connection request currently in process.

A.8.1 Time T15

Time T15 is the time that the Source waits before aborting the connection by deasserting the REQUEST signal. One reason for the abort may be that the Destination did not respond within the time that the Source expected a response.

A.8.2 Time T16

Time T16 is the time that the Source must wait before reasserting the REQUEST signal to start another connection sequence. As shown in figure A.8, the CONNECT signal may be seen as true after the REQUEST signal is deasserted due to the cable delays. Hence, Time T16 must be long enough to allow for the deassertion of REQUEST to propagate to the Destination, the Destination to respond by deasserting CONNECT (if it had been asserted), and for the deasserted CONNECT signal to propagate back to the Source.

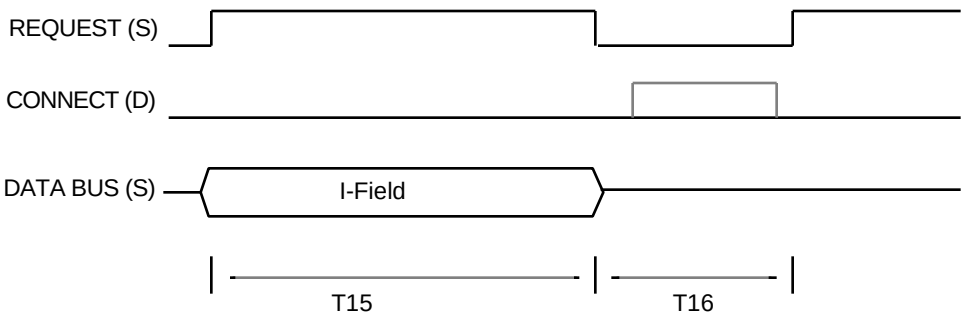


Figure A.8 – Aborted connection sequence

Annex B
(informative)

Implementation suggestions

B.1 Data rate option control

The HIPPI-PH supports data rate options of 800 Mbit/s and 1600 Mbit/s by using different widths on the DATA BUS and one or two cables. The data rate options are specified in 5.2. The ULP should be aware of the data rate option in use since it affects the burst size passed between the ULP and the HIPPI-PH. It is also feasible to build an HIPPI-PH that could be automatically configured to support either data rate option. Clause B.1 is intended as guidance to the implementor desiring to use the data rate options features.

B.1.1 Data rate detection

There is one INTERCONNECT signal when using the 800-Mbit/s data rate option; there are two INTERCONNECT signals when using the 1600-Mbit/s data rate option. These are described in 5.3.1.1 and 5.3.1.2. Circuitry may be used to differentiate between these two cases.

A disadvantage in using the INTERCONNECT signals for the data rate detection function is that the INTERCONNECT signals were originally intended for use in detecting a disconnected or broken cable, or when power was off at the remote end. Another function is now added to the signal, removing some of its uniqueness. However, Cable-A will always be present with either data rate option and can continue to function as the broken cable or power-down detection device. Cable-B present could then be used to differentiate between the 800-Mbit/s and 1600-Mbit/s data rate options.

B.1.2 Infrequent switching

If switching between the different data rates is an infrequent event, then the implementor may wish to add a parameter to the PHSM_STATUS.Confirm primitive defining the option in use. The ULP would then poll the HIPPI-PH to determine the option in use.

B.1.3 Frequent switching

If switching between the different data rates is more frequent, e.g., when connected to different rate interfaces through a crossbar switch, then the implementor may wish to add parameters to the PH_RING.Indicate and PH_ANSWER.Indicate primitives. This would provide the data rate option information on a per connection basis without the ULP having to poll the HIPPI-PH, but also implies that the ULP will have to check the data rate option on every connection.

B.2 Source READY counter

The Source is required in 5.3.6 to accept a minimum or 63 READY indications from the Destination, with each READY indication giving the Source permission to send another burst. A 64-word FIFO has been suggested as an efficient implementation of the S_READY_Ctr. As each READY indication is received, a word is strobed into the FIFO. The FIFO Output Ready signal is used as the S_READY_Ctr > 0 indicator. The counter is decremented by removing a word from the FIFO as the Source sends each burst.

