HIGH-PERFORMANCE PARALLEL INTERFACE - 6400 Mbit/s Physical Layer

(HIPPI-6400-PH)

March 4, 1996

Secretariat:

Information Technology Industry Council (ITI)

Open Issue - The Abstract is mostly copied from the Scope on page 1, and that text is open for discussion, i.e., the Abstract needs to be reviewed before it is final.

ABSTRACT: This standard specifies a physical-level, point-to-point, full-duplex, link interface for transmitting digital data at 6400 Mbit/s over parallel copper cables across distances of TBD m, or over parallel fiber-optic cables across distances of TBD m. Small fixed-size micro-packets provide an efficient, low-latency, structure for small messages, and a building block for large messages. Services are provided for transporting data streams specified by HIPPI-PH, ANSI X3.183-1991, which is limited to 25 m distances, and 800 or 1600 Mbit/s data rates.

NOTE:

This is an internal working document of X3T11, a Technical Committee of Accredited Standards Committee X3. As such, this is not a completed standard. The contents are actively being modified by X3T11. This document is made available for review and comment only. For current information on the status of this document contact the individuals 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-6400-PH Technical Editor)
Los Alamos National Laboratory
CIC-5, MS-B255
Los Alamos, NM 87545
(505) 667-5502, Fax: (505) 665-7793
E-mail: det@lanl.gov

Comments on Rev 0.15, March 4, 1996

This is a preliminary document undergoing lots of changes. Many of the additions are just place holders, or are put there to stimulate discussion. Hence, do not assume that the items herein are correct, or final – everything is subject to change. This page tries to outline where we are; what has been discussed and semi-approved, and what has been added or changed recently and deserves your special attention. This summary relates the changes since the previous revision, and is done in lieu of margin bars, which would be too numerous in this preliminary state of the document. Also, previous open issues are outlined with a single box, new ones are marked with a double bar on the left edge of the box.

Please help us in this development process by sending comments corrections, and suggestions to the Technical Editor, Don Tolmie, of the Los Alamos National Laboratory, at det@lanl.gov. If you would like to address the whole group working on this document, send the message to hippi-ext@think.com.

1. Figure 2 showing the micro-packet format was changed, and now encompasses what was in four previous figures.

2. The control bits were re-numbered in the reverse order.

3. The TSEQ number wrap was changed from x'FF' to x'FE' and value x'FF' is now used for RSEQ values to be ignored.

4. Some of the micro-packet types were changed.

5. Assignments were made for the data and control bits to specific signal lines. The time sequences for the bits on the signal lines was added. The ordering on the signal lines was changed from bit-wise to nibble-wise.

Contents

Page

Tables

Figures

Foreword (This Foreword is not part of American National Standard X3.xxx-199x.)

Open Issue - The first paragraph is a copy of the first paragraph of the Scope on page 1, and that text is open for discussion.

This American National Standard specifies a physical-level, point-to-point, full-duplex, link interface for transmitting digital data at 6400 Mbit/s over parallel copper cables across distances of TBD m, or over parallel fiber-optic cables across distances of TBD m. Small fixed-size micro-packets provide an efficient, low-latency, structure for small messages, and a building block for large messages. Services are provided for transporting data streams specified by HIPPI-PH, ANSI X3.183-1991, which is limited to 25 m distances, and 800 or 1600 Mbit/s data rates.

This standard provides an upward growth path for legacy HIPPI-based systems.

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, Information Technology Industry Council, 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 to be included in the published standard by the ANSI Editor.)

Subcommittee X3T11 on Device Level Interfaces, which developed this standard, had the following participants:

> (List of X3T11 Committee members, and other active participants, at the time the document is forwarded for public review, will be included by the Technical Editor.)

Introduction

This American National Standard specifies a physical-level, point-to-point, full-duplex, link interface for transmitting digital data at 6400 Mbit/s over parallel copper cables across distances of TBD m, or over parallel fiber-optic cables across distances of TBD m. Small fixed-size micro-packets provide an efficient, low-latency, structure for small messages, and a building block for large messages. Services are provided for transporting data streams specified by HIPPI-PH, ANSI X3.183-1991, which is limited to 25 m distances, and 800 or 1600 Mbit/s data rates.

Open Issue - Are characteristics correct ? Complete ?

Characteristics of a HIPPI-6400-PH physical-layer interface include:

– A data transfer bandwidth of 6400 Mbit/s (800 MByte/s).

– A full-duplex link capable of independent full-bandwidth transfers in both directions simultaneously.

– Four virtual circuits providing a limited multiplexing capability.

– A fixed size transfer unit, i.e., a micro-packet, for hardware efficiency.

– A small transfer unit resulting in low latency for short messages, and a building block for large messages.

– Credit-based flow control that prevents buffer overflow.

– End-to-end, as well as link-to-link, check sums.

– Automatic retransmission of errored data providing guaranteed, in-order, error free, data delivery.

– An ac coupled parallel electrical interface for limited distance applications, and a parallel fiber-optic interface for longer distances.

– Support for carrying legacy HIPPI-800 and HIPPI-1600 traffic.

American National Standard
for Information Technology –

High-Performance Parallel Interface – 6400 Mbit/s Physical Layer (HIPPI-6400-PH)

1 Scope

Open Issue - Agreement on scope, which is also used in other places in the front matter.

This American National Standard specifies a physical-level, point-to-point, full-duplex, link interface for transmitting digital data at 6400 Mbit/s over parallel copper cables across distances of TBD m, or over parallel fiber-optic cables across distances of TBD m. Small fixed-size micro-packets provide an efficient, low-latency, structure for small messages, and a building block for large messages. Services are provided for transporting data streams specified by HIPPI-PH, ANSI X3.183-1991, which is limited to 25 m distances, and 800 or 1600 Mbit/s data rates.

Specifications are included for:

– automatic retransmission of errored data;

– the format of a small data transfer unit called a micro-packet;

– a message structure that includes routing information for network applications;

– end-to-end, as well as link-to-link, check sums;

– the timing requirements of the parallel signals;

– a parallel interface using copper coaxial cable;

– a parallel interface using ribbon fiber-optic cable.

2 Normative references

The following American National Standard contains provisions which, through reference in this text, constitute provisions of this American National Standard. At the time of publication, the edition indicated was valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of the standard listed below.

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

ANSI X3.210-1992, *High-Performance Parallel interface, Framing Protocol (HIPPI-FP)* .

3 Definitions and conventions

3.1 Definitions

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

Open Issue - Agreement on definitions to date

3.1.1 bit error rate (BER): The statistical probability of a transmitted bit being erroneously received in a communication system. The BER is measured by counting the number of erroneous bits at the output of the receiver and dividing by the total number of bits.

3.1.2 credit: A credit corresponds to one micro-packet's worth of buffer space available in the Destination's VC buffer.

3.1.3 Destination: The equipment that receives the data.

3.1.4 HIPPI-PH: High-Performance Parallel Interface - Mechanical, Electrical, and Signalling Protocol Specification (HIPPI-PH), ANSI X3.183-1991. Data is transmitted in parallel over copper twisted-pair cables.

3.1.5 HIPPI port: A HIPPI-6400-PH, or HIPPI-PH, Source or Destination.

3.1.6 link: A full-duplex connection between HIPPI-6400-PH devices.

3.1.7 micro-packet: The basic transfer unit consisting of 32 data bytes and 64 bits of control information.

3.1.8 optional: Characteristics that are not required by HIPPI-6400-PH. However, if any optional characteristic is implemented, it shall be implemented as defined in HIPPI-6400-PH.

3.1.9 Source: The equipment that transmits the data.

3.1.10 virtual channel (VC): One of four paths within a link.

3.2 Editorial conventions

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

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

The word *shall* when used in this American National standard, states a mandatory rule or requirement. The word *should* when used in this standard, states a recommendation.

3.2.1 Binary notation

Binary notation is used to represent relatively short fields. For example a two-bit field containing the a binary value of 10 is shown in binary format as b'10'.

3.2.2 Hexadecimal notation

Hexadecimal notation is used to represent some fields. For example a two-byte field containing a binary value of 11000100 00000011 is shown in hexadecimal format as x'C403'.

3.3 Acronyms and other abbreviations

ACK	acknowledge
CR	credit amount parameter
CRC	cyclic redundancy check
ECRC	end-to-end CRC
HIPPI	High-Performance Parallel Interface
LCRC	link CRC
ns	nanoseconds
RSEQ	receive sequence number
TSEQ	transmit sequence number
VC	virtual channel
VCR	virtual channel CRedit selector
μs	microseconds
Ω	ohms

4 System overview

This clause provides an overview of the structure, concepts, and mechanisms used in HIPPI-6400-PH.

4.1 Links

HIPPI-6400-PH defines a point-to-point physical link for transferring micro-packets. The physical links, as shown in figure 1, are bi-directional. The logical links are simplex, i.e., the data inbound and outbound are completely separate. Some control information, e.g., credit, flows in the reverse direction, and it is included in the micro-packets flowing in the reverse direction. This is why the physical links must be bi-directional with information flowing in both directions simultaneously.

A link is composed of two Sources that transmit information, and two Destinations that receive information. Each end of a link has a Source and a Destination.

The data path is 16 bits wide for a copper implementation, and is 16 (or fewer) bits wide for a fiber implementation. The control path is one-fourth the width of the data path, e.g., the control path for a copper implementation would be 4 bits wide. CLOCK and FRAME are individual signals carried on separate conductors.

Open Issue – What do we call an interface with 16 data bits, a 16-bit wide interface after the number of data bit, or a 20-bit interface after the number of data and control bits, or a 22-bit interface after the total number of signal lines?

4.2 Virtual channels

Four virtual channels, VC0,VC1,VC2, and VC3 are available in each direction on each link. The VCs are assigned to specific message sizes and connection methods. All of the micro-packets of a message are transmitted on a single VC, i.e., the VC number does not change as the micro-packets travel from the original Source to the final Destination over one or more links. The VCs provide a multiplexing capability, preventing a large message from blocking a small high-priority message until the large message has completed.

Open Issue - Is the last sentence still true, i.e., what is the algorithm for transmitting from the different VCs?

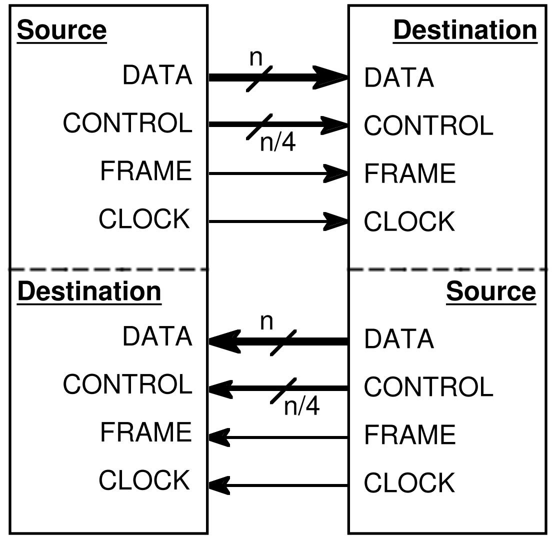


Figure 1 – HIPPI-6400-PH link showing signal lines

Open Issue – Should figure 1 show n and n/4 lines as the general case, or just use 16/8 and 4/2 since those are the widths we are specifying in detail?

4.3 Micro-packet

Micro-packets are the basic transfer unit. As shown in figure 2 a micro-packet is composed of 32 data bytes and 64 bits of control information. At 6400 Mbit/s a micro-packet is transmitted every 40 ns. Credit and retransmit operations are done on micro-packets.

The 64 bits of control information in each micro-packet includes parameters for:

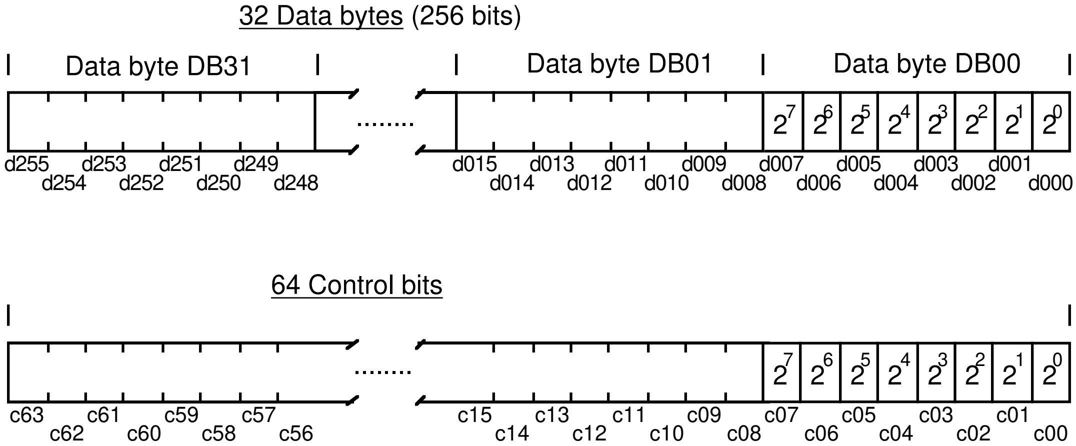
– selecting a VC;

– detecting missing micro-packets;

– denoting the types of information in the micro-packet;

– marking the last micro-packet of a message;

– signalling that the message was truncated at its originator, or damaged en-route, and should be discarded;



Naming conventions:

Data bytes are labeled capital DB and a two-digit number, e.g., DB00
Data bits are labeled lower case d and a three-digit number, e.g., d000
Control bits are labeled lower case c and a two-digit number, e.g., c00

Figure 2 – Micro-packet format and naming conventions

Open Issue – Figure 2 replaces figures 2, 3, 7, and 8 of rev 0.1. Is the new figure understandable, correct, and sufficient, or should any of the original figures be retained?

– passing credit information from the Destination to the Source;

– Link-level and end-to-end checksums.

4.4 Message

As shown in figure 4, messages are logical groups of micro-packets which have the same VC. The first micro-packet of a message, i.e., the Header micro-packet, contains information used to route through a HIPPI-6400 fabric. The last micro-packet of the message is marked with the TAIL bit.