B.2.1 READY indication overruns

If the Destination sends more READY indications than the FIFO can hold, then the excess will automatically be discarded. For example, assume that the Destination has buffering for 100 bursts and sends 100 READY indications, and the Source can only accept 64 READY indications. In this case, the Source will discard 36 READY indications. This is not a problem as long as the Destination sends additional READY indications whenever it frees burst buffers. Hence, on a physically long link there may be pauses while the READY indications and/or data are in transit.

B.2.2 Drops and pickups

If a READY indication is missed by the Source, then it is equivalent to ignoring one burst buffer in the Destination. A false READY indication may cause the Source to send a burst before the Destination is ready to accept it.

B.2.3 Continuous READY indications

There are also cases where the Destination may continually send READY indications with a free-running oscillator, e.g., when it knows that it can handle the bursts at the maximum rate. An example would be a graphics frame buffer that just updates its display with the data supplied.

B.3 I-Field sampling

To enhance the reliability of the I-Field, the Destination may optionally sample the I-Field multiple times and check for consistency before asserting the CONNECT signal.

B.4 Short bursts

Short bursts were included in the HIPPI-PH to (1) decrease the latency of short packets, and (2) to aid in transmitting packets whose sizes were not equal to an integral number of 256-word bursts. It is realized that the support of short bursts adds complexity. There are some Destinations that have no requirement for short bursts, e.g., some frame buffers. If a Destination that cannot handle short bursts receives a short burst, then it should break the connection by deasserting CONNECT.

B.5 Switching and the I-Field

The HIPPI-PH is a point-to-point interface, which with the use of the I-Field can control physical layer switches. For example, in figure B.1, the Source ULP could pass a Destination address to its HIPPI-PH via the CCI. The HIPPI-PH would in turn pass the CCI to the physical layer switch as the I-Field at connection time. The physical layer switch would use the I-Field to select the desired output port to the Destination.

Several physical layer switches in series could be accommodated, but some switches may modify the I-Field as individual switches are traversed. For example, they may move the address into the appropriate I-Field position for the next switch. Hence, the I-Field received by the Destination may be different from the I-Field supplied by the original Source.

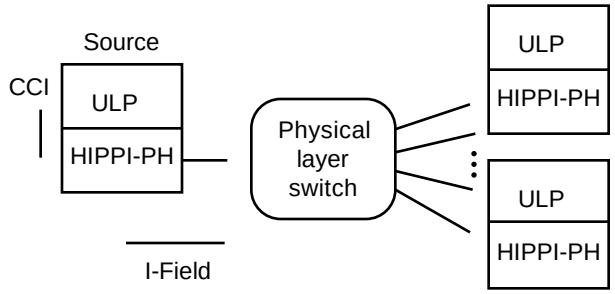


Figure B.1 – Physical layer switching

The original Source CCI (I-Field) is lost completely in other architectures which interpose intermediate nodes between the original Source and final Destination. This is illustrated in figure B.2 where the intermediate nodes use protocols above the physical layer. There is no mechanism specified to make the I-Field between HIPPI-PHs C and D the same as the I-Field between HIPPI-PHs A and B.

It may be useful for vendors to use the I-Field for other physical layer operations at connect time, e.g., reset the Destination. But since the I-Field might not be transmitted intact across physical layer switches, implementors are cautioned about using the I-Field for control of final Destinations.

To promote interoperability, it is recommended that low order I-Field bits be used for addressing, and if used, physical layer control be assigned to high order I-Field bits. Physical layer switches should preserve as many high order I-Field bits as possible that are not used for addressing.

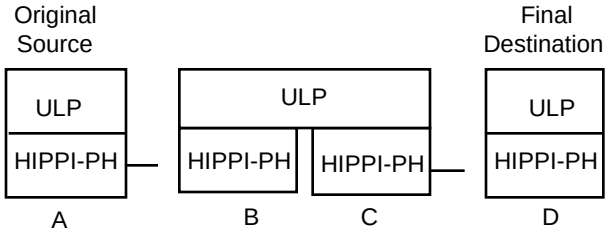


Figure B.2 – Switching with intermediate nodes

B.6 Byte ordering

The HIPPI Framing Protocol standard (HIPPI-FP) defines the mapping of an ordered byte stream to HIPPI-PH words. This byte ordering information is included here for reference. The byte positions within the HIPPI-PH words, for both the 32-bit and 64-bit HIPPI-PHs, are shown in figure B.3. Byte 0 is the first byte in the ordered byte stream, byte 1 is the second byte, etc.

Table B.1 – Byte assignments

Byte No.	Data signals on 800 Mbit/s HIPPI	Data signals on 1600 Mbit/s HIPPI
0	D31-D24	D31-D24
1	D23-D16	D23-D16
2	D15-D08	D15-D08
3	D07-D00	D07-D00
4	D31-D24	D63-D56
5	D23-D16	D55-D48
6	D15-D08	D47-D40
7	D07-D00	D39-D32

Figure B.3 shows the complete mapping of a 16-byte ordered byte stream on an 32-bit HIPPI-PH, and the same ordered byte stream on a 64-bit HIPPI-PH.

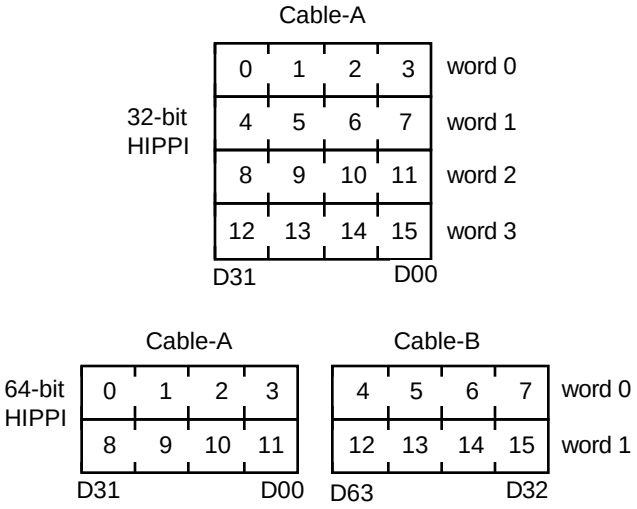


Figure B.3 – Ordered byte stream mapping

Within each byte on the HIPPI-PH cable the highest numbered signal is the most significant bit of the byte. An example for byte 0 is shown in figure B.4.

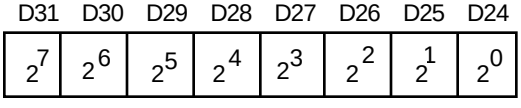


Figure B.4 – Bit significance within byte 0

Annex C
(informative)

Error checking

C.1 Byte parity

The PARITY BUS implements odd parity, as specified in 5.3.3, for each byte transmitted on the DATA BUS. Parity signal P0 is calculated from D00 to D07, P1 from D08 to D15, and Pn from D(8n) to D(8n+7). Figure C.1 shows parity calculated over a variety of bytes.

C.2 LLRC

A length/longitudinal redundancy checkword (LLRC), as specified in 5.1.5, is sent from the Source to the Destination on the DATA BUS during the first CLOCK period following each burst.

NOTE – The error detection mechanism provided by the LLRC and byte parity may be inadequate (by itself) for certain applications since it does not detect certain 4-bit, 6-bit, 8-bit, etc. errors within a burst.

C.2.1 LLRC count parameter

The count is the number of words in the burst, modulo 256. The LLRC count parameter includes the count in D00 to D07, and zeros in D08 to D31. When using the 1600-Mbit/s data rate option, the LLRC also includes the count in D32 to D39, and zeros in D40 to D63. The least significant bit of the count is in D00 and D32.

Modulo 256 means that the count uses only the low order 8 bits of the burst's word count. A length of 256 words results in an LLRC count parameter of all zeros.

C.2.2 LLRC Calculation

The LLRC is the longitudinal even parity function across all of the words in the burst and the LLRC count parameter. For example, D00 of the LLRC is the modulo 2 sum (exclusive-OR) of: (1) D00 of the first word of the burst, (2) D00 of the second word of the burst, etc., through (n) D00 of the last word of the burst, and (n+1) D00 of the LLRC count parameter. The LLRC includes correct byte parity as specified in 5.3.3 and C.1.