Micro-packet Transmission order		
1	Routing information	c00–c63
2	1st 32 bytes of user data	c00–c63
3	2nd 32 bytes of user data	c00–c63
⋮	⋮	
n	Last bytes of user data	c00–c63

Figure 3 – Message format

4.5 FRAME and CLOCK signals

The FRAME signal, carried on a separate signal line, marks the beginning of micro-packets. Both edges of the CLOCK signal, also carried on a separate signal line, are used for strobing the data. The data, control, and FRAME signals from a Source are synchronous with that Source's CLOCK signal. The CLOCK rate is dependent on the width of the data bus, e.g., a 16-bit data bus utilizing 4b/5b encoding requires the CLOCK line to run at 500 MHz and each data and control line may transition every 2 ns.

4.6 Flow control

Credit-based flow control is used. As shown in figure 4 the credits are assigned on a VC basis, i.e., VC0's credits are separate from VC1's credits. The Destination end of a link grants credits to match the number of free receive buffers for a particular VC. The Source consumes credits as it moves micro-packets from the VC Buffers to the Output Buffer. Note that flow control is on a link basis, i.e., hop-by-hop.

4.7 Retransmission

Go-back-N retransmission is used. The link-level CRC of each micro-packet is checked at the Destination side of a link; at the Input Buffer in figure 4. Correct micro-packets are acknowledged, incorrect micro-packets are discarded. Retransmission of incorrect micro-packets is automatic. Note that retransmission is independent of the VC used, and also independent of the credit information, i.e., retransmission occurs between the Output and Input Buffers in figure 4 while VC and credit information pertains only to the VC Buffers. Retransmission is on a link basis, i.e., hop-by-hop.

Open Issue – Should both the LCRC and ECRC be checked at each node?

4.8 Check functions

As shown in table 1, two 16-bit cyclic redundancy checks (CRCs) are used, and they use different polynomials. The end-to-end CRC (ECRC) covers just the data portion of the micro-packets containing user data, and is unchanged from original Source to final Destination. The ECRC is accumulated over the entire message, i.e., it is not re-initialized at micro-packet boundaries. The ECRC does not cover the message Header micro-packet.

The link CRC (LCRC) covers all of the data and control bits of a micro-packet, with the exception of itself and the ECRC. The LCRC is initialized for each micro-packet, and must be calculated fresh for each link since other control fields change.

4.9 Copper physical layer (optional)

Open Issue - Length of coax interface.

Open Issue - Overall shield on coax cable?

Open Issue - AC coupling to coax, e.g., transformer, capacitor

Open Issue – Is micro-coax used, or micro-twinax?

Open Issue – Is it a single cable for full-duplex, or two cables?

The optional HIPPI-6400-PH copper variant uses a cable with 44 micro-coax conductors, 22 in each direction, and an overall shield. The length is limited to TBD. The signals are transformer coupled to the micro-coax to accomodate some difference in the ground potential between the equipment.

4.10 Fiber physical layer (optional)

Open Issue - Width of fiber interface, e.g., 16+4+1+1, or 8+2+1+1?

Open Issue - Length of fiber interface.

The optional HIPPI-6400-PH fiber variant uses a fiber-ribbon cable with 12 multi-mode fibers. The length is limited to TBD.

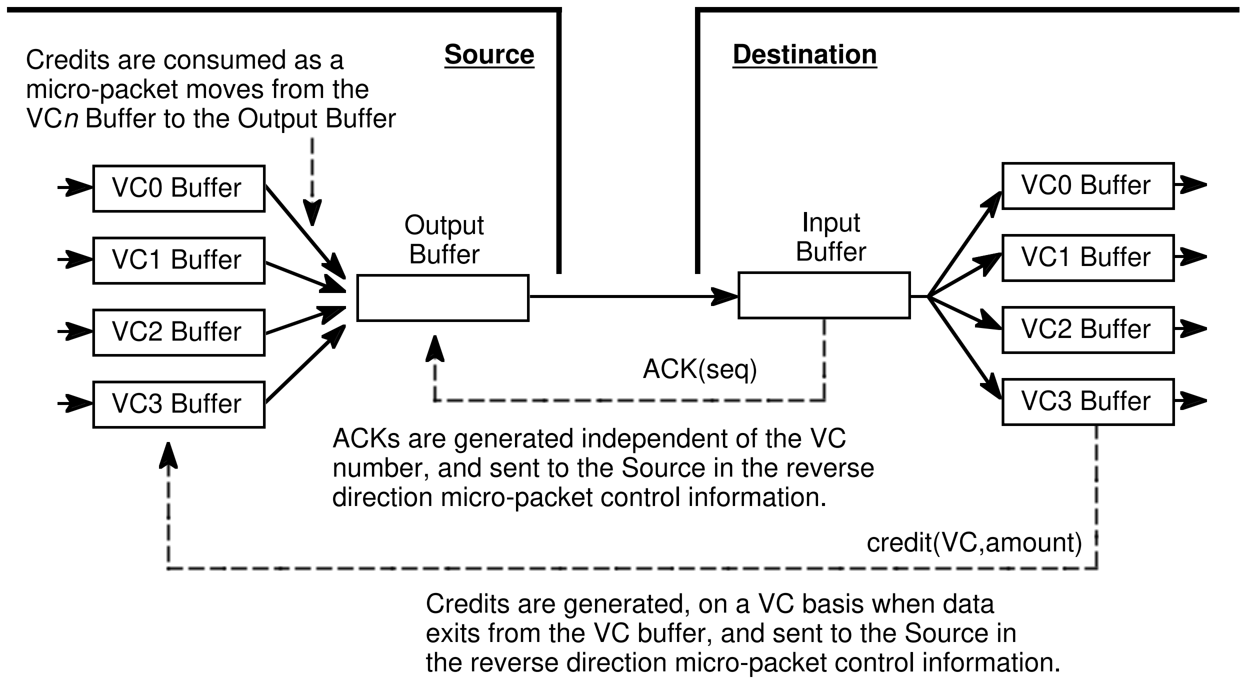


Figure 4 – Reverse direction control information

Table 1 – CRC coverage's in a 128-byte message

Micro-packet number	Data Bytes DB00 – DB31 contents	ECRC coverage	LCRC coverage
1	Header	Header	Header, c00 – c31
2	Bytes 1 – 32	Bytes 1 – 32	Bytes 1 – 32, c00 – c31
3	Bytes 33 – 64	Bytes 1 – 64	Bytes 33 – 64, c00 – c31
4	Bytes 65 – 96	Bytes 1 – 96	Bytes 65 – 96, c00 – c31
5	Bytes 97 – 128	Bytes 1 – 128	Bytes 97 – 128, c00 – c31

Open Issue - Is the ECRC value in the first micro-packet (i.e., the Header micro-packet) "don't care"? Zeros?

5 Service interface

Open Issue - Is there any reason not to include a service interface

Open Issue - Should the service interface be in the body of the document, in clause 4, in another clause, or in a normative annex?

This clause specifies the services provided by HIPPI-6400-PH. The intent is to allow ULPs to operate correctly with this HIPPI-6400-PH. How many of the services described herein are chosen for a given implementation is up to that implementor; however, a set of HIPPI-6400-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 5 shows the relationship of the HIPPI-6400-PH interfaces.