Figure C.1 shows the LLRC calculated for a 12-word short burst and a DATA BUS width of 16 bits (the actual DATA BUS width is 32 or 64 bits). The count of 12 words is used in the LLRC count parameter.

Implementors should note that it is feasible to exclusive-OR the count at either the beginning or the end of the burst. At the beginning of the burst may require less hardware in the Source HIPPI-PH, but also requires that the Source knows the length of the burst at the beginning of the burst. The Destination never knows ahead of time how long the burst will be, hence can only add the length at the end of the burst.

Note that the parity signals on the LLRC word are correct in the byte parity direction, and are not part of the LLRC calculation.

C.3 Burst Length Check

To check the correctness of the words in the burst and the length of each burst, the Destination computes an LLRC in the same way as the Source, and compares the result with the LLRC received from the Source. If the two are not equal, then either one or more words have been transferred in error, or have been picked up or dropped.

C.4 Sample LLRC circuit

A representative LLRC circuit is shown in figure C.2. The register and the count are both initialized to zero before the burst begins. Exclusive-OR #1 generates the modulo 2 sum of the words in the burst, accumulating the sum in the register. The count is incremented for each word of the burst. Exclusive-OR #2 adds in, modulo 2, the count portion of the LLRC.

C.4.1 Source LLRC details

At the Source, the count increments to 1 as the first modulo 2 sum is strobed into the register. The count is incremented and the register is strobed as each of the other words of the burst are passed to the Destination. At the end of the burst the LLRC n outputs are passed to the Destination as the LLRC.
An alternative circuit for the Source that may require fewer gates involves preloading the count value into the register before strobing the first burst word. Exclusive-OR #2 is not used in this alternative; LLRC 00 to 07 and 32 to 39 come straight from the register, like LLRC bits 08 to 31. This alternative requires that the length of the burst be known at the start of the burst, not necessarily true in all cases.

C.4.2 Destination LLRC details

At the Destination, the count and register operate the same as at the Source while the burst is being received. The count is incremented and the register is strobed as each word of the burst is received from the Source. When the LLRC is received, the register is strobed, but the count is not incremented (the BURST signal is false). After receiving the LLRC word all of the LLRC n output signals should be equal to zero. If they are not equal to zero then an error has been detected.

	Words in the burst												Count Parameter	LLRC
Data D15	0	1	0	1	0	0	1	1	1	0	0	1	0	**0**
Data D14	1	1	1	0	0	1	0	1	1	1	0	0	0	**1**
Data D13	1	0	1	1	1	1	0	1	1	0	1	1	0	**1**
Data D12	0	0	0	1	1	0	0	0	0	1	0	0	0	**1**
Data D11	1	1	0	0	0	1	0	1	0	0	0	0	0	**0**
Data D10	0	1	1	1	0	1	1	1	0	0	0	0	0	**0**
Data D09	0	0	0	0	0	1	1	1	1	1	1	1	0	**1**
Data D08	1	0	0	0	1	1	1	1	1	0	0	0	0	**0**
Parity 1	**1**	**1**	**0**	**1**	**0**	**1**	**1**	**0**	**0**	**0**	**1**	**0**		**1**
Data D07	1	0	0	0	1	0	1	0	0	1	1	0	0	**1**
Data D06	0	0	0	1	1	1	0	0	1	0	1	1	0	**0**
Data D05	0	0	0	1	0	1	1	1	1	0	1	0	0	**0**
Data D04	0	1	0	0	0	0	1	1	0	0	0	0	0	**1**
Data D03	1	0	1	1	1	0	0	0	1	0	1	0	1	**1**
Data D02	1	1	1	0	1	1	1	0	1	1	1	0	1	**0**
Data D01	0	0	0	0	0	0	0	0	1	1	1	1	0	**0**
Data D00	0	0	1	1	0	0	0	1	1	1	1	1	0	**1**
Parity 0	**0**	**1**	**0**	**1**	**1**	**0**	**1**	**0**	**1**	**1**	**0**	**0**		**1**