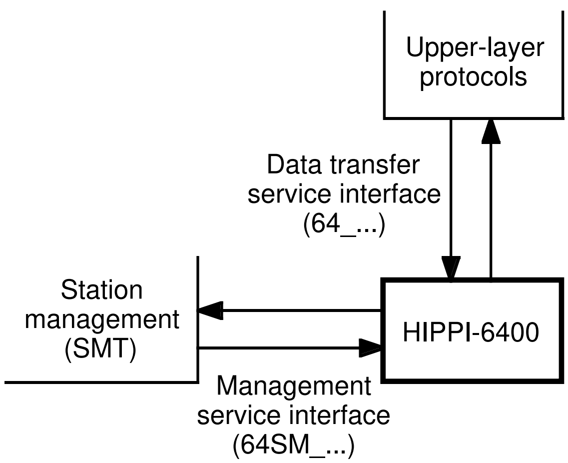


Figure 5 – HIPPI-6400-PH service interface

5.1 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-6400-PH service primitives are of four types.

– *Request primitives* are issued by a service user to initiate a service provided by the HIPPI-6400-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-6400-PH to acknowledge a Request.

– *Indicate primitives* are issued by the HIPPI-6400-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.

5.2 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-6400-PH users are depicted on either side of the vertical bars while the HIPPI-6400-PH acts as the service provider.

NOTE - The intent is to flesh out the service primitives similar to what is in HIPPI-PH today.

Open Issue - Do we need primitives to inform the SMT of the skew alignment status?

6 Micro-packet contents

6.1 Bit and byte assignments

As shown in figure 2, each micro-packet shall consist of 32 data bytes and 64 bits of control information. The data bytes shall be numbered DB00 - DB31. DB00 shall be transmitted first. The data bits in the micro-packet shall be numbered d000 – d255. d000 shall be transmitted first.

The 64 bits of control information shall be numbered as bits c00 – c63. Control bit c00 shall be transmitted first. As shown in figure 2, a field with a numerical value shall have its high-order bit in the highest numbered bit position.

The control information shall contain the following parameters located in the bits specified:

VC (2 bits, c00-c01) – The virtual channel selector. (See 6.2.)

TYPE (4 bits, c02-c05) – Identifies the type of information within the micro-packet. (See 6.3.)

TAIL (1 bit, c06) – TAIL = 1 identifies the last micro-packet of a message. TAIL = 0 means that more micro-packets for this message follow.

ABORT (1 bit, c07) – ABORT = 1 means that the message has been abnormally terminated up-stream, i.e., discard the message. ABORT = 0 means that the message is OK so far.

TSEQ (8 bits, c08-c15) – Sequence number of transmitted micro-packet. (See 6.4.)

RSEQ (8 bits, c16-c23) – Sequence number associated with micro-packet ACK. (See 6.4.)

VCR (2 bits, c24-c25) – Virtual channel number associated with credit addition. (See 6.5.)

CR (6 bits, c26-c31) – Amount of credit to add to the virtual channel specified in VCR. (See 6.5.)

ECRC (16 bits, c32-c47) – End-to-end checksum covering only the data bytes in this message. (See 6.6.) In all other micro-packet ECRC is unused and may be any value.

> *Open Issue – Should the ECRC be in the control bits, or in the data path?*

LCRC (16 bits, c48-c63) – Link level checksum covering the 32 data bytes, and the c63 through c32 control bits, in this micro-packet. (See 6.6.)

6.2 Virtual channel (VC) selector

Four virtual channels shall be available in each direction on a link. Messages on the virtual channels shall be assigned as follows:

– VC0 = Connectionless messages with a maximum size of 64 data micro-packets (2048 bytes) plus one Header micro-packet (32 bytes).

– VC1 = Connectionless messages with a maximum size of 4096 data micro-packets (128 KBytes) plus one Header micro-packet (32 bytes).

– VC2 = Connectionless messages with a maximum size of 4096 data micro-packets (128 KBytes) plus one Header micro-packet (32 bytes).

– VC3 = Connection-oriented messages of unlimited size. (See xxxx.) Each message contains a Header micro-packet as the first micro-packet of the message

> *Open Issue – Should we re-number the VCs so that we can use adaptation across the two medium sized VCs? This would have VC0 and VC1 = connectionless 128 KBytes, VC2 = connectionless 2 KBytes, and VC3 = same as now.*

6.3 TYPE parameter

The 4-bit TYPE parameter shall indicate the contents of the micro-packet. The values for TYPE shall be:

Open Issue - The different types of micro-packets listed below are just a wild guess by Don Tolmie, and have not been reviewed by anyone, i.e., don't put too much faith in them. They were put in so that we will have something on paper that we can use to start the discussion.

xxxx = Data micro-packet. The Data Bytes shall carry user information. All of the control information bits shall have their normal meanings.

xxxx = Header micro-packet. The Data Bytes shall carry routing information. The VC value shall be the same as the data micro-packets in this message. The control bits shall have their normal meanings. (See xxxx.)

Open Issue - Is the Header micro-packet a separate type, or should it be just the first Data micro-packet?

xxxx = Idle micro-packet. Idle micro-packets are used to transfer ACKs and credit information when there is no data to transfer. (See xxx.)

xxxx = Null micro-packet. A Null micro-packet is a gap filler, transmitted when there is no data, ACK or credit information, to transfer. (See xxx.)

Open Issue – If there is only an ACK to transfer, do we need an Idle micro-packet, or is a Null micro-packet sufficient?

Open Issue – Do we need a special type for broadcast/multicast to help switches support ARP?

Open Issue – What are the values for the TYPE parameter?

Open Issue – This table is highly speculative, and needs a detailed review.

Table 2 – Micro-packet contents summary

	Data micro-packet	**Header micro-packet**	**Idle micro-packet**	**Null micro-packet**	**?? micro-packet**
Data Bytes contents	32 bytes of user data	Routing information	all zeros	any	
VC	any	any	VC0	any	
TYPE value	????	????	????	????	
TAIL	= 1 on last micro-packet	0	0	0	
ABORT	= 1 if aborted	0	0	0	
TSEQ	increments	increments	increments	don't care	
RSEQ	ACK	ACK	ACK	ACK ??	
VCR	any	any	any	don't care	
CR	any	any	any	don't care	
ECRC value	accumulating	initialized	initialized	initialized ??	
LCRC	normal	normal	normal	normal	

Open Issue – What generic name can we call these micro-packets to differentiate them from things like the re-training micro-packets that use a special coding on the FRAME signal?

6.4 Sequence number parameters

The transmit sequence number (TSEQ) shall increment by one for each Data, Header, and Idle micro-packet transmitted, and shall wrap from x'FE' to x'00'. The receive sequence number (RSEQ) shall be used to acknowledge (ACK) these micro-packets. RSEQ shall equal the TSEQ of the micro-packet being acknowledged. RSEQ = x'FF' indicates that no ACK is being transmitted.

NOTES

1 The TSEQ and RSEQ parameters are independent of the virtual channel used to transmit the micro-packet.

2 The TSEQ and RSEQ parameters are local to a specific link. For example, a micro-packet that transverses more than one link will most likely have different TSEQ numbers on the different links.

The wrap at x'FE' shall be taken into account when processing ACKs. For example, if the pervious ACK had RSEQ = x'F7', and an ACK with RSEQ = x'03' is received, then the micro-packets whose TSEQ value = x'F8' through x'FE', and from x'00' through x'03', are acknowledged and their memory may be reused by the Source.

6.5 Credit update parameters

The Destination shall insert the VCR and CR parameters in micro-packets to inform the Source that CR number of micro-packet buffers have been freed up for the VC indicated by VCR. This gives the Source permission to transmit CR number of micro-packets on this VC. The Source shall increase its Credit Counter for this virtual channel by the value in CR. The Source Credit Counter shall be able to record up to 255 credits. A credit update that would exceed 255 shall result in an error indication. (See xxx.)