Figure C.1 – Parity and LLRC example

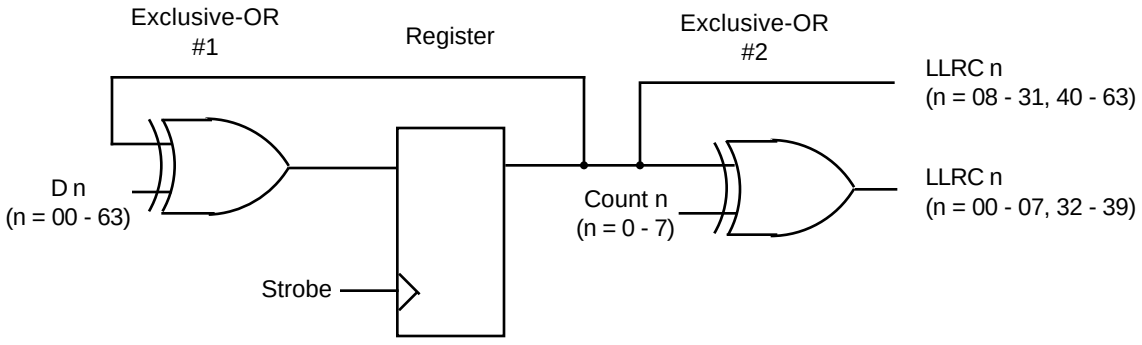


Figure C.2 – Representative LLRC circuit

Annex D
(informative)

Propagation delay calculation example

A block diagram of a theoretical implementation of the HIPPI-PH, including representative part numbers, is shown in figure D.1.

D.1 CLOCK

The timing reference is a 50-MHz local oscillator meeting the stability and accuracy requirements of 7.1. The double frequency oscillator is divided by 2 to ensure that the CLOCK symmetry requirements are met. The differential outputs of the left-hand divide-by-two flip-flop are used to drive the CLOCK cable line directly. This configuration asserts CLOCK at the connector early in a timing cycle as required to meet the hold time requirements of 7.3. These times are asymmetrical with respect to CLOCK to facilitate "tuning" at the Destination, if necessary, because delays may be added in the CLOCK line easier than in the data lines.

To avoid disturbing the loading of the CLOCK lines, a second flip-flop is used to divide the local oscillator to drive the internal Source logic. It is assumed in this model that the second half of the same dual flip-flop used to drive the CLOCK line will be used for this function and that the skew between the two flip-flops will be small compared to other delays.

D.2 Loading data

The rising edge of CLOCK is used to load data into the output register. The minimum and maximum relative delays for the data and CLOCK paths to the Source connector are shown on the figure. The 1.0 ns maximum trace and drop-cable skews shown just before the Source connector is a relative measurement of the maximum difference between the fastest and slowest trace and drop-cable pairs. If CLOCK is on the fastest pair, it is assigned a minimum delay of 0 ns. If it is on the slowest pair, it is assigned a maximum delay of 1.0 ns.

D.3 Cable skew

The maximum cable-induced skew of 8.0 ns is based on allowing 3.5 ns for delta propagation delay between pairs of a cable, 3.5 ns for inter-symbol distortion on a pair, and 1 ns for length differences between Cable-1 and Cable-2. A length difference of 10 cm may be reasonable if the cables are cut from the same reel.

D.4 Setup time

To meet the minimum Source setup time, Tss, of 10 ns with respect to the falling edge of CLOCK, the maximum delay for the data should not exceed the minimum delay for the CLOCK by more than Tss. Because this maximum is 8.8 ns, this design would have a margin of 1.2 ns to cover unknowns such as actual errors in our assumed perfect CLOCK.

D.5 Hold time

To meet the minimum Source hold time, Tsh, of 20 ns with respect to the falling edge of CLOCK, the data should not change at the connector before the rising edge of CLOCK. Because the minimum delay for the data is 2.5 ns, and the maximum CLOCK delay is 1.0 ns, this example would have a margin of 1.5 ns.

D.6 Tuning delay

The Destination should correctly capture data that meet a setup time, Tds, of 2 ns, and a hold time, Tdh, of 12 ns with respect to the falling edge of CLOCK at the Destination connector. The necessary tuning delay for our offset CLOCK can be calculated to meet the setup and hold times for the input register from the worst case delays through the logic.

The minimum tuning delay will cause the earliest possible CLOCK to arrive at the input register just as the latest possible data meet the register setup time requirements. The minimum tuning delay is 3.45 ns (maximum data delay) + 1.5 ns (input register setup time) - 2.0 ns (Tds) - 1.8 ns (minimum CLOCK delay), or 1.15 ns for the example.

The maximum tuning delay will cause the latest possible CLOCK to arrive at the input register just as the earliest data barely meet the input register hold time requirements. The maximum tuning delay is 12.0 ns (Tdh) + 0.6 ns (minimum data delay) - 1.0 ns (input register hold time) - 5.65 ns (maximum CLOCK delay), or 5.95 ns for the example.

Hence, this example should operate within specifications for a tuning delay between 1.15 ns and 5.95 ns. Operating margins will be improved by centering the delay as accurately as possible between these extremes.

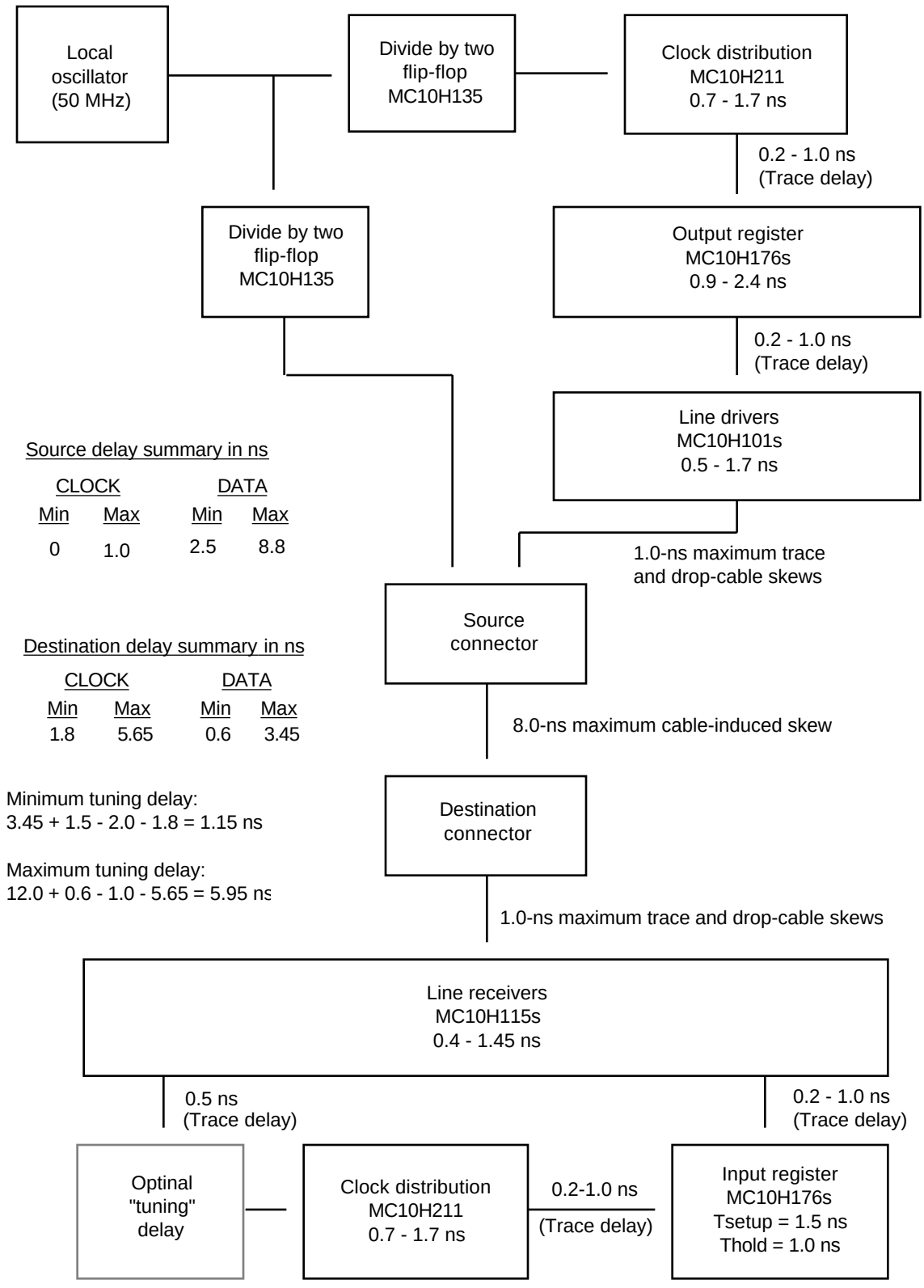


Figure D.1 – Propagation delay example block diagram

Annex E
(informative)

Component options

E.1 Cable availability and color coding

Madison cable, part number D0SD07BTIA is an example of a suitable product available commercially. This information is given for the convenience of users of this standard and does not constitute an endorsement by ANSI, or other publisher of this standard, of this product.