NOTES

1 The CR value is an incremental update value, not the number of buffers currently available in the Destination.

2 At 40 ns per micro-packet, and 5 ns per meter of cable, each credit is equivalent to about 8 meters of cable. Hence, a Credit Count, and Destination buffer capacity per virtual channel, of 255 will support full bandwidth on a 1 km link when round trip time is taken into account and Destination latency is low.

6.6 Check functions

Open Issue - What is the polynomial for the end-to-end CRC, and what is it initialized to.

Open Issue - Is the Header micro-packet included in the ECRC?

Two cyclic redundancy checks (CRCs) are used. The end-to-end CRC (ECRC) shall include only the data byte portion of the micro-packets in its calculation. The ECRC calculation shall be initialized to all ones at the start of a message, and shall include all of the data bytes in all of the micro-packets of the message. The ECRC polynomial shall be:

(include CRC-16 polynomial)

NOTE - The intent is to use the CRC specification in the FDDI document as a template for specifying the polynomial.

Open Issue - What is the polynomial for the link CRC, and what is it initialized to?

Open Issue - What is the order that the data bytes and control information is fed to the LCRC calculation? What I have included is just a guess and a place-holder.

The link CRC (LCRC) shall cover all of the data bits, and the control bits except for the ECRC and LCRC. The LCRC shall be initialized to all ones for each micro-packet. The LCRC calculation shall include the data bytes in the order shown in table 3, based on 16-bit quantities as inputs to the calculation.

Open Issue – The following table is a place holder only, don't depend on any of its information.

Table 3 – LCRC data ordering

Order	High-order Byte	Low-order Byte
1	DB0	DB1
2	DB2	DB3
3	DB4	DB5
4	DB6	DB7
5	c63–c54	c53–c48
6	DB8	DB9
7	DB10	DB11
8	DB12	DB13
9	DB14	DB15
10	c47–c40	c39–c32
11	DB16	DB17
12	DB18	DB19
13	DB20	DB21
14	DB22	DB23
15	c31–c24	c23–c16
16	DB24	DB25
17	DB26	DB27
18	DB28	DB29
19	DB30	DB31

The ECRC polynomial shall be:

6.7 Idle micro-packets

Idle micro-packets are used to fill in the gaps when there is no data to send, but there is credit update to send. Idle micro-packets shall include the incrementing TSEQ parameter.

Open Issue – Is the above text necessary when the Idle micro-packet is already defined in 6.3 and table 2?

6.8 Null micro-packets

Null micro-packets shall be inserted by the Source when no other micro-packets are available for transmission, and shall be inserted periodically to provide a chance to compensate for CLOCK rate differences between the Source and Destination.

Open Issue - Null micro-packets may be needed to account for clock drift and clock rate differences. How large a difference do we need to accomodate?

Open Issue - What is the maximum time between Null micro-packets?

6.9 Skew Adjustment micro-packets

Open Issue – Can the skew adjustment signals be generated with the 4b/5b coding or do they need special patterns that cannot be generated with the 4b/5b coding?

Open Issue – If the skew adjustment uses the normal 4b/5b coding then can we use the Null micro-packet?

Open Issue – Should the skew adjustment micro-packet be identified by a special pattern of the FRAME signal? If so, what is it?

Open Issue – What is the maximum time between Skew Adjustment micro-packets?

Open Issue – Is there a maximum rate or, or minimum time between, Skew Adjustment micro-packets?

Open Issue – If we need special patterns outside the 4b/5b coding for skew adjustment, then we probably also need the same, or similar, patterns during boot-up. Does a Boot-up micro-packet signal that a Boot-up is coming? Do we need a Boot-up micro-packet TYPE if we identify the function with a special pattern on the FRAME signal?

6.10 Other control micro-packets

Open Issue – Are there other control micro-packets, e.g., boot-up? reset? that should be included? Is this the appropriate place to do it?

7 Source micro-packet operations

Some of the parameters in the micro-packets are specific to the message, while other parameters are independent of the message. For example, the virtual channel number is message specific, and TSEQ is message independent.

7.1 Header micro-packet

Open Issue – Is the name "Header micro-packet" appropriate? Is there a better name, e.g., Routing micro-packet?

The first micro-packet of a message is called the Header micro-packet. It shall contain information to control routing through a HIPPI-6400-PH fabric, and a message identifier. The format of the Header micro-packet shall be:

ROUTE (4 bytes, DB00 - DB03) - ??

Open Issue - What information is contained in the ROUTE parameter? How big is the parameter? Is it the same as the I-Field? Is it defined in this document? In another document like HIPPI-SC?

MSG-id (4 bytes, DB04 - DB07) - ??

Open Issue - What is the format and content of the MSG-id? Any restrictions on its meaning, e.g., from a certain host, etc.?

AGE (2 bytes, DB08 – DB09) ??

Open Issue – How big is the AGE parameter, what is its function?

VC3 Header (8 bytes, DB10 – DB17) ??

Open Issue – How big is the VC3 Header parameter, and what is its function?

Open Issue – Is the best name for this parameter?

Open Issue – Are zeros used if a VC3 Header is not present? Is it indicated by a single bit somewhere?

Allow Alternate VC (bit xx)

Open Issue – If we can use VC1 as an alternate path to VC0, then this bit may be needed to allow that to happen.

DB18 through DB31 are reserved and shall be transmitted as zeros.

Open Issue - Are there other parameters in the Header micro-packet, if so what are they?

Open Issue – What about the control bits? TAIL, ABORT, ECRC, etc.

Open Issue – Can a Header micro-packet carry some user data?

7.2 Message specific control parameters

All of the micro-packets for a message shall be transmitted with the following parameters in the control bits of each micro-packet.

Open Issue - Does the HIPPI-6400-PH physical layer assign the VC based on message size, or does the ULP assign the VC value?

VC = value based on message size. (See 6.2.)

TAIL = 1 for last micro-packet in the message. TAIL = 0 for first and other micro-packets.

ABORT = 0, unless set by the ULP during the transfer of data from the ULP to HIPPI-6400-PH. After sending a micro-packet with ABORT = 1, all following micro-packets of this message shall discard without transmission .

Open Issue - Are there other reasons that HIPPI-6400-PH would set the ABORT bit?

Open Issue - Should HIPPI-6400-PH be responsible for discarding micro-packets after receiving an ABORT bit? Will it see any more micro-packets?

ECRC = ECRC calculation for the data bytes in this micro-packet, and in the previous micro-packets of this message. Note that this is a cumulative calculation, with the ECRC value different in each micro-packet, i.e., as the data bytes of each micro-packet are included. (See 6.6.)

7.3 Message independent control parameters

The following control parameters are independent of the message being sent, i.e., they operate across all of the virtual channels, and whether there is a message to be sent or not.

TYPE = value appropriate for the data and control information contained in the particular micro-packet. (See 6.3.)

TSEQ = increment by one for each micro-packet transmitted. TSEQ shall not advance beyond what has been acknowledged, i.e., it shall not over-write unacknowledged micro-packets. When retransmitting micro-packets, the original value of TSEQ shall be used.