In the interest of uniformity when using this cable, the color coding and pin assignments shown in table E.1 are recommended.

E.2 Cable lengths

It would be advantageous if the industry could limit the cable length options to a reasonable number. The recommended cable lengths are either 15 m (+10 cm, -0 cm) [49.2 ft (+4 in, -0 in)] or 25 m (+10 cm, -0 cm) [82 ft (+4 in, -0 in)] measured at the connector mating interface. The overall cable length is measured at 18 to 24 °C with the cable in an unrestrained state (i.e., lying flat on a flat surface).

E.3 Connector alignment guide

The optional connector alignment guide detailed in figure E.1 is recommended for use with the bulkhead connector when the connector is mounted in the horizontal plane. This alignment guide provides additional support for the cable connector and cable.

E.4 Maximum connector footprint

The connectors, connector shells, and connector alignment guides should allow mounting multiple connectors on a 19 mm (0.75 in) by 95.25 mm (3.75 in) grid. Figure E.2 shows a maximum extent, with the outline box centered over the connector.

E.5 Connector availability

AMP Incorporated Amplimite 050 Series Connectors are examples of suitable products available commercially. This information is given for the convenience of users of this standard and does not constitute an endorsement by ANSI, or other publisher of this standard, of these products.

E.6 Connector jackscrew torque

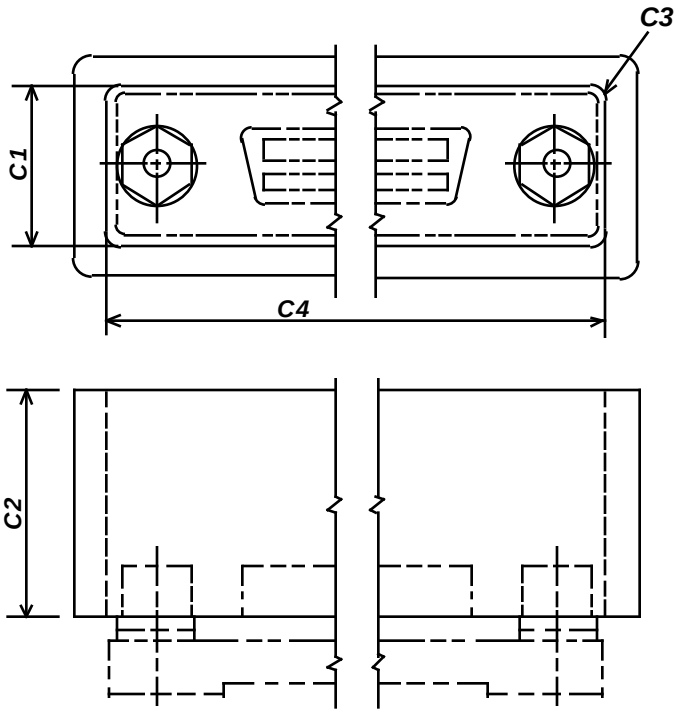
It is recommended that the connector jackscrews be constructed of high strength material capable of supporting a minimum of 1.2 N-m (11 in-lb) of torque.