Open Issue - What happens if TSEQ increments beyond what has been acknowledged? How do you keep this from happening? Is this an issue?

RSEQ = the TSEQ associated with the micro-packet being acknowledged. When retransmitting micro-packets, the original value of RSEQ shall be used.

VCR = Virtual channel number associated with the credit addition specified in the CR parameter. When retransmitting micro-packets, the original value of VCR shall be used.

CR = amount of credit to add to the virtual channel specified in VCR. When retransmitting micro-packets, the original value of CR shall be used.

NOTE - The VCR and CR parameters are supplied by the Destination-side of the link for delivery to the Source-side at the far end of the link.

LCRC = LCRC calculation for the data bytes and control bits in this micro-packet. (See 6.6.) When retransmitting micro-packets, the original value of LCRC shall be used.

7.4 Virtual channel priorities

The algorithm for choosing a micro-packet from the group of Source-side VC Buffers tries to keep a message flowing once started. That is, if the VC continues to have micro-packets to send, and has credit available, then that VC has priority over the other VCs. If the VC does not have a micro-packet to send, or does not have credit available, then the oldest micro-packet among the other VCs shall be sent next. If no micro-packets are available to send, or have credit, then an Idle micro-packet shall be sent on VC0.

Open Issue – Is this the correct algorithm? If so, how is age determined? How many bits in the AGE parameter? Is it programmable? How?

Open Issue – Should we sequence between the VCs on a rotary basis? If so, rotate every micro-packet? Every 256 micro-packets?

Open Issue – Should we give VC0 a higher priority?

Open Issue – If VC0 has a higher priority, then what do you go after servicing VC0?

7.5 Credit update indications on Source side

Credit update indications from the far end are received on the local Destination side, and passed to the local Source side, as shown in figure 4. A credit update shall increase the available credit, by the amount in the CR parameter, on the virtual channel whose number is the value in the VCR parameter.

7.6 ACK indications on Source side

ACK indications from the far end are received on the local Destination side, and passed to the local Source side, as shown in figure 4. An ACK indication acknowledges all of the transmitted micro-packets whose TSEQ ≤ RSEQ, i.e., the memory allocated to these micro-packets may be re-used. RSEQ = x'FF' shall be ignored.

Two consecutive ACKs with the same RSEQ value, and RSEQ ≠ x'FF', shall trigger a retransmit operation of all of the micro-packets in the Output Buffer (see figure 4) whose TSEQ is greater than or equal to the value in the duplicated RSEQ.

Open Issue – How many consecutive times should RSEQ = last good micro-packet be sent? Twice? Three times? Should RSEQ = x'FF' then be sent?

Open Issue – Is a time-out used to trigger a retransmit rather than a duplicate RSEQ? If so, how long is the timer?

Open Issue – Are there micro-packets with a higher priority than retransmitted micro-packets? Idles? Skew Adjustment? Null? If so, where do they get inserted in the data stream?

Open Issue – Is there a maximum number of times a retransmit will be tried?

7.7 ACKs and credit updates to far end

The local Destination side sends ACKs and credit update information to the far end by first queuing them to the local Source side, as shown in figure 4. The Source side shall transmit this information in micro-packets using the appropriate control bits.

Since the ACKs and credit update information do not share their fields with any other parameters they can be sent with every micro-packet. The algorithm for sending credit updates for the different virtual channels is to service the most needy VC first, i.e., the VC with the minimum number of outstanding credits.

Open Issue - Is this algorithm for sending credits correct?

The local Destination may queue multiple ACK RSEQ parameters before one is transmitted by the local Source end. The RSEQ parameter should be over-written so that the ACK message transmitted uses the latest value of RSEQ.

Open Issue – Is the above statement true? Is it necessary?

8 Destination micro-packet operations

8.1 Checking for errors

Upon receipt of a micro-packet, the Destination shall check the LCRC and ECRC for correctness. If correct, the micro-packet shall be delivered to the virtual channel buffer designated by the VC parameter.

Open Issue - Is the ECRC checked at each Destination end of a link, or only at the final Destination?

Open Issue - Is the ECRC checked on each micro-packet, or only on the one with TAIL = 1?

Open Issue - How does a device know that it is the last one in the chain and should check the ECRC?

Skipping a TSEQ number will result in an ECRC error. Losing a TAIL bit will result in concatenating two messages and an ECRC error.

Open Issue - Are these error conditions correct?

Open Issue - Are there other errors that are checked for?

8.2 Generating ACKs

If the LCRC is correct, the Destination shall queue, to the local Source side, an ACK with an RSEQ whose value equals the value of the TSEQ parameter in the received micro-packet. This acknowledges this micro-packet.

If the LCRC is incorrect, the Destination shall queue, to the local Source side, an ACK with RSEQ equal the TSEQ parameter of the last correctly received micro-packet. This requests retransmission of the flawed micro-packet, and all other micro-packets transmitted since this flawed micro-packet. The Destination shall discard all micro-packets until the originally flawed micro-packet is received correctly.

9 Initialization

Three levels of initialization, or reset, are specified, power-on or cold start, warm start, and link reset. They differ in the amount of hardware reset, amount of time until the link is available, and the amount of data lost.

Open Issue - What is the difference between cold and warm start?

9.1 Cold start

Cold start is triggered by Source power-on. A cold start shall ???

Open Issue - What are the results of a cold start?

A cold start operation shall be signaled to the far end by the local Source side toggling the FRAME signal at the same rate as the CLOCK signal, and in-phase with the CLOCK signal for one second. During this time the Destination side shall use the dynamic skew compensation to align the FRAME signal to the CLOCK signal.

Open Issue - Are there actions or resets that occur during this second?

Open Issue – Is one second the correct amount of time for the reset?

At the end of the second, if the local Destination side has received a cold start indication, i.e., the FRAME signal toggling at the same rate as the CLOCK signal, then the local Source side shall decrease the toggling rate of the FRAME signal to one-half the rate of the CLOCK signal.

Open Issue - What happens if the local Destination end has not received a cold start indication, i.e., FRAME toggling at CLOCK rate?

Open Issue - Are there any special actions that occur at this time?

Open Issue - How long do you stay in this state, and how do you get out?

Open Issue - When do you do the dynamic skew alignment on the data and control bits?

9.2 Warm start

Warm start is triggered by

Open Issue - What triggers a warm start?

A warm start shall

Open Issue - What are the results of a warm start?

9.3 Link Reset

Link Reset shall be triggered by ??.

Open Issue - What triggers a Link Reset?

A Link Reset shall

Open Issue - What are the results of a Link Reset?

10 Signal line encoding

10.1 Signal line bit assignments

The data bytes and control bits shall be transmitted on the signal lines specified in table 4 for a 16-bit wide interface, and as specified in table 5 for an 8-bit wide interface. Nomenclature for the data and control bits is detailed in figure 2. Data signal lines are labeled capital D and a two-digit number, e.g., D00. Control signal lines are labeled capital C and a one-digit number, e.g., C0

Table 4 – Signal line bit assignments in a 16-bit system

Signal lines																			
C3	C2	C1	C0	D 15	D 14	D 13	D 12	D 11	D 10	D 09	D 08	D 07	D 06	D 05	D 04	D 03	D 02	D 01	D 00
12	08	04	00	060	056	052	048	044	040	036	032	028	024	020	016	012	008	004	000
13	09	05	01	061	057	053	049	045	041	037	033	029	025	021	017	013	009	005	001
14	10	06	02	062	058	054	050	046	042	038	034	030	026	022	018	014	010	006	002
15	11	07	03	063	059	055	051	047	043	039	035	031	027	023	019	015	011	007	003
28	24	20	16	124	120	116	112	108	104	100	096	092	088	084	080	076	072	068	064
29	25	21	17	125	121	117	113	109	105	101	097	093	089	085	081	077	073	069	065
30	26	22	18	126	122	118	114	110	106	102	098	094	090	086	082	078	074	070	066
31	27	23	19	127	123	119	115	111	107	103	099	095	091	087	083	079	075	071	067
44	40	36	32	188	184	180	176	172	168	164	160	156	152	148	144	140	136	132	128
45	41	37	33	189	185	181	177	173	169	165	161	157	153	149	145	141	137	133	129
46	42	38	34	190	186	182	178	174	170	166	162	158	154	150	146	142	138	134	130
47	43	39	35	191	187	183	179	175	171	167	163	159	155	151	147	143	139	135	131
60	56	52	48	252	248	244	240	236	232	228	224	220	216	212	208	204	200	196	192
61	57	53	49	253	249	245	241	237	233	229	225	221	217	213	209	205	201	197	193
62	58	54	50	254	250	246	242	238	234	230	226	222	218	214	210	206	202	198	194
63	59	55	51	255	251	247	243	239	235	231	227	223	219	215	211	207	203	199	195

NOTES:

1 The two-digit numbers in the C*n* columns are the control bits, c*nn*.

2 The three-digit numbers in the D*nn* columns are the data bits, d*nnn*.

3 The bits in a column are transmitted on the associated signal line, top entry first, bottom entry last. For example in the left column for signal line C3, c12 is transmitted first, and c63 is transmitted last.

4 The four-bit groupings in a column denote 4-bit code groups for encoding/decoding to/from the 5-bit transmission codes specified in table 6.

Open Issue – Tables 4 and 5 need to be reviewed for correctness and clarity.

Table 5 – Signal line bit assignments in an 8-bit system

Signal lines									
C1	C0	D 07	D 06	D 05	D 04	D 03	D 02	D 01	D 00
08	00	056	048	040	032	024	016	008	000
09	01	057	049	041	033	025	017	009	001
10	02	058	050	042	034	026	018	010	002
11	03	059	051	043	035	027	019	011	003
12	04	060	052	044	036	028	020	012	004
13	05	061	053	045	037	029	021	013	005
14	06	062	054	046	038	030	022	014	006
15	07	063	055	047	039	031	023	015	007
24	16	120	112	104	096	088	080	072	064
25	17	121	113	105	097	089	081	073	065
26	18	122	114	106	098	090	082	074	066
27	19	123	115	107	099	091	083	075	067
28	20	124	116	108	100	092	084	076	068
29	21	125	117	109	101	093	085	077	069
30	22	126	118	110	102	094	085	078	070
31	23	127	119	111	103	095	087	079	071
40	32	184	176	168	160	152	144	136	128
41	33	185	177	169	161	153	145	137	129
42	34	186	178	170	162	154	146	138	130
43	35	187	179	171	163	155	147	139	131
44	36	188	180	172	164	156	148	140	132
45	37	189	181	173	165	157	149	141	133
46	38	190	182	174	166	158	150	142	134
47	39	191	183	175	167	159	151	143	135
56	48	248	240	232	224	216	208	200	192
57	49	249	241	233	225	217	209	201	193
58	50	250	242	234	226	218	210	202	194
59	51	251	243	235	227	219	211	203	195
60	52	252	244	236	228	220	213	204	196
61	53	253	245	237	229	221	213	205	197
62	54	254	246	238	230	222	214	206	197
63	55	255	247	239	231	223	215	207	199

NOTES:

1 The two-digit numbers in the C*n* columns are the control bits, c*nn*.

2 The three-digit numbers in the D*nn* columns are the data bits, d*nnn*.

3 The bits in a column are transmitted on the associated signal line, top entry first, bottom entry last. For example in the left column for signal line C1, c08 is transmitted first, and c63 is transmitted last.

4 The four-bit groupings in a column denote 4-bit code groups for encoding/decoding to/from the 5-bit transmission codes specified in table 6.

10.2 Source-side encoding for dc balance

Open Issue – This text needs to be reviewed for correctness and clarity.

The transmitted signals shall be encoded to achieve dc balance on each signal line. Table 6 specifies the 5-bit signal line codes corresponding to the 4-bit input codes from tables 4 and 5. For example, on signal line C3 in a 16-bit system, the first 4-bit code consists of bits c12, c13, c14, and c15. c15 is the most-significant bit, and c12 is the least-significant bit.

A running count, called the Disparity Count, shall be kept of all the ones and zeros transmitted since the link was reset. The Disparity Count shall be incremented for each one transmitted, and decremented for each zero transmitted.

Open Issue – Should the Disparity Count be reset on Skew Adjustments?

The appropriate 5-bit code value from table 6, based on the current value of the Disparity Count, shall be transmitted low-order bit first.

For example in a 16-bit system, if:
c12 = 1 (least-significant bit)
c13 = 0
c14 = 0
c15 = 0 (most-significant bit)
Disparity Count = +1 before encoding
then transmit on C3:
1
0
1
0
0
Disparity Count = 0 after encoding. This example corresponds to the first column in table 4, and the second row in table 6.

NOTES

1 The range for the Disparity Count is from +4 to -5.

2 The Disparity Count may also be updated by adding or subtracting the value of Delta Disparity shown in table 4. Add Delta Disparity if Disparity Count < 0; subtract if ≥ 0.

3 The 5-bit code is derived by inserting a 1 in the middle of the 4-bit code, and then transmitting either the true or complement value of the resultant 5-bit quantity.

4. The maximum run length, i.e., the longest string of continuous 1s or 0s, is 11. The string of 4-bit code points creating the maximum run length is x'efc'. Start with Disparity Count = +3 or +4 for a string of 11 zeros. Start with Disparity Count = -4 or -5 for a string of 11 ones.

Table 6 – Line coding

4-bit code	5-bit code when Disparity < 0	5-bit code when Disparity ≥ 0	Delta Disparity
0000	11011	00100	3
0001	11010	00101	1
0010	11001	00110	1
0011	00111	11000	1
0100	10011	01100	1
0101	01101	10010	1
0110	01110	10001	1
0111	01111	10000	3
1000	01011	10100	1
1001	10101	01010	1
1010	10110	01001	1
1011	10111	01000	3
1100	11100	00011	1
1101	11101	00010	3
1110	11110	00001	3
1111	11111	00000	5

The data and control signal lines shall be synchronized with the CLOCK and FRAME signals as shown in table 7. In the previous example for the C3 signal in a 16-bit system, the right most "xxTxx" pattern would be 00101, and the right-most 1 bit would be transmitted first. The "T" in the middle of the "xxTxx" symbol stands for the True/Complement bit inserted in the middle of a 4-bit code.