E.7 Line driver and receiver availability

The emitter coupled logic (ECL) 10KH family of semiconductor components meet the electrical specifications of 8.1 and 8.2. This information is given for the convenience of users of this standard and does not constitute an endorsement by ANSI, or other publisher of this standard, of these products.

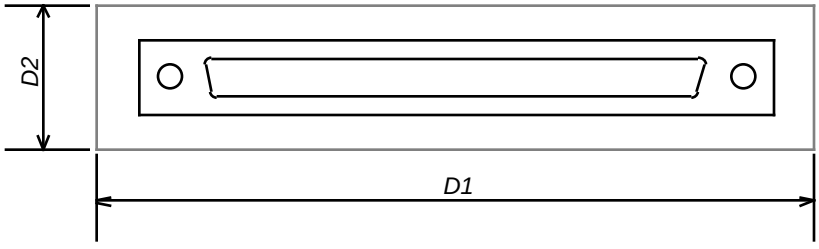
Table E.1 – Connector wire assignments

	Pin+		Pin-	
Pair #	Pin	Color	Pin	Color
1	1	White/Tan	2	Tan/White
2	3	White/Brown	4	Brown/White
3	5	White/Pink	6	Pink/White
4	7	White/Orange	8	Orange/White
5	9	White/Yellow	10	Yellow/White
6	11	White/Green	12	Green/White
7	13	White/Blue	14	Blue/White
8	15	White/Violet	16	Violet/White
9	17	White/Gray	18	Gray/White
10	19	Tan/Brown	20	Brown/Tan
11	21	Tan/Pink	22	Pink/Tan
12	23	Tan/Orange	24	Orange/Tan
13	25	Tan/Yellow	26	Yellow/Tan
14	27	Tan/Green	28	Green/Tan
15	29	Tan/Blue	30	Blue/Tan
16	31	Tan/Violet	32	Violet/Tan
17	33	Tan/Gray	34	Gray/Tan
18	35	Brown/Pink	36	Pink/Brown
19	37	Brown/Orange	38	Orange/Brown
20	39	Brown/Yellow	40	Yellow/Brown
21	41	Brown/Green	42	Green/Brown
22	43	Brown/Blue	44	Blue/Brown
23	45	Brown/Violet	46	Violet/Brown
24	47	Brown/Gray	48	Gray/Brown
25	49	Pink/Orange	50	Orange/Pink
26	51	Pink/Yellow	52	Yellow/Pink
27	53	Pink/Green	54	Green/Pink
28	55	Pink/Blue	56	Blue/Pink
29	57	Pink/Violet	58	Violet/Pink
30	59	Pink/Gray	60	Gray/Pink
31	61	Orange/Yellow	62	Yellow/Orange
32	63	Orange/Green	64	Green/Orange
33	65	Orange/Blue	66	Blue/Orange
34	67	Orange/Violet	68	Violet/Orange
35	69	Orange/Gray	70	Gray/Orange
36	71	Yellow/Green	72	Green/Yellow
37	73	Yellow/Blue	74	Blue/Yellow
38	75	Yellow/Violet	76	Violet/Yellow
39	77	Yellow/Gray	78	Gray/Yellow
40	79	Green/Blue	80	Blue/Green
41	81	Green/Violet	82	Violet/Green
42	83	Green/Gray	84	Gray/Green
43	85	Blue/Violet	86	Violet/Blue
44	87	Blue/Gray	88	Gray/Blue
45	89	Violet/Gray	90	Gray/Violet
46	91	White	92	Tan
47	93	Gray	94	Brown
48	95	Blue	96	Pink
49	97	Violet	98	Orange
50	99	Green	100	Yellow



Dimension	mm	in
C1	11.56	0.455
C2	16.61 ± 0.76	0.654 ± 0.030
C3	0.51 max. R	0.020 max. R
C4	84.99	3.346

Figure E.1 – Connector alignment guide



Dimension	mm	in
D1	92.25	3.75
D2	19	0.75

Figure E.2 – Maximum connector footprint

Alphabetical index