Table 7 – Encoded signal lines in a 16-bit system

	Transmitted last			Transmitted first
Data, Header, or Idle micro-packet				
CLOCK	01010	10101	01010	10101
FRAME	00000	00000	11111	11111
D*nn* or C*n*	xxTxx	xxTxx	xxTxx	xxTxx
Null micro-packet				
CLOCK	01010	10101	01010	10101
FRAME	00000	00000	11111	11111
D*nn* or C*n*	xxTxx	xxTxx	xxTxx	xxTxx
Skew Adjustment micro-packet				
CLOCK	01010	10101	01010	10101
FRAME	00000	00000	11111	11111
D*nn* or C*n*	xxTxx	xxTxx	xxTxx	xxTxx
Link Reset micro-packet				
CLOCK	01010	10101	01010	10101
FRAME	00000	00000	11111	11111
D*nn* or C*n*	xxTxx	xxTxx	xxTxx	xxTxx
?? micro-packet				
CLOCK	01010	10101	01010	10101
FRAME	00000	00000	11111	11111
D*nn* or C*n*	xxTxx	xxTxx	xxTxx	xxTxx

Open Issue – Is the CLOCK signal in the right phase, i.e., a "1" for the first data bit?

Open Issue – What are the patterns for the FRAME signal in the different types of micro-packets, what is here is just a place holder.

Open Issue – Is the Skew Adjustment micro-packet a true micro-packet occuring in 40 ns, or does it take longer, and hence should be called a Skew Adjustment Interval?

Open Issue – Is Skew Adjustment the right name, or should it be called "Re-Training"?

Open Issue – How many unique codings are there, i.e., some of the entries in table 7 may be eliminated.

Table 8 – Encoded signal lines in an 8-bit system

	Transmitted last							Transmitted first
Data, Header, or Idle micro-packet								
CLOCK	01010	10101	01010	10101	01010	10101	01010	10101
FRAME	00000	00000	00000	00000	11111	11111	11111	11111
D*nn* or C*n*	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx
Null micro-packet								
CLOCK	01010	10101	01010	10101	01010	10101	01010	10101
FRAME	00000	00000	00000	00000	11111	11111	11111	11111
D*nn* or C*n*	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx
Skew Adjustment micro-packet								
CLOCK	01010	10101	01010	10101	01010	10101	01010	10101
FRAME	00000	00000	00000	00000	11111	11111	11111	11111
D*nn* or C*n*	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx
Link Reset micro-packet								
CLOCK	01010	10101	01010	10101	01010	10101	01010	10101
FRAME	00000	00000	00000	00000	11111	11111	11111	11111
D*nn* or C*n*	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx
?? micro-packet								
CLOCK	01010	10101	01010	10101	01010	10101	01010	10101
FRAME	00000	00000	00000	00000	11111	11111	11111	11111
D*nn* or C*n*	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx	xxTxx

Open Issue – What are the patterns for the FRAME signal in the different types of micro-packets, what is here is just a place holder. This may be different from the coding for a 16-bit system.

Open Issue – How many unique codings are there, i.e., some of the entries in table 8 may be eliminated.

10.3 Destination-side decoding

The received signals shall each be decoded in groups of five bits according to table 6.

NOTES

1 The FRAME signal marks the beginning of a 5-bit code.

2 Decoding can be implemented by examining the middle bit of the 5-bit code; if 1 then use the outer bits uncomplemented, if 0 then complement before use.

3 There are no illegal 5-bit codes.

10.4 Logic levels

A logical 1 shall be transmitted on an electrical interface as the more positive electrical signal. A logical 1 shall be transmitted on an optical interface as greater light output power than for a logical 0.

Open Issue - Are these proposed logic levels OK?

10.5 Start of micro-packet

The start of a micro-packet shall be signaled by the FRAME signal going from logical 0 to 1.

Open Issue – The FRAME signal will probably need more exotic decoding, e.g., more of a pulse width decoding so that start of frame = 0 to 1 after being 0 for n bit times.

11 Maintenance and control features

Open Issue - What maintenance or control messages are possible?

Open Issue - Do we want to standardize all possible maintenance and control messages?

Open Issue - What is the relative priority of maintenance and control messages?

Open Issue - Should we have a MIB for a HIPPI-6400-PH link? For other HIPPI-6400-PH?

12 Dynamic skew compensation

Open Issue - Is dynamic skew compensation mandatory or optional?

Open Issue - What is the technique for skew compensation during Idle micro-packets?

Open Issue - How often are you required to send an Idle micro-packet?

13 Copper interface (optional)

13.1 General

Open Issue - Width of interface – 16+4+1+1

Open Issue - Baud rate, tolerance

Open Issue - What is the shape of the FRAME signal?

13.2 Electrical output interface

Open Issue - Voltage, current, risetime specifications.

Open Issue - What is the output skew specification?

Open Issue - What coupling is used to cable – capacitor? Transformer? What are the parameters?

13.3 Electrical input interface

Open Issue - Input voltage levels

Open Issue - What is the input skew specification?

Open Issue - What coupling is used to cable – capacitor? Transformer? What are the parameters?

13.4 Electrical connector

Open Issue - Connector pin assignments

Open Issue - Connector specifications?

Open Issue - Different connectors at Source and Destination? Same with keying?

13.5 Cable specifications

Open Issue - Cable type, number of conductors

Open Issue - Cable length – max., min.

Open Issue - Cable electrical specs. – impedance, loss, relative skew, ...

Open Issue - Cable mechanical – shielding, size, material

13.6 Grounding

Open Issue - Ground shield at Source? Destination? Both?

Open Issue - What to do with unused conductors? Unused pins?

13.7 Cable termination

Open Issue - Termination value

Open Issue - Are terminators used on all cables? A function of length?

Open Issue - Termination location – at receiver? in connector?

14 Optical interface (optional)

14.1 General

Open Issue - Width of interface – 8+2+1+1?

Open Issue - Signalling frequency, tolerance

Open Issue - What is the shape of the FRAME signal?

14.2 Optical output interface

Open Issue - Optical parameters – wavelength, power, …

Open Issue - What is the output skew specification?

14.3 Optical input interface

Open Issue - Optical parameters – power...

Open Issue - What is the input skew specification?

14.4 Optical connector

Open Issue - Connector pin assignments

Open Issue - Connector specifications?

Open Issue - Different connectors at Source and Destination? Same with keying?

Open Issue - Field termination of connectors

14.5 Optical cable specifications

Open Issue - Cable type, number of fibers

Open Issue - Cable length

Open Issue - Cable optical specs. – loss, crosstalk, relative skew, ...

Open Issue - Cable mechanical – fiber pitch, size, jacket material

15 Mapping to HIPPI-800

I have a proposal that has been accepted in principal by the working group, but have not had time to add in the text. It will be included in a future version of the document. D. Tolmie

Open Issue - Are there limitations to HIPPI-FP?

Open Issue - Should a connection header be on every message across the connection-oriented VC?

Open Issue - Are there extra parameters when going from HIPPI-800 to -6400? If so, what do we do with them?

Open Issue - Are there extra parameters in HIPPI-FP, or HIPPI-SC, that are not in HIPPI-6400-PH? If so, what do we do with them?

Open Issue - Are there extra parameters in HIPPI-6400-PH, that are not in HIPPI-FP? If so, what do we do with them?

Open Issue – Should the mapping to HIPPI-800 information go in the body of this document? in an annex? in a different document?

xx Other Open Issues

Open Issue - Should we have an annex describing the use of HIPPI-6400-PH links in networks? It could show a group of links crossing a switch and illustrate how the VC is unchanged through the fabric, and most of the other parameters are local to the links